Recognition of Human Daily Life Action and Its Performance Adjustment based on Support Vector Learning

Masamichi Shimosaka[1], Taketoshi Mori[1],
Tatsuya Harada[1], and Tomomasa Sato[1]

The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
{simosaka, tmori, harada, tomo}@ics.t.u-tokyo.ac.jp,
WWW home page: http://www.ics.t.u-tokyo.ac.jp/

Abstract. This paper presents a recognition method of human daily life action. The system deals with actions related to regular human activity such as walking or lying down. The main features of the proposed method are: 1)simultaneous recognition, 2)expressing unclarity in humans' recognition, 3) defining similarity between two motions by utilizing kernel functions, which is derived from expressions of action based on human knowledge, 4)robust learning capability based on Support Vector Machine. The comparison with neural networks optimized by back propagation algorithm and decision trees generated by C4.5 proves that the accuracy of recognition in the proposed method is superior to the others. Recognition of daily life action is expected to ensure smooth communication between humans and robots and to enhance support functionality in intelligent systems.

1 Introduction

Robots including humanoids are expected to support humans in everyday tasks and activities in the future. To achieve assistance in a wide range of areas, it will be important for robots to be able to communicate effectively with humans. Recognition and understanding of human actions has potential to contribute to this ability and many other applications, such as human computer interaction and search engine for multi media databases.

In almost all the researches, the target actions are sign gestures. For example, Starner et al. made a system which recognizes the American sign language[1] and Wilson et al.[2] implemented a system which recognizes users' gesture. However, it is considered that regular actions, such as Walking and Sitting, will be important as the target actions to recognize in support systems for everyday life. The authors have designed and implemented a recognition system for such regular actions[3]. The main features of this system have been 1) simultaneous recognition, 2) expressing unclarity in humans' recognition, 3)using description of action, which is based on human knowledge. The detailed information of these features are described in section 2.

Because this system has lacked learning ability, in other words, the performance of the system must have been tuned by hand, there have been problems of the system in the point of extensibility and versatility. To solve this problem, the main contribution of this paper is to design and implement the recognition algorithm whose performance can be tuned by a learning process from sample motion data.

2 Recognition of Human Daily Life Action

2.1 Primitive Features in Our Recognition Method

The proposed recognition system in this paper takes over the features of our former recognition system[3] described below.

Simultaneous Recognition It is preferable that a recognition system for human daily life actions can output multiple action names as recognition result. For example, we can readily recognize the two actions involved when observing someone waving his or her hand while walking.

In order that output of a recognition system become closer to human recognition, we should design the system to have this ability. Concretely speaking, a recognition system should be designed to be able to output multiple action names at the same time. We designed our system to have the ability of outputting multiple action names simultaneously.

Unclarity in Recognition Humans can not always give absolute decision whether some action really occurs or not when watching someone acting. For instance, decision whether lying or not made by human may contain unclarity when observing someone getting up. In order that the output of a recognition system will become closer to human recognition, the system should be designed to have ability to output fuzzy recognition results. We designed that our system can output non only decisive result but also unclarity result.

Utilizing Expressions of Action by Human Feature selection and extraction is considered as one of the most important elements in action recognition, because the specific features of each action such as the motion and the pose of the body region have wide variation. For example, the forward motion of hip could be one of the features of walking, meanwhile the direction of head is considered as irrelevant feature of sitting.

Human can intuitively extract specific features of each action without trouble. In other words, human can easily express an action by representing the motion or the pose of body parts. Thus we attend to these expressions. Consequently, our system only uses relevant features of the target action, which are derived from expressions.

2.2 Criterion of Performance

Performance of an action recognition system should be evaluated as well as many other pattern recognition applications such as OCR, face recognition and speech recognition. The main criterion of the evaluation is the accuracy of recognition results. The correctness of recognition result is basically based on humans' judgment.

While humans can intuitively recognize human motion with adequate several action names, there are several way of evaluating the accuracy of recognition result. Because of the variety of attended actions and time segmentation problem, it is hard to evaluate the accuracy of recognition result properly.

Thus, in order to make us evaluate the performance of a recognition system easily, the correctness of the recognition result by humans is defined as follows. On observing someone acting, even if the person is performing several actions simultaneously, reference data of recognition result is generated in sync with the input motion as the discrimination of one action by paying attention to this action only. In other words, the correctness of recognition results is generated by humans from moment to moment with someone's motion as decision result whether one specific action occurs or not.

The recognