Online Recognition and Segmentation for Time-Series Motion with HMM and Conceptual Relation of Actions

Taketoshi Mori, Yu Nejigane, Masamichi Shimosaka, Yushi Segawa, Tatsuya Harada and Tomomasa Sato
Graduate School of Information Science and Technology
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan
{*tmori, nejigane, simosaka, segawa, harada, tomo*}*@ics.t.u-tokyo.ac.jp*

***Abstract*— In this paper, we propose a robust online action recognition algorithm with a segmentation scheme that detects start and end points of action occurrences. In other words, the algorithm estimates reliably what kind of actions occurring at present time. The algorithm has following characteristics. 1) The algorithm incorporates human knowledge about relation between action names in order to simplify and toughen the algorithm, thus our algorithm can label robustly multiple action names at the same time. 2) The algorithm uses time-series *Action Probability* that represents the likelihood of each action occurrence at every frame time. 3) The classification technique with hidden Markov models (HMMs) enables the algorithm to detect robustly and immediately the segmental points. The experimental results using real motion capture data show that our algorithm not only decreases effectively the latency for detecting the segmental points but also prevents the system from making unnecessary segments due to the error of time-series action probability.**

***Index Terms*— Action Recognition, Segmentation, Hidden Markov Model, Support Vector Machine, Motion Capture**

I. Introduction

Recognizing human action [1], [2], [3] is one of essential foundations to achieve smooth communication between intelligent systems, especially robots, and human. It is also a key technical element in achieving analysis and surveillance of human activity by intelligent systems. In order to achieve immediate support for human life by such systems, it is a very important factor that action recognition runs online. In this context, the meaning of "online" is similar to "filtering", not "smoothing" in signal processing. In other words, the system must recognize action name at the present time from history of the input motion source until the present time.

In order to realize high performance daily action recognition system, we must consider the time-dependency problem. In this context, the time-dependency problem means that human requires a certain time interval to behave that action. This means an action at current frame is dependent on the previous frames and will be the same if the previous frame action does not end.

Our previous work [4] proposed online action recognition with statistical learning techniques, that can react immediately the changes (segmental point) of human action. However, their frame-wise recognition result does not incorporate the property of the time-dependency. Thus its recognition result tends to be unstable. Inamura et al. [5] proposed an elegant action recognition algorithm using hidden Markov models (HMMs). In the system, the input motion should be segmented with a constant intervals before the recognition process executes. Because it is probable that the constant length interval contains the segmental point of actions, such as the transition from walking to running, the system cannot detect immediately the change of human action (segmentation point). Other Markov-based recognition systems [6], [7], [8] has been proposed and has proved to have good performance for detecting the segmental points, however, the assumption in their systems does not fit for the property of great variety of actions. Their main target actions are only dynamic actions, such as walking, jogging and running. They do not focus on static actions like sitting and lying. Thus their system will be poor when the systems include not only dynamic actions but also static actions, such as sitting and lying, because it is hard to optimize the transition probability of Markov process due to the fact that the interval of action occurrence has very wide variety. For example, it is possible for human to behave stable actions such as lying and sitting very long, not temporal. On the other hand, it is impossible for human to behave dynamical actions such as jumping and turning the body in long interval. Thus, the simple Markov-based recognition will fail to recognize action transition or detect the segmental point once the system recognize motion as stable action like lying. This is because the transition probability of Markov process will be very low from lying state to not-lying state when the Markov transition probability is learned with maximum likelihood estimation from real daily life action data.

Consequently, we focus on the start and end point of the target action occurrence, not on the frame-wise transition of actions. Markov-based methods assign the point-wise recognized result by using past recognition result, however, our method assumes the recognition result is the same with the result of the previous frames until the segmental point is detected. Ge [9] proposed semi-Markov models that fits better for detecting segmental point than the simple Markov-based recognizer, however, they limit the interval length a

priori. Thus some stable actions, such as lying and sitting cannot be applied to this method.

In addition to the time-dependency problem, we must consider another property of daily actions: simultaneousness. This means that it is often that multiple actions occurs at the same time. Result like "waving hand while standing" is an example of this phenomena. We categorize daily actions so as to make the system be able to handle multiple action names at the same time.

Consequently, we propose a novel online recognition method that can detect immediately the segmental points of actions. Our method has the following characteristics, 1) the system can label multiple action names at the same time, 2) the system uses the knowledge about categorization of actions in order to simplify the algorithm, 3) the system uses HMMs to detect the segmental points, 4) the system uses SVM-based classification for the input of HMMs.

The rest of the paper proceeds as follows. Section II outlines our proposed algorithm: input / output, what is conceptual relation of daily actions, and processing flow of the algorithm. Section III introduces the details of the algorithm using HMMs, a part of our proposed method in section II. Section IV presents results of several experiments about online recognition and segmentation. We conclude in section V with some directions for future research.

