

Two-stage Hidden Markov Model in Gesture Recognition for Human Robot Interaction

Regular Paper

Nhan Nguyen-Duc-Thanh¹, Sungyoung Lee² and Donghan Kim^{3*}

¹ Department of Electronics and Radio Engineering, Kyung Hee University, Korea

² Department of Computer Engineering, Kyung Hee University, Korea

³ Department of Electronics and Radio Engineering, Kyung Hee University, Korea

* Corresponding author E-mail: donghani@khu.ac.kr

Received 02 May 2012; Accepted 26 May 2012

DOI: 10.5772/50204

© 2012 Nhan et al.; licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/3.0>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract Hidden Markov Model (HMM) is very rich in mathematical structure and hence can form the theoretical basis for use in a wide range of applications including gesture representation. Most research in this field, however, uses only HMM for recognizing simple gestures, while HMM can definitely be applied for whole gesture meaning recognition. This is very effectively applicable in Human-Robot Interaction (HRI). In this paper, we introduce an approach for HRI in which not only the human can naturally control the robot by hand gesture, but also the robot can recognize what kind of task it is executing. The main idea behind this method is the 2-stages Hidden Markov Model. The 1st HMM is to recognize the prime command-like gestures. Based on the sequence of prime gestures that are recognized from the 1st stage and which represent the whole action, the 2nd HMM plays a role in task recognition. Another contribution of this paper is that we use the output Mixed Gaussian distribution in HMM to improve the recognition rate. In the experiment, we also complete a comparison of the different number of hidden states and mixture components to obtain the optimal one, and compare to other methods to evaluate this performance.

Keywords Hidden Markov Model (HMM), Human-robot Interaction (HRI), gesture recognition, Conditional Random Field (CRF), natural interaction

1. Introduction

Human-robot interaction (HRI) is the interdisciplinary study of interaction dynamics between humans and robots, the study of how humans interact with robots, and how best to design and implement robot systems capable of accomplishing interactive tasks in human environments. Much research specializing in HRI comes from a variety of fields including electronics, mechanics, industry, design, human-computer interaction, artificial intelligence, robotics, natural language understanding and computer vision. In robotics, the major goal is to place some kind of robotic assistant or companion in the normal home environment for people to be able to communicate with in a human-like fashion. The challenging problem for researchers is making the robot very similar to humans and making friendlier human-

robot interactions. Over the last decade, the “seeing”, as the most prominent and equally important modality for *natural interaction*, is becoming a major research issue.

Visual recognition of human actions provides a bridge for a non-verbal as well as verbal communication between a human and the robot, both of which are highly ambiguous. Gesture recognition plays an important role and much attention is paid to this area. It enables the robot's anticipation of human actions leading to proactive robot behaviour, especially in passive, more observational situations. Gesture recognition is being widely used compared to other methods such as sound or touch communication. In gesture recognition, the current approaches try to deal with more diverse gestures. This leads to time consuming and low recognition rate problem.

In order to solve the two above problems, there is much current research into different methods such as HMM, CRF (conditional random field), particle filtering and condensation, finite-state machine as statistical modelling, optical flow, skin colour, connectionist model, etc.

Among them, HMM is the most frequently used tool for gesture recognition due to its advantages ([2-7]). HMM has the ability to model a time-domain process that demonstrates a Markov property that is useful in making assumptions, when considering the positions and orientations of gestures through time. HMM is rich in mathematical structures and has been found to efficiently model spatial-temporal information in a natural way. Therefore, this tool has been successfully applied for spatial-temporal objects such as speech recognition, protein modelling and gesture recognition.

Another method is the Particle Filtering and Condensation Algorithm ([15,16]). The particle filters have been very effective in estimating the state of dynamic systems from sensor information. The Condensation Algorithm (Conditional Density Propagation Over Time) makes use of random sampling in order to model arbitrarily complex probability density functions. That is, rather than attempting to fit a specific equation to observed data, it uses N weighted samples to approximate the curve described by the data. Each sample consists of a *state* and a *weight* proportional to the probability that the state is predicted by the input data. As the number of samples increases, the precision with which the samples model the observed probabilistic density function (*pdf*) increases. However, this tool has limitations in the case of various prediction motion models.

Another well-known model used in gesture recognition is Conditional Random Fields (CRF). These models are discriminative models that are undirected graphical models and were developed for labelling data. CRF use

an exponential distribution to model the entire sequence given the observation sequence. Instead of constructing each model for each class like HMM (Generative Model), and then maximizing the likelihood of generating all the examples of a given gesture class, which is not necessarily optimal for discriminating the gesture class against other gestures, CRF is a single model for the joint probability of the sequences $p(y|O, \theta)$, which are initially built without labels for non-gesture patterns. Moreover, CRF has the advantages of avoiding the requirement of conditional independence of observations and the label bias problem [10]. In addition, the discriminative model can be modified into Hidden-CRF ([9,11,13]) where the underlying graphical model captures spatial dependencies between hidden object parts. We will discuss different models in detail in session III.