II. Online Recognition and Segmentation with Conceptual Relation of Action

A. Input and Output

The input of the algorithm are time-series data of motion features depicted by $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t$, where $\boldsymbol{x}_t$ represents the motion feature at time t-th frame. $\mathcal{X}_T$ contains a certain length history of the motion features until T-th frame. As an output of the algorithm, l_t represents the action names to be recognized at t-th frame. Thus, the algorithm estimates l_T from $\mathcal{X}_T$. And the algorithm detects the segments of action occurrence at the same time. When $T = 10$, for example, our algorithm assigns $l_{10} = \{\text{sitting, looking away}\}$. And our algorithm may assign $l_{90} = \{\text{standing}\}$ when $T = 90$ (see Fig. 1). The algorithm also chunks actions from segmentation results as $\{\text{looking away}, 1-\alpha, 80\}$, $\{\text{sitting}, 1-\alpha, 40\}$, $\{\text{standing}, 60, 100+\alpha\}$ when $T = 100$, where each parenthesis represents an action segment as $\{action\_name, start\_frame, end\_frame\}$ and α represents any positive value ($100+\alpha$ means the action does not end at the present time, similarly, $1-\alpha$ means that the system cannot understand the start point of the action before the system watches human).

It is not easy to estimate time-series labels and detect the action segments, because it is frequent that the labels assigned with multiple action names. A label $l_{20} = \{\text{looking away, sitting}\}$ in Fig. 1 is an example for that. There are many action names which occur at the same time. They have no

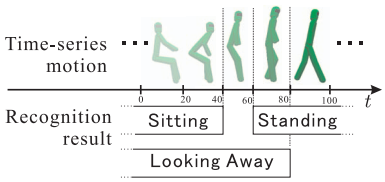

Fig. 1. Input and output of the algorithm is shown. Input: time-series of human motion. Output: segmented recognition results of the system

exclusive relation. In this context, a subjective "exclusive" means one action never occurs when another action occurs. The complexity due to the output labels with multiple action names will be reduced by incorporating human knowledge about daily action. We use knowledge about categorization of actions positively.

B. Conceptual Relation of Action in Daily Life

In this research, we adopt the existing categorization method proposed in our past work [10]. This categorization scheme utilizes the hierarchical relation of action names. For example, *sitting on floor* **is a kind of** *sitting*. This looks like "inheritance" in Object Oriented Programming (OOP). The reason why we adopt this categorization scheme is that an action has exclusive relation to the others in the same group. Specifically, a group of action categorized by gazing full-body posture, which we call root group, contains standing, lying and sitting. Obviously, they have exclusive relation to the others. Thus, it is expected for our algorithm to be simple and robust and the system never output conflicted recognition result when we run and separate the algorithm via each action group. The conflicted result means the misclassified result like "lying on side and sitting". Furthermore, the output of system can be at several levels of details of the recognition.

However, there are groups of actions that do not inherit the root group when we gaze only a part of body, such as neck and arms. For example, look away cannot be a kind of sitting, standing or lying. Thus, we modify the categorization scheme with hard-hierarchical representation of the previous work to semi-hierarchical representation of actions. Specifically, we annotate the relation between groups if and only if there exists hierarchical relation, else they are independent from the others. Fig. 2 shows our categorization scheme (this is the implementation used in the experiments of this paper.)

C. Process for Online Recognition and Segmentation with Semi-Hierarchical Representation

In order to incorporate the property of simultaneous and to exploit fully the knowledge of hierarchical representation of actions, the recognition and segmentation algorithm mentioned in section III runs in parallel with each action groups. Then the system integrates the recognition and segmentation result in each action groups to eliminate the conflict of the result from view of the hierarchical relations. Hence,

the proposed algorithm is summarized as follows. First, the recognition and segmentation algorithm in each group of actions executes. Next, the error correction runs when the recognition and segmentation result in one group conflicts the parental group's result. Specifically our algorithm revises the result of the child group of actions. Fig. 3 shows the processing flow of the our proposed algorithm. In the next section, we will describe the details of the recognition and segmentation method in a group of actions.