On the other hand, gesture recognition for a mobile robot imposes several requirements on the system corresponding to a specific robot. The system should be fast enough to fit in the real-time mobile robot's environment and the kind of gesture should be suitable for a robot. For example, a gesture for controlling a cleaning robot should be simple and a hand command-like gesture (turn left, turn right, moving, rotating...) while a gesture for interacting with a humanoid robot should be like a human action (walking, sitting, beating, jumping...). Most research in HRI uses the above methods, however, they just deal with simple gestures or one of a few different kinds of gestures. In order to solve this limitation, we propose an approach in which the *2-stages Hidden Markov Model* is responsible for both recognizing the prime command-like gestures and task recognition.

The remainder of this paper is organized as follows. Section II is some related works. Section III shows the backgrounds directly related to our approach. Section IV explains the main system 2-stage HMM. Section V is the experiments, results and some evaluations. Section VI concludes this paper.

2. Related Work

There are many gesture recognition systems for HRI, however, they can be categorized into two kinds: Template Matching-based and State-space-based approaches. Template matching-based is not really suitable for gesture recognition. It is based on the spatial distance between template and input data while the gesture is an object in the domain of temporal variability. In recent years, the second approach has been most frequently used and developed.

Yang et al. [2] introduced a method for recognition of whole-body key gestures in HRI. They extracted 13 feature points to represent the whole body gesture and

then mapped to codeword of HMM for recognizing 10 gestures with an accuracy of 94.9%. A design of transition model was proposed to accurately spot key gestures from the continuous motion of the human body. In order to reduce the complexity of the model, data-dependent statistics and relative entropy were applied to cut down the number of states. Elmezain et al. [7] also used a similar approach to Yang, however, they extended the model by increasing the weights of self-transition feature functions to improve the accuracy with CRFs. The accuracies obtained are 93.31% and 90.49% corresponding to the HMM and CRF methods.

Over the last decade, there has been some research which tried to improve the accuracy of the HMM method. Andrew et al. [3] extended the standard hidden Markov model method of gesture recognition to include a global parametric variation in the output probabilities of the states of HMM. They defined a parameterized gesture in which the output densities are the function of the parameter vector $b_j(x_i; \theta)$. This leads the parametric HMM to handle arbitrary smooth dependencies.

Recently, there has been increasing interest in using CRFs in the vision community. Sminchisescu et al. [17] introduced conditional graphical models as complementary tools for human motion recognition and present an extensive set of experiments that show how these typically outperform HMMs in classifying not only diverse human activities like walking, jumping, running, picking or dancing, but also for discriminating among subtle motion styles like normal walk and wander walk. Wang et al. [10] constructed a discriminative sequence model with hidden state and called it the Hidden CRF Model. They implemented three different kinds of model to compare the performance. The results pointed out that HCRF reached the best outcome with 71.88% (percentage accuracy) while the outcome of HMM and CRF are 65.33% and 66.53% respectively. They also conducted the experiment with two kinds of gesture datasets - head and arm gesture dataset. Prior to this Quattoni et al. [13] proposed an extension of the CRF framework to recognize objects in a car with low accuracy (60%). Lafferty et al. used CRF to solve the label bias problem. Vinh L. et al. [12] proposed an implementation of the Semi-Markov CRF and outperformed the previous work in terms of computation complexity. The average precision was 88.47%. Generally, the CRF method solved the limitations of the HMM method, that is the requirement of conditional dependence of observations, but the class of CRF is much more expensive, because it allows arbitrary dependencies on the observation sequence. That is the reason why the recognition rate of CRF is generally lower than HMM.

3. Background

In this section, we briefly review the theory of Hidden Markov Model and some problems related to the training process and human gesture representation. We also point out the limitation of previous works in gesture recognition.

3.1 Conventional Hidden Markov Model

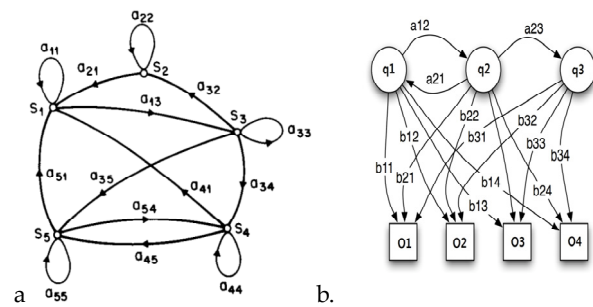


Figure 1. a. A Markov chain with 5 states and selected transitions. b. Compact HMM

The conventional HMM is expressed as the following [6]. HMM is the mathematical tool to model signals, objects...that have the temporal structure and follow the Markov process. HMM can be described compactly as $\lambda = (A, B, \pi)$ (Figure 1b) where,

$A = \{a_{ij}\}$: the state transition matrix

$$a_{ij} = P[q_{t+1} = s_j | q_t = s_i], 1 \leq i \leq N \quad (1)$$

$B = \{b_j(k)\}$: the observation symbol probability distribution

$$b_j(k) = P[O_t = v_k | q_t = s_j], \\ 1 \leq j \leq N, 1 \leq k \leq M \quad (2)$$