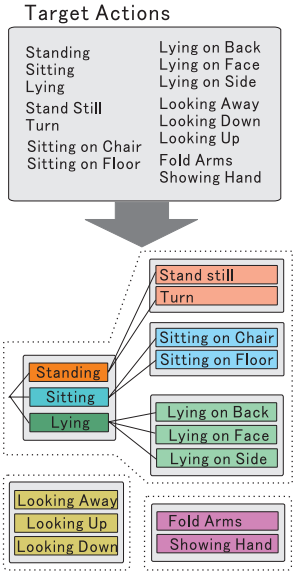

Fig. 2. Semi-Hierarchical Representation of Actions

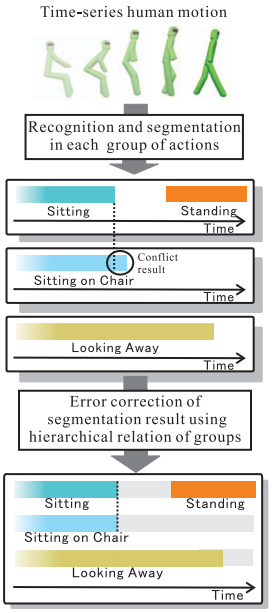

Fig. 3. Recognition and Segmentation Process with Semi-Hierarchical Representation of Actions

III. Online Recognition in a Group of Actions with HMM

In this section, we explain the details of the online recognition method in a group of actions with HMM, a part of our proposed method.

A. Overview

1) Defining Action Probability: The procedure discussed in this section estimates frame-wise action labels and detects action segments for actions in a group. Each action label contains not multiple action names but only one action name because an action has exclusive relation to the others in a group of actions. You may think it is easy to apply some classical multi-class pattern classification algorithms, such as K-nearest neighbor method [11], Gaussian process classification [12], and combinatorial scheme of support vector machines [13] to this problem. This is, however, not a simple matter. It is because it often occurs that the system should not label any action names. We must incorporate the situation of $l_t^{(g)} = \phi$, where g means ID of the action group, $l_t^{(g)}$ depicts the label in g-th group at time t, and $\phi = \mathrm{NULL}$. Thus, the following equation that is often seen in the multi-class classification problems does not satisfy our problem, $\sum_{k=1}^{K} p(l_t^{(g)} = a_k^{(g)}|\mathcal{X}_t) = 1$, where $a_k^{(g)}$ represents k-th, $(k = 1, \ldots, K_g)$ action name in g-th group. Instead, the following equation as $p(l_t^{(g)} = \phi|\mathcal{X}) + \sum_{k=1}^{K} p(l_t^{(g)} = a_k^{(g)}|\mathcal{X}_t) = 1$ satisfies our problem. On the other hand, it is not easy to design $p(l_t^{(g)} = \phi|\mathcal{X}_t)$ in general. Thus we use the following simple but satisfactory equation written as

$$p(l_t^{(g)} = a_k^{(g)}|\mathcal{X}_t) + p(l_t^{(g)} \neq a_k^{(g)}|\mathcal{X}_t) = 1, \ \forall k. \tag{1}$$

In this research, we call this probability function *Action Probability*. In the following, we make a simplified notation $q_t^{(g,k)} = p(l_t^{(g)} = a_k^{(g)}|\mathcal{X}_t)$. From the definition of the *Action Probability*, the minimum risk that the system misclassified the label at time t seems

$$\text{if } q_t^{(g,\lambda)} > 0.5 \text{ then } l_t^{(g)} = a_\lambda^{(g)}, \text{ else } l_t^{(g)} = \phi \tag{2}$$

where $\lambda = \arg\max_k q_t^{(g,k)}$, however, the result of this classification rule will be poor. This is because *Action Probability* at each time does not incorporate the time-series dependency of action occurrence. Hence, our recognition and segmentation method does not use classification rule in (2), but uses sequential pattern of *Action Probability* (see. Fig. 4). The strategy of this procedure is to detect the global change of the value $q_t^{(g,k)}$. Before we mention this, we proceed with the description of the implementation for *Action Probability*.

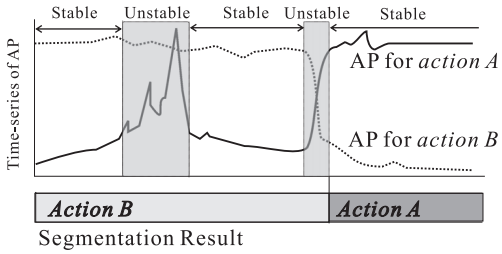

Fig. 4. Recognition and segmentation based on *Action Probability* in a group of actions

2) Implementing Action Probability using SVM: The basic idea is to wrap a binary classification result (whether input motion should be categorized into the target action or not) to interpret it as *Action Probability*. Roughly speaking, there are two approaches to prepare the binary classification result. The first approach is a kind of one-class classification techniques [10], [14], [15], where the learning process is executed without *negative* instances. The second approach is a kind of two-class classification techniques [16], [17]. The first approach discriminates from the likelihood of probability function that corresponds to the target action. The second approach classifies directly whether the input motion to be categorized or not. In this research, we adopt the two-class classification approach. Specifically we adopt a kernel-based

online action recognition method [4] based on support vector machines (SVM) [13]. As a wrapping method that enhances the binary output of SVM to a probability density function, we adopt Platt's method [18]. Thus $q_t^{(g,k)}$ can be defined as