$\pi = \{\pi_i\}$: the initial state distribution

$$\pi_i = P[q_1 = s_i] \quad (3)$$

Set of states: $S = \{s_1, s_2, \dots, s_N\}$

State at time t: q_t

Set of symbols: $V = \{v_1, v_2, \dots, v_M\}$

Given the observation sequence $O_1^T = O_1 O_2 \dots O_T$ and a model $\lambda = (A, B, \pi)$, how do we efficiently compute $P(O | \lambda)$, i.e., the probability of the observation sequence given the model. We have to do two steps:

Training: based on the input data sequences $\{O\}$, we calculate and adjust $\lambda = \bar{\lambda}$ to maximize likelihood $P(O | \lambda)$

Recognizing: based on $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$ for each class, we will assign the class in which the likelihood $P(O|\lambda)$ is maximized.

The observation symbol probability distribution $P[O_t = v_k | q_t = s_j]$ can be discrete symbols or continuous variables. If the observations are discrete symbols, we can represent the observation model as a matrix:

$$B(i, k) = P(O_t = k | q_t = s_i) \quad (4)$$

If the observations are vectors in R^L , it is common to represent $P[O_t | q_t]$ as a Gaussian:

$$P[O_t = y | q_t = s_i] = N(y; \mu_i, \Sigma_i) \quad (5)$$

$$N(y; \mu, \Sigma) = \frac{1}{(2\pi)^{L/2} |\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(y - \mu)^T \Sigma^{-1}(y - \mu)\right] \quad (6)$$

A more flexible representation is a mixture of M Gaussians:

$$P[O_t = y | q_t = s_i] = \sum_{m=1}^M P(M_t = m | q_t = s_i) \times N(y; \mu_{m,i}, \Sigma_{m,i}) \quad (7)$$

where M_t is a hidden variable that specifies which mixture component to use and $P(M_t = m | q_t = s_i) = C(i, m)$ is the conditional prior weight of each mixture component. In our approach, we both implement continuous and discrete output variable distribution for 1st and 2nd HMM stages respectively. The output distribution problem is going to discuss in detail the model training part.

3.2 Extended HMM

In order to solve the limitation of conventional HMM, high temporal correlations in the signal are not fully captured, since data are supposed to depend only on current states; some improved approaches were constructed such as: Hierarchical HMM, Semi-HMM, 2D HMM, Factorial HMM, Coupled HMM, Asynchronous IO-HMM. Hierarchical HMM (HHMM) is an extension of the HMM that is designed to model domains with hierarchical structure and/or dependencies at multiple length/time scales. In an HHMM, the states of the stochastic automaton can emit single observations or strings of observations. Those that emit single observations are called “production states” and those that emit strings are termed “abstract states”. The strings emitted by abstract states are themselves governed by sub-HHMMs, which can be called recursively. When the sub-HHMM is finished, control is returned to wherever it was called from; the calling context is memorized using a depth-limited stack.

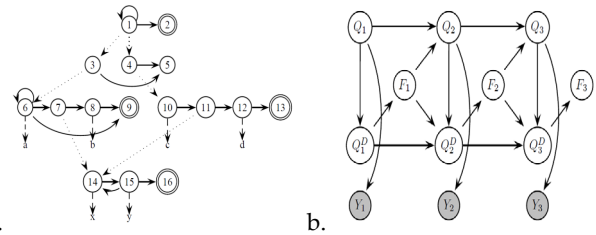


Figure 2. a. State transition diagram for a four-level HHMM. b. A variable-duration HMM modelled as a 2-level HHMM

An HHMM cannot make a horizontal transition before it makes a vertical one; hence it cannot produce the empty string. For example, in Figure 2a, it is not possible to transition directly from state 1 to 2. While HHMMs are less powerful than stochastic context-free grammars (SCFGs) and recursive transition networks, both of which can handle recursion to an unbounded depth, unlike HHMMs, they are sufficiently expressive for many practical problems, which often only involve tail-recursion (i.e., self transitions to an abstract state). Furthermore, HHMMs can be made much more computationally efficient ($O(T^3)$) than SCFGs.

A Semi-Markov HMM (more properly called a Hidden Semi-Markov Model or HSMM) is like an HMM except each state can emit a sequence of observations. The probability we remain in state i for exactly d steps is $p(d) = (1 - p)d^{d-1}$, where a_{ii} is the self-loop probability. It is called semi-Markov because to predict the next state, it is not sufficient to condition on the past state: we also need to know how long we have been in that state. We can model this as a special kind of 2-level HHMM, as shown in Figure 2b.

3.3 Training Model

The goal in HMM training is to determine model parameters, the transition probabilities a_{ij} and b_{jk} , from an ensemble of training samples. There is no known method for obtaining the optimal or most likely set of parameters from data, but we can nearly always reach a good solution using a straightforward technique. As mentioned in the previous part, we have to maximize likelihood:

$$P(\Omega | \lambda) = \prod_{n=1}^S P(O^n | \lambda) \quad (8)$$

Where,

Ω : set of training sequences, n : size of training set.

Consider an input sequence of length T : $O^T = O_1 O_2 \dots O_T$, we have:

$$P(O^T | \lambda) = \sum_{r=1}^{r_{\max}} P(O^T | q_r^T) P(q_r^T) \quad (9)$$

Where each r indexes a particular sequence $q_r^T = \{q(1), q(2), \dots, q(T)\}$ of T hidden states. $r_{\max} = N^T$ is possible terms in (9), corresponding to all possible state sequences of length T . Because we are dealing with a first-order Markov process, the second factor can be rewritten as:

$$P(q_r^T) = \prod_{t=1}^T P(q(t) | q(t-1)) \quad (10)$$

On the other hand, our assumption is that the output probabilities depend only upon the hidden state; we can write the first factor as

$$P(O^T | q_r^T) = \prod_{t=1}^T P(O(t) | q(t)) \quad (11)$$

Substitute (10), (11) into (9), we have

$$P(O^T | \lambda) = \sum_{r=1}^{r_{\max}} \prod_{t=1}^T P(O(t) | q(t)) P(q(t) | q(t-1)) \quad (12)$$

Now, we use the Forward-Backward or Baum-Welch algorithm, an instance of a generalized Expectation-Maximization Algorithm to train the model to reach the maximum likelihood. The general approach will be to iteratively update the weights. We do this by defining:

$$\alpha_i(t) \stackrel{\text{def}}{=} P(O_{1,t-1}, q_t = s_i) \quad (13)$$

$$\beta_i(t) \stackrel{\text{def}}{=} P(O_{t,T} | q_t = s_i) \quad (14)$$

$\alpha_i(t)$ Represent the probability that model is in state q_i at step t having generated the first t elements of sequence O^T . $\beta_i(t)$ represent probability that the model is in state $q_i(t)$ and will generate the remainder of the given target sequence, i.e., from $t+1 \rightarrow T$. Based on the definition above, we easily infer:

$$\begin{aligned} \alpha_i(1) &= \pi_i \\ \alpha_j(t) &= \sum_i \alpha_i(t-1) a_{ij} b_{jO_t} \end{aligned} \quad (15)$$

$$\begin{aligned} \beta_i(T) &= 1 \\ \beta_i(t) &= \sum_j \beta_j(t+1) a_{ij} b_{jO_t} \end{aligned} \quad (16)$$

We can also derive,

$$\begin{aligned} P(O | \lambda) &= \sum_{i=1}^N \alpha_i(T) \\ &= \sum_{i=1}^N \beta_i(1) = \sum_{i=1}^N \alpha_i(t) \beta_i(t) \end{aligned} \quad (17)$$

The probability of being in state s_i at time t , and state s_j at time $t+1$, given the model and the observation sequence, i.e.,

$$\xi_{ij}(t) = P(q_t = s_i, q_{t+1} = s_j | O, \lambda) \quad (18)$$

From definition of the forward-backward algorithm, we can rewrite $\xi_{ij}(t)$ in the form:

$$\begin{aligned} \xi_{ij}(t) &= \frac{\alpha_i(t) a_{ij} b_{jO_{t+1}} \beta_j(t+1)}{P(O | \lambda)} \\ &= \frac{\alpha_i(t) a_{ij} b_{jO_{t+1}} \beta_j(t+1)}{\sum_{i=1}^N \sum_{j=1}^N \alpha_i(t) a_{ij} b_{jO_{t+1}} \beta_j(t+1)} \end{aligned} \quad (19)$$

And the probability of being in state s_i at time t :

$$\gamma_i(t) = \sum_{j=1}^N \xi_{ij}(t) \quad (20)$$

By maximizing (using standard constrained optimization techniques) Baum's auxiliary function

$$Q(\lambda, \bar{\lambda}) = \sum_Q P(Q | O, \lambda) \log[P(Q, O | \bar{\lambda})] \quad (21)$$

over $\bar{\lambda}$. It has been proven by Baum and his colleagues [18] that maximization of $Q(\lambda, \bar{\lambda})$ leads to increased likelihood, i.e.,

$$\max_{\bar{\lambda}} [Q(\lambda, \bar{\lambda})] \Rightarrow P(O | \bar{\lambda}) \geq P(O | \lambda) \quad (22)$$

we can get the estimation of an HMM. A set of the reasonable re-estimation formulas for π , A and B are

$$\begin{aligned} \bar{\pi} &= \text{expected frequency in state } i \\ \text{at time } (t=1) &= \gamma_i(1) \end{aligned} \quad (23)$$

$$\begin{aligned} a_{ij} &= \frac{\# \text{ of transitions from state } i \text{ to } j}{\# \text{ of transitions from state } i} \\ &= \frac{\sum_{t=1}^T \xi_{ij}(t)}{\sum_{t=1}^T \gamma_i(t)} = \frac{\sum_{t=1}^T \xi_{ij}(t)}{\sum_{j=1}^N \sum_{t=1}^T \xi_{ij}(t)} \end{aligned} \quad (24)$$

$$\begin{aligned} b_{jk} &= \frac{\# \text{ of times in state } j \text{ and symbol } v_k}{\# \text{ of times in state } j} \\ &= \frac{\sum_{t=1}^T \gamma_j(t)}{\sum_{t=1}^T \gamma_j(t)} = \frac{O_t = v_k}{\sum_{t=1}^T \gamma_j(t)} \end{aligned} \quad (25)$$

4. 2-Stage HMM

In this section, we present details of our proposed model, the 2-stage HMM with full-covariance Gaussian output distributions (FCGD), applied to recognize 10 prime gestures for human-robot interaction. In HRI, hand gesture plays an important role and is a major representation of human activity. We conduct our system using this assumption.