$$\begin{aligned} q_t^{(g,k)} &= p(l_t^{(g)} = a_k^{(g)} | \mathcal{X}_t) \\ &= \frac{1}{1 + \exp(-\sigma \mathrm{sign}(l_t^{(g)} = a_k^{(g)}) f(\mathcal{X}_t))}, \end{aligned} \quad (3)$$

where function f depicts output of SVM for $\mathcal{X}_t$, $\mathrm{sign}(c)$ represents an indicator function and σ depicts a positive constant. Specifically, $\mathrm{sign}(c)$ returns +1 if c is true, else returns -1. This satisfies *Action Probability* defined in (1). At most time, f tends to output positive value if the target action occurs, else negative. Because $|f(\mathcal{X}_t)|$ corresponds to the reliability of the output of SVM, it is natural to formulate *Action Probability* as written in (3).

B. Modeling time-series Action Probability with HMM

1) Stable and Unstable intervals: We can find that there are two kinds of drastic changes of time-series of *Action Probability*. One is due to the temporal noise or error of *Action Probability*. The other comes from the segmental point of actions (see Fig. 4). This is normal response of *Action Probability*. Someone may think that it is simple and efficient to use "moving averaging" models in order to eliminate high frequency noise, however, the moving averaging models have trade-off problem. When we set the big window size in this method, the latency to detect segmental points occurs. Instead, when we set the small windows size in order to react quick response for the segmental points, the performance of the chunking process will be poor. Thus, we design our algorithm to meet the performance of not only eliminating the error of *Action Probability* but also detecting quickly the segmental points. What is worse, the segmentation using moving averaging models will fail when several *Action Probabilities* are both high and the difference of them is very small.

From Fig. 4, you may find it is impossible to determine whether noise or segmental points just observing the drastic change of time-series $q_t^{(g,k)}$. Our approach waits to determine the segmental points until drastic change of $q_t^{(g,k)}$ disappears. This seems to be critical in the latency of the segmentation, however, it is not so critical. It is because most intervals of the drastic change is very short. Hence, we must categorize two kinds of time-series model: the stable and unstable intervals of time-series $q_t^{(g,k)}$ and make classification rule. This rule classifies which categories fits better for the current time-series *Action Probability*. In this research, we use hidden Markov models [19] for time-series *Action Probability* modeling.

2) Designing HMMs for Stable Intervals: In order to represent the stable intervals, the corresponding time-series $q_t^{(g,k)}$ must satisfy the following conditions. One condition is that the largest $q_t^{(g,k)}$ is near to 1 while the others is around to 0 and the situation retains at a certain interval. Another condition is that the largest $q_t^{(g,k)}$ is near to 0 and this situation keeps at a certain interval. The former condition can be interpreted as "we can assign easily one action name as the label at current time", similarly, the latter as "we can assign ϕ as the label at present time". Because it is hard to prepare both conditions in a HMM, we make two HMMs, respectively. We call the HMM for the former situation Stable-P HMM, and the HMM for the latter situation Stable-N HMM.

In Stable-P HMM, we focus on the first and second largest $q_t^{(g,k)}$. This is because there is significant difference between the largest $q_t^{(g,k)}$ and the second largest $q_t^{(g,k)}$ in this situation. Hence, we set the observation variable for HMM $\boldsymbol{y}_t \in \mathbb{R}^2$ at frame t as $\boldsymbol{y}_t = (q_t^{(g,\lambda)}, q_t^{(g,\zeta)})^\mathsf{T}, \boldsymbol{y}_{t-1} = (q_{t-1}^{(g,\lambda)}, q_{t-1}^{(g,\zeta)})^\mathsf{T}, \ldots$, where λ means the ID of action name that satisfies the largest $q_t^{(g,k)}$ at frame t. Similarly, ζ depicts ID for the second. This means $\lambda = \arg\max_k q_t^{(g,k)}, \zeta = \arg\max_{k \neq \lambda} q_t^{(g,k)}$. Stable-P HMM must output $\boldsymbol{y}$ that satisfies $\{\boldsymbol{y}_t\}_2 \ll \{\boldsymbol{y}_t\}_1 \approx 1$. Thus, we design Stable-P HMM as a single state Markov model. The emitter function of Stable-P HMM is Gaussian distribution. On the other hand, the output of Stable-N HMM must satisfy $\{\boldsymbol{y}_t\}_1 \approx 0$ in a certain interval. Then, we also design Stable-N HMM as a single state Markov model. The emitter function of Stable-N HMM is also Gaussian distribution. The parameters of Stable-{P/N} HMMs are obtained from EM algorithm [20] using real time-series *Action Probability* data. We illustrate the imagery of the state transition model projected onto the observation space $\boldsymbol{y}$ (see Fig. 5).