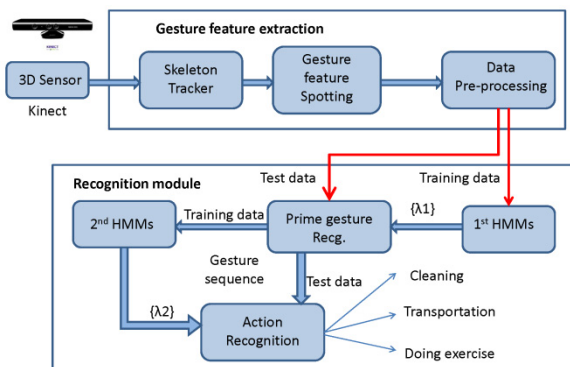


Figure 3. Proposed framework

A simplified scheme of the approach is given in Figure 3. The first step is to detect the human skeleton from the sequence of image frames. The camera used is the 3D camera Kinect, released by Microsoft Corp. The randomized decision tree and forest method [19] is applied to get the whole skeleton. The skeleton is then processed and the skeletal feature vector is obtained for the training and recognizing process. Although the randomized decision tree and forest method track the whole skeleton, in this approach we just use two hands and two elbows from the skeleton for recognition. (respectively shown in Figure 4 a, b).

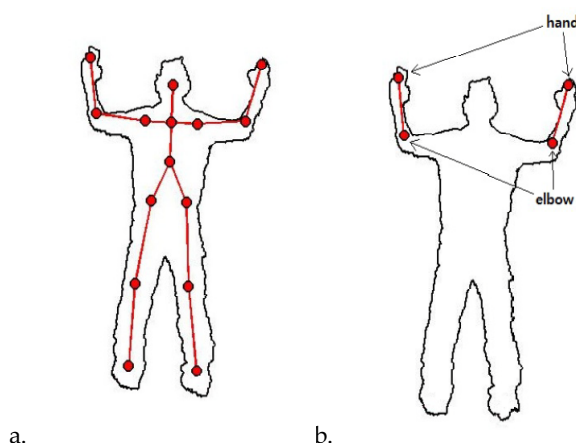


Figure 4. a. Human body skeleton representation. b. Hand and elbow position for recognition

The first stage plays a role in recognizing the prime gesture. The prime gesture is applied to control the robot, therefore, they are simple, command-like gestures. The

information of skeleton points in 3-D allows us to calculate and represent the feature points. In terms of gesture, we extract the feature vector based on the relative difference between the current position and the starting position of the gesture. The feature vector, observation in HMM, includes eight components:

$$O = (d_1, d_2, d_3, d_4, direction_1, direction_2, direction_3, direction_4).$$

Where, d_1, d_2, d_3, d_4 : are distance from current position to starting point position of left hand, left elbow, right hand, right elbow respectively

$direction_1, direction_2, direction_3, direction_4$: are the direction of current motion at left hand, left elbow, right hand, right elbow respectively.

The data stream going in 2nd HMM is the sequence of recognized gestures, marked 1, 2 ... 10 (totally we have 10 prime gestures). Therefore, the observation of the 2nd stage is considered to be discrete and univariate distribution (1-D).

In our work, we implement two observation densities: continuous and discrete distribution corresponding to 1st and 2nd stage. In continuous distribution case, for solving the limitation of gesture recognition which assumes all gesture feature variation are NOT correlated with each other (pair-wise independent), we propose our HMM which is able to explicitly utilize a mixture of full-covariance Gaussian distributions (FCGM-HMM) instead of the diagonal covariance Gaussian mixture hidden Markov model (DCGM-HMM). Thus, (5), (6) can be rewritten as follow:

$$b_j(O) = \sum_{m=1}^M c_{jm} N(O, \mu_{jm}, \Sigma_{jm}), 1 \leq j \leq N \quad (26)$$

With the constraints,

$$\begin{aligned} \sum_{m=1}^M c_{jm} &= 1 \leq j \leq N \\ c_{jm} &\geq 0; 1 \leq j \leq N, 1 \leq m \leq M \end{aligned} \quad (27)$$

The re-estimation formulas for the coefficients of mixture density, i.e., c_{jk} , μ_{jk} and Σ_{jk} are of the form

$$\bar{c}_{jk} = \frac{\sum_{t=1}^T \gamma_{jk}(t)}{\sum_{t=1}^T \sum_{k=1}^M \gamma_{jk}(t)} \quad (28)$$

$$\bar{\mu}_{jk} = \frac{\sum_{t=1}^T \gamma_{jk}(t) \cdot O(t)}{\sum_{t=1}^T \gamma_{jk}(t)} \quad (29)$$

$$\bar{\Sigma}_{jk} = \frac{\sum_{t=1}^T \gamma_{jk}(t) \cdot [O(t) - \mu_{jk}] [O(t) - \mu_{jk}]^T}{\sum_{t=1}^T \gamma_{jk}(t)} \quad (30)$$

Where $\gamma_{jk}(t)$ is the probability of being in state j at time t with k -th mixture component accounting for $O(t)$.

The 2nd stage plays a role in recognizing the whole task based on the sequence of prime gestures. They are marked 1, 2,...,10 (totally we have 10 prime gestures). Therefore, the observation of the 2nd stage is considered to be discrete and univariate distribution (1-D).

$$\gamma_{jk}(t) = \left[\frac{\alpha_j(t) \beta_j(t)}{\sum_{j=1}^N \alpha_j(t) \beta_j(t)} \right] \times \left[\frac{c_{jk} N[O(t), \mu_{jk}, \Sigma_{jk}]}{\sum_{m=1}^M c_{jm} N[O(t), \mu_{jm}, \Sigma_{jm}]} \right] \quad (31)$$

Now, we want to talk about the covariance matrix of the output distribution of the 1st stage in detail. As we know, whenever we want to act, talk to others or try to express something, two hand gesture always go together with this. Eight components mentioned above harmonically change to represent what we are doing or what the action is. For example, if we are jogging, the hands together with the elbows and the shoulders make a fixed angular shape and they move up and down together at the same velocity. Moreover, hands, elbows and shoulders are directly connected to be arm. These obviously point out that the eight components correlate each other. That is the reason why the model for that is constructed without considering the full covariance matrix is limited and not reality. To overcome this limitation, our approach takes into account the *full-covariance Gaussian distributions* (FCGD) Σ_{jm} of the output.

5. Experiments and Evaluations

To evaluate our approach overall, we carried out three kinds of experiments. The first experiment aims to obtain the optimal number of hidden states and mixture components. The second one is to compare the recognition rate of the first stage with that of the other method that uses the *diagonal covariance Gaussian mixture hidden Markov model* (DCGM-HMM). Last but not least, we do the whole experiment for the 2-stage HMM on iRobot Create to classify two kinds of tasks: cleaning and transporting.

5.1 Number of states and mixture components

Because there is no algorithm to get the optimal number of hidden states and mixture components, in order to reach as good a performance as possible, we increase the number of states and mixture components for each of experiments. For the training dataset, we use 50 samples for each gesture. The model parameters are derived through the training process using the EM method with FCGD. The number of iterations for class training is 50 at maximum. The loop will end whenever the logarithm of likelihood does not increase. All the experiments were implemented in Visual C++ and Matlab R2010a. As Table 1 shows, the optimal number of states and mixture components are five and five respectively. This experiment is conducted with 20 samples for each gesture class. Generally, these numbers are different in different datasets because they are directly affected by observation distribution.

No. of states	Number of mixture components					
	1	2	3	4	5	6
2	74	83	86.5	88	89.5	89
3	79	88.5	90	90	91.5	92
4	83	86	91.5	92	93	92
5	87	90.5	90	92.5	95.5	90
6	87.5	87	89	90	94.5	88.5

Table 1. The average recognition rate with different number of hidden states and mixture components

5.2 Experiment for first stage

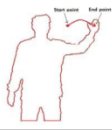

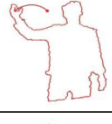
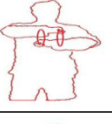
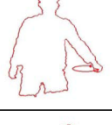
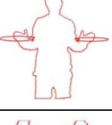
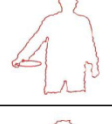
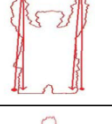
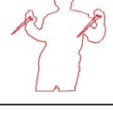
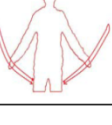
Gesture	Description	Gesture	Description
	Turn right (1)		Move zigzag (6)
	Turn left (2)		Rotation(7)
	Turn right motor on (3)		Speed down (8)
	Turn left motor on (4)		Speed up (9)
	Move ahead (5)		Stop (10)

Figure 5. Prime gesture and description

Given gestures		Recognized gestures										Precision (%)
Gesture label	Number of gestures	1	2	3	4	5	6	7	8	9	10	
1	30	29	0	1	0	0	0	0	0	0	0	96.67
2	30	0	28	0	1	0	0	1	0	0	0	93.33
3	30	1	0	29	0	0	0	0	0	0	0	96.67
4	30	0	2	0	28	0	0	0	0	0	0	93.33
5	30	0	0	0	0	30	0	0	0	0	0	100
6	30	0	0	0	0	0	30	0	0	0	0	100
7	30	0	0	0	0	0	0	30	0	0	0	100
8	30	0	0	0	0	0	0	0	27	2	1	90
9	30	0	0	0	0	0	0	0	0	30	0	100
10	30	0	0	1	0	0	0	0	2	2	25	83.33
Total	300											95.33