3) Designing HMMs for Unstable Intervals: As a matter of course, we require multiple *Action Probabilities* in order to represent unstable intervals. Here we use the same variable $\boldsymbol{y}_t$ used in Stable-{P/N} HMMs. We design the five states hidden Markov Model as Unstable HMM. In each state of Unstable HMM, the emitter probability function is Gaussian distribution. In order to make clear distinction between Stable-{P/N} HMMs and Unstable HMM, we design the state diagram for Unstable HMM as Fig. 6. This state diagram implies the situation $\{\boldsymbol{y}_t\}_2 \ll \{\boldsymbol{y}\}_1 \approx 1$ and $\{\boldsymbol{y}_t\}_1 \approx 0$ never continues and is likely to output ambiguous values of *Action Probability*.

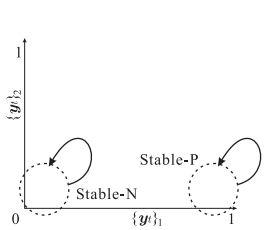

Fig. 5. State transition diagram for Stable-{P/N} HMM

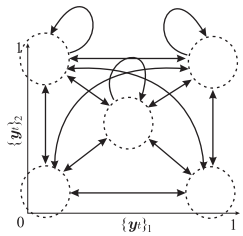

Fig. 6. State transition diagram for Unstable HMM

C. Online recognition and segmentation using stable and unstable HMM

In this section, we describe how to estimate time-series action names and detect action segments (tuple of action name, start point of occurrence, end point) with Stable-P, Stable-N HMM, and Unstable HMM. At first, the algorithm detects the changes λ, where $\lambda = \arg\max_k q_t^{(g,k)}$, as a cue of the segmental points ($t = t_0$). Then, the algorithm starts to classify the current time-series $q_{t_0:t}^{(g,k)}$ is in stable or unstable interval by calculating the likelihoods of the HMMs. Table I shows the description of our proposed algorithm.

TABLE I

ONLINE SEGMENTATION ALGORITHM IN g-TH GROUP OF ACTIONS

	Setting; SP: Stable-P HMM, **SN**: Stable-N HMM, **U**: Unstable HMM, fixed the maximal interval length for HMM: T_I
0	**Initialization;** $t = 0$, $t_0 = 0$, $l_t^{(g)} = \phi$
1	Calculating *Action Probabilities* of all the target actions in the group $q_t^{(g,k)}$, sorting them and selecting largest two $q_t^{(q,k)}$ as $\boldsymbol{y}_t$.
2	Assigning temporary labels at time t as follows. $$\text{if } \{\boldsymbol{y}_t\}_1 \leq 0.5, \quad l_t^{(g)} = \phi, \text{ else }, \ l_t^{(g)} = a_\lambda$$ where $\lambda = \arg\max_k q_t^{(g,k)}$.
3	Checking the temporary labels as follows. $$\begin{array}{ll} \text{if } l_{t_0}^{(g)} \neq l_t^{(g)} & \text{goto } \mathbf{4}. \\ \text{else} & t_0 \leftarrow t \text{ and goto } \mathbf{6}. \end{array}$$
4	Correcting the temporary labels by discriminating stability of time-series AP as follows. $$l_t^{(g)} \leftarrow l_{t_0}^{(g)} \ \text{ if } \begin{cases} l_t^{(g)} = \phi, \ \ln p_{\mathbf{SN}}(\boldsymbol{y}_{t_0:t}) < \ln p_{\mathbf{U}}(\boldsymbol{y}_{t_0:t}) \\ l_t^{(g)} \neq \phi, \ \ln p_{\mathbf{SP}}(\boldsymbol{y}_{t_0:t}) < \ln p_{\mathbf{U}}(\boldsymbol{y}_{t_0:t}) \end{cases}$$ else keeping $l_t^{(g)}$ and $t_0 \leftarrow t$.
5	Updating flag frame t_0. if $t - t_0 > T_I$, then $t_0 \leftarrow t$
6	Fixing $l_t^{(g)}$, $t \leftarrow t + 1$, and returning to **1**.

The advantage of the algorithm is that the length of interval $t_0 : t$ is not fixed but variable. This mechanism limits the latency for detecting the reliable segmental points as small as possible. Of course, we must give upper bound constant T_I for the interval.

IV. EXPERIMENTAL RESULTS

In this section, we illustrate the performance of the proposed recognition and segmentation method using time-series human motion data. We evaluated two aspects of the algorithm. One is for the performance of the segmentation in a group of actions. The other is for advantageous improvement to incorporate the semi-hierarchical representation of groups of actions (see Fig. 2). As the evaluation for recognition performance in a group of actions, we calculated the performance scores in three aspects, 1) the frame-wise accuracy for online recognition, 2) the accuracy for the timing of the detected start and end point of the action segments, 3) the accuracy of the number of the action segments. As for the accuracy of the number of the action segments, we compared the number of the action segments detected by the proposed method and labeled by human. As the evaluation for effects on the semi-hierarchical representation, we evaluated the revised and the error corrected frames by using constraints of it.