Table 2. Prime gesture recognition result of our approach

After obtaining the optimal number of states and mixture components, we conduct the experiment to recognize 10 kinds of gestures with 30 gestures for each of them. The gestures are described in detail in Figure 5. Our gesture vocabulary is chosen by regarding *command-like gestures* which are suitable for robot control. The gesture descriptions are concerned with iRobot control. As the result show in Table 2, the recognition rate ranges from 83.33 % to 100 %. The accuracy average is 95.33 %. In order to show the performance of our method, we compare the recognition rate in two cases: DCGD and FCGD (Figure 6)

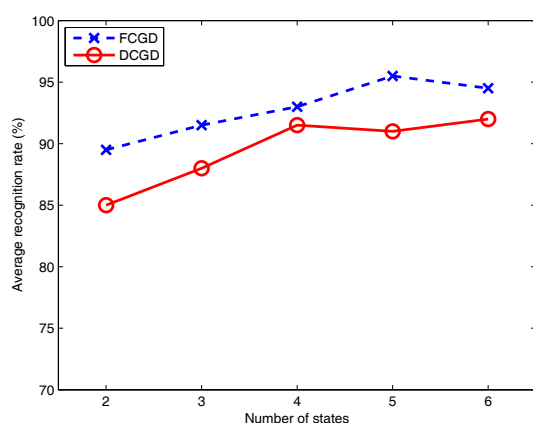


Figure 6. Comparison chart for DCGD and FCGD

5.3 iRobot Create experiment

In this section, we evaluate the whole model based on implementation on iRobot Create. The hardware setup includes an iRobot Create connected to a computer via a Bluetooth device. The Kinect camera is connected to the computer through a USB interface. The program is implemented on Matlab for data processing and the

computation part, while Visual C++ is used for robot control (Figure 7). The experiment is carried out using the following process: the user performs the prime gesture that is pre-defined to be recognized in the 1st stage. The control signal is then sent to the robot. This process will be repeated until the robot finishes the whole task. Based on the training of the 2nd stage, the system will determine which task the robot should execute. We let the robot do 40 tasks belonging to two kinds: cleaning and transporting with 358 prime gestures in total. The results for task recognition are shown in Table 3. This experiment focuses on task recognition instead of prime gesture recognition.

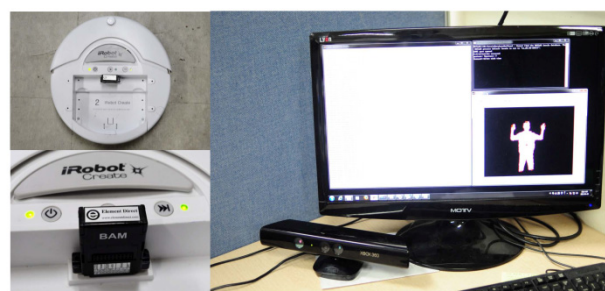


Figure 7. Hardware setup

Given task	Recognized task	
	Cleaning	Transporting
Cleaning	19	1
Transporting	1	19

Table 3. Task recognition result on iRobot

Finally, we want to discuss computational complexity of our model compared with 1-stage HMM. Much research on gesture or activity recognition followed the conventional form of HMM in which the recognized object is treated as one observation sequence (1-stage).

The complexity was computed as $O(N^2T)$ (*) (the forward-backward algorithm in the training process). Where,

N : Number of states

T : Length of observation

We divided our approach into two stages using these notations:

N_1, N_2 : Number of states of stage 1 and stage 2.

T_1, T_2 : Length of observation of stage 1 and stage 2.

under the assumptions:

$$N \geq \frac{(N_1 + N_2)}{\gamma}, T = T_1 T_2 \quad (32)$$

These assumptions are reasonable with γ, N_1, N_2 are determinable given N . The complexity is proportional of $O(T_2 \cdot N_1^2 T_1 + N_2^2 T_2)$. (*) can be rewritten as:

$$\begin{aligned} O(N^2 T) &\geq O\left[\left(\frac{N_1 + N_2}{\gamma}\right)^2 T_1 T_2\right] \\ &= O\left[\frac{N_1^2 T_1 T_2 + N_2^2 T_1 T_2 + 2N_1 N_2 T_1 T_2}{\gamma^2}\right] \\ &\approx O(N_1^2 T_1 T_2 + N_2^2 T_1 T_2 + N_1 N_2 T_1 T_2) \\ &\geq O(T_2 \cdot N_1^2 T_1 + N_2^2 T_2) \end{aligned} \quad (33)$$

6. Conclusions

In this paper, we have presented an approach for HRI using human gesture. The 2-stage HMM is implemented for the robot to recognize prime gestures and then classify what kind of task the robot should execute. In the gesture recognition part, we improved the recognition rate with full-covariance Gaussian distributions (FCGD) output of HMM. This outperforms the previous research with an average precision of 95.33% in the deployment part. We use 10 kinds of gestures to control the iRobot and simultaneously recognize the whole task. Moreover, we evaluate the advantage of our model in terms of computational complexity compared to the 1-stage HMM. Our work therefore might bring a significant contribution not only to gesture recognition, but also to the HRI area.

For future work, we will extend our gesture database to different kinds of robots: humanoid robot, teaching robot. We will also use multiple sensors, such as audio sensor and tactile sensor, for diversification of input and to make the robot HRI friendlier. Moreover, the whole body gesture and auto-recognized multi-gesture problem will be considered in near-term future work.

7. Acknowledgments

This research is supported by MKE (Ministry of Knowledge Economy), Korea, supervised by the NIPA Program (National IT Industry Promotion Agency) (NIPA-2009-(C1090-0902-0002)), the Industrial Strategic Technology Development Program (No. 10035544-2010-01), Transportation and Maritime Affairs (MLTM) of Korean government and the Industrial Strategic Technology Development Program (No. 10035544-2010-01) funded by the Ministry of Knowledge Economy (MKE) Korea and a grant from Construction Technology Innovation Program (CTIP) funded by Ministry of Land.

8. References

- [1] Mitra S, Tinku A (2007) Gesture recognition: A survey. IEEE Trans. Systems, man, and cybernetics, Vol. 37, No. 3.
- [2] Yang H.D, Park A.Y, Lee S.Y, (2007) Gesture spotting and recognition for Human-Robot Interaction. IEEE Trans. Robotics, Vol. 23, No. 2.
- [3] Wilson A.D, Bobick A.F (1999) Parametric Hidden Markov models for gesture recognition. IEEE Trans. Pattern analysis and machine intelligence, Vol. 21, No. 9.
- [4] Nam Y, Wohn K, (1997) Recognition of hand gestures with 3D, nonlinear arm movement. Pattern recognition letters, Vol. 18, 105-113.
- [5] Park Y.S, Kim E.Y, Jang S.S, Park S.H, Park M.H, Kim H.J (2005) HMM-Based gesture recognition for Robot control. LNCS 3522, pp. 607-614.
- [6] Nam Y, Wohn K (1996) Recognition of Space-Time hand-gestures using hidden Markov model.
- [7] Elmezain M, Ayoub A.H, Sadek S, Michaelis B, (2010) Robust methods for hand gesture spotting and recognition using hidden Markov model and conditional random fields. IEEE Proc. International symposium on, pp 131-136.
- [8] Yang J, Xu Y.S (1994) Hidden Markov model for gesture recognition. Master thesis, The Robotics Institute Carnegie Mellon Univ. Pittsburgh, Pennsylvania 15213.
- [9] Charles S, Andrew Mc (2006) *Introduction to Statistical Relational Learning* MIT Press.
- [10] Wang S.B, Quattoni A, Morency L.P, Demirdjian D, Darrell T (2006) Hidden conditional random fields for gesture recognition. IEEE Proc. Computer society conference, Vol. 2, pp. 1521-1527.
- [11] John L, McCallum A, Pereira F (2001) Conditional random fields: Probabilistic models for segmentation and labelling sequence data. ICML Proc. International conference on machine learning.
- [12] Vinh L.T, Le X.H, Ngo Q.H, Kim H.I, Han M.H, Lee Y.K Lee S.Y (2011) Semi-Markov conditional random fields for accelerometer-based activity recognition. Journal of Applied Intelligence, Vol. 35, No. 2.

- [13] Quattoni A, Collins M, Darrell T (2004) Conditional random field for object recognition. Proc. Conference on Neural Information Processing Systems.
- [14] Liu Y, Shrilberg E, Stolcke A, Harper M (2005) Comparing HMM, maximum entropy, and conditional random fields for disfluency detection. Proc. Conference on Speech Communication and Technology, pp. 3313-3316.
- [15] Arulapalam S, Maskell S, Gordon N, Clapp T (2001) A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking. IEEE Trans. Signal Process., Vol. 50, No. 2, pp. 174-188.
- [16] Kwok C, Fox D, Meila M (2004) Real-time particle filters. Proc. IEEE. Vol. 92, No. 3, pp. 469-484.
- [17] Sminchisescu C, Kanaujia A, Metaxas D (2006) Conditional models for contextual human motion recognition. Computer Vision and Image Understanding. Vol. 104, No. 2-3, pp. 210-220.
- [18] Shotton J, Fitzgibbon A, Cook M, Sharp T, Finocchio M, Moore R, Kipman A, Blake A (2011) Real-Time Human Pose Recognition in Parts from Single Depth Images. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR-2011)*. 21-25 June 2011, Colorado Springs, U.S.A.
- [19] Baum L.E, Petrie T, Soules G, Weiss N (1970) A maximisation techniques occurring in the statistical analysis of probabilistic functions of Markov chains. The Annals of Mathematical Statistics, Vol. 41, No. 1, pp. 164 – 171.