TABLE II

SPECIFICATION OF MOTION FOR TEST USED IN THE EXPERIMENT

Actor	4 Males in 20s
♯ of files	50
Total time	750 seconds, [avg. 15 seconds]
♯ of segments	338 (assigned by human)

Motion Dataset: In the following sentences, we describe motion data for training and evaluating used in the experiments. All the motion data contain human skeletal configuration and the time-series joints angles acquired by a magnetic motion capture system with 30 Hz. The skeletal configuration in the experiments has 36 degrees of freedom. Specifically, the format of the motion data is BVH. We categorize the motion data used in the experiments into two categories. One motion dataset serves as the training data for the *Action Probability* implemented in (3). In this experiment, we used ICS action database [21] for this dataset. ICS action database contains $3 \sim 4$ seconds pre-segmented motion clips about 25 actions. The motions are labeled with action names frame-wise. The other category serves as the testing data for our proposed recognition and segmentation method. The specification and the quantum of the motion data for testing our proposed method is summarized in Table II. The thumbnails of the example of the test motion in this experiment is shown in Fig. 7.

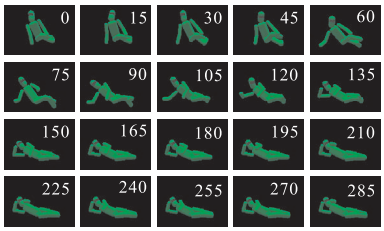

Fig. 7. Thumbnails of motion data example for testing the proposed method is shown. The figures, for example, 270, indicate frame number from the start.

Evaluation Method: As the performance scores for the frame-wise accuracy and timing accuracy of detecting the segmental point, we used F-measure as a basic performance score in the experiments. F-measure with adjustable positive parameter β is defined as

$$F_\beta = \frac{1}{\frac{\beta}{\beta+1}\frac{1}{R} + \frac{1}{\beta+1}\frac{1}{P}}, \ \beta > 0,$$

where R denotes recall and P denotes precision performance. Because F-measure can be interpreted as a harmonic mean of the recall and the precision, a higher F-measure indicates

the higher performance of the classifier. In this experiment, the adjustable parameter in F-measure, β, was set at 1.0.

For the frame-wise accuracy, R and P denotes as follows

$$P = \frac{\sum_{t,g} [\![\hat{l}_t^{(g)} = l_t^{(g)}]\!][\![\hat{l}_t^{(g)} \neq \phi]\!]}{\sum_{t,g} [\![\hat{l}_t^{(g)} \neq \phi]\!]},\ R = \frac{\sum_{t,g} [\![\hat{l}_t^{(g)} = l_t^{(g)}]\!][\![l_t^{(g)} \neq \phi]\!]}{\sum_{t,g} [\![l_t^{(g)} \neq \phi]\!]},$$

where $\hat{l}$ denotes estimated action labels, on the other hand, the action labels made by human is represented by l. A function $[\![var]\!]$ returns 1 if var is true, else returns 0. As for the accuracy of timing of the detection of segmental points, it is not proper to use the indicator function $[\![\cdot]\!]$. It's because this returns 0 in either case the difference between true and estimated segmental point is very small or too large. Thus we re-define the indicator function whose value is not zero but small when the difference is large. Specifically, we use Gaussian function as $\exp(-|\hat{t} - t|^2/15^2)$,where $\hat{t}$ represents estimated segmentation points, and t depicts ones made by human. So, for the accuracy of the timing of the segmental points, R and P denotes as follows

$$P = \frac{\sum_{i,l,g} \exp\Big(\frac{-|\hat{t}_{i,l}^{(g)} - t_{*,l}^{(g)}|^2}{15^2}\Big)}{\sum_{i,l,g} [\![\hat{t}_{i,l}^{(g)} \neq \phi]\!]},\ R = \frac{\sum_{i,l,g} \exp\Big(\frac{-|\hat{t}_{*,l}^{(g)} - t_{i,l}^{(g)}|^2}{15^2}\Big)}{\sum_{i,l,g} [\![t_{i,l}^{(g)} \neq \phi]\!]},$$

where $\hat{t}_{i,l}^{(g)}$ means i-th estimated segmentation points of action label l in g-th group, $t_{*,l}^{(g)}$ depicts ones which are made by human and are the closest to $\hat{t}_{i,l}^{(g)}$ in P. The same goes for $\hat{t}_{*,l}^{(g)}$ and $t_{i,l}^{(g)}$ in R. We calculated the ratio: (♯ of segments made by system) / (♯ of segments made by human) for accuracy of the number of the action segments. This implies that the ratio closer to 1.0 is better performance.

In order to clarify the performance of the proposed method, we compared the other online recognition method using time-series *Action Probability*. Classification for the first method is the same as (2). The second one uses moving averaging methods in order to eliminate the high frequency noise of *Action Probability*. This means that the moving averaging method uses feature vector $\boldsymbol{r}_t^{(g)}$ written as $\{\boldsymbol{r}_t^{(g)}\}_k = \sum_{\tau=0}^{T_R-1} q_{t-\tau}^{(g,k)}/T_R$. The classification rule for the moving averaging method can be written in (2) except replacing $q_t^{(g,k)}$ by $\boldsymbol{r}_t^{(g)}$.

Condition and Parameters: In the following, we describe the specific conditions and the parameters used in this experiment. In order to achieve our implementation of *Action Probability*, we used the same condition for SVMs reported in our previous report [4]. We optimized σ in (3) by MAP (maximum a posteriori) estimation. In this experiment, we use a Gamma distribution as the prior distribution of σ. Specifically we set the distribution as $p(\sigma) = \Gamma(\gamma, \gamma^{-1}) \propto \sigma^{\gamma-1}\exp(-\gamma\sigma)$, where $\gamma > 0$ (we set $\gamma = 0.1$ in the experiments). In the experiment we used a part of motions in ICS action database in order to learn SVM and σ. We used a kind of cross validation techniques (5-fold) for the optimization procedure of σ.

In order to make Stable-P, Stable-N, and Unstable HMMs, we prepare the segments that contain time-series *Action Probability*. The preparation is done manually by watching and cutting for the corresponding HMMs. All the parameters of the HMMs are optimized with the EM algorithm from this segmented time-series *Action Probability*. We calculated the performance in several conditions by changing the interval window size $T_I = 5, 10$ and $T_R = 5, 10, 25$.

Result: The performance for each condition of each algorithm is shown in Table III. In Table III, AP+threshold represents the compared method without smoothing, and MA denotes the compared moving averaging method. The figures in parenthesis denotes upper bound of length of intervals: T_I for HMM and T_R for the moving averaging method, respectively. In each method, the frame-wise accuracy achieves similar score, however, our method is superior in removing the unnecessary action segments than the other methods. As we noted above, the moving averaging method gets worse in the timing accuracy of detecting segmental points, when we set $T_R = 25$ in order to remove unnecessary action segments as much as our proposed method can do. This result implies our recognition method can not only eliminate the error of *Action Probability* but also detect the segmental point quickly. The recognition and segmentation result of the proposed method with time-series *Action Probability* for the input motion in Fig. 7 is shown in Fig. 8.

TABLE III
RECOGNITION PERFORMANCE IN EACH METHOD

	frame-wise	timing	ratio of ♯ of segments
Our Method(5)	0.81	**0.55**	1.10
Our Method(10)	0.81	**0.55**	**0.97**
AP+threshold	**0.82**	0.52	1.60
MA(5)	**0.82**	0.53	1.42
MA(10)	0.81	0.53	1.32
MA(25)	0.80	0.51	0.96

Advantageous Effect using Semi-Hierarchical Relation: As a result of the effect using the semi-hierarchical representation of actions, the number of the revised frames and the error corrected frames is illustrated in Table IV. This result shows that our approach is very simple but has certain advantage when we compare the recognition method without hierarchical relations.

TABLE IV
ERROR CORRECTION USING STRUCTURES OF GROUPS OF ACTIONS

	♯ of revised frame	♯ of corrected frame
HMM(5)	971	643
HMM(10)	859	531

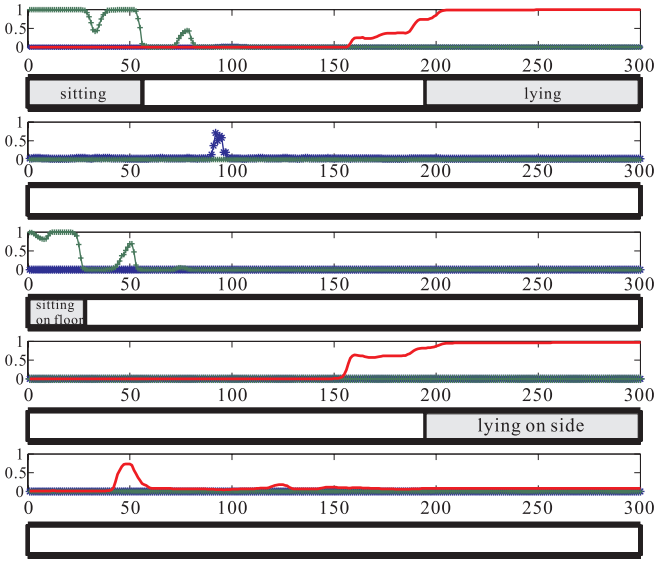

Fig. 8. Recognition and segmentation result of our method is shown. The 1st row of the figure represent root group's time-series *Action Probability*. The 2nd shows the estimated labels for the root group. The 3rd and the 4th represent *"standing group"*'s result, the 5th and 6th represent *"sitting group"*'s result, the 7th and 8th represent *"lying group"*'s result, and the 9 th and 10th represent *"looking group"*'s result, respectively.

V. Conclusion

In this paper, we propose a robust online action recognition algorithm using segmentation scheme that detects the start and end points of action occurrence. Our algorithm utilizes a semi-hierarchical representation for categorizing actions in order to simplify the algorithm. As a part of the algorithm, we use time-series of *Action Probability*, the likelihood of action occurrence. *Action Probability* is calculated from the result of SVM-based online action recognizer [4]. The system can classify robustly and immediately whether the current time is to be segmented or not by stability analysis for *Action Probability* with hidden Markov models. The experimental result using real motion capture data shows that our algorithm does not only prevent the system from making the unnecessary segments due to the error of time-series *Action Probability* but also decreases the latency for detecting the segmental point better than the moving average method. In addition, our algorithm can be applied to dynamic actions such as walking with the action recognition method considering dynamics of motion [22].

Our suggestion for future work is as follows. Our algorithm works well, however, there are multiple parameters and conditions given a priori. Thus we will explore how to reduce these parameters. Also, the error correction algorithm using the semi-hierarchical representation of groups is good but naive. Hence, we have a plan to make alternate for that.

References

[1] J. Davis and A. Bobick. The representation and recognition of human movement using temporal templates. In *Proceedings of the 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 928–934, 1997.

[2] J. Yamato, J. Ohya, and K. Ishii. Recognizing Human Action in Time-Sequential Images using Hidden Markov Model. In *Proceedings of the 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 379–385, 1992.

[3] T. Mori, K. Tsujioka, M. Shimosaka, and T. Sato. Human-like action recognition system using features extracted by human. In *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1214–1220, 2002.

[4] T. Mori, M. Shimosaka, T. Harada, and T. Sato. Recognition of actions in daily life and its performance adjustment based on support vector learning. *International Journal of Humanoid Robotics*, 1(4):565–583, 2004.

[5] T. Inamura, Y. Nakamura, H. Ezaki, and I. Toshima. Imitation and primitives symbol acquisition of humanoids by integrated mimesis loop. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pages 4208–4213, 2001.

[6] C. Bregler. Learning and recognizing human dynamics in video sequences. In *Proceedings of the 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 568–574, 1997.

[7] V. Pavlović, J. Rehg, and J. MacCormick. Learning swtiching linear models of human motion. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 981–987, 2001.

[8] J. Kohlmorgen and S. Lemm. A dynamic HMM for on-line segmentation of sequential data. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, 2002.

[9] X. Ge. *Segmental semi-Markov models and applications to sequence analysis*. PhD thesis, University of California Irvine, 2002.

[10] T. Mori, Y. Segawa, M. Shimosaka, and T. Sato. Hierarchical recognition of daily human actions based on continuous hidden Markov models. In *Proceedings of the Sixth IEEE International Conference on Automatic Face and Gesture Recognition*, pages 779–784, 2004.

[11] R. Duda, P. Hart, and D. Stork. *Pattern classification (2nd ed.)*. Wiley Interscience, 2000.

[12] C. Williams and D. Barber. Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1342–1351, 1998.

[13] B. Schölkopf and A. Smola. *Learning with kernels*. MIT Press, 2002.

[14] D. Tax, A. Ypma, and R. Duin. Outlier detection using classifier instability. In *Proceedings of the Seventh International Workshop on Structural and Syntactic Pattern Recognition*, 1998.

[15] B. Schölkopf, R. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt. Support vector method for novelty detection. In S. Solla, T. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 582–588, 2000.

[16] S. Romdhani, P. Torr, B. Schölkopf, and A. Blake. Efficient face detection by a cascaded support-vector machine expansion. *Proceedings of The Royal Society A*, 460(2501):3283–3297, 2004.

[17] M. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

[18] J. Platt. Probabilities for SV machines. In A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74, 1999.

[19] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, volume 77, pages 257–285, 1989.

[20] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B(Methodological)*, 39(1):1–38, 1977.

[21] T. Mori, M. Shimosaka, and K. Tsujioka. ICS action database. http://www.ics.t.u-tokyo.ac.jp/action/.

[22] M. Shimosaka, T. Mori, T. Harada, and T. Sato. Marginalized bags of vectors kernels on switching linear dynamics for online action recognition. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 3083–3088, 2005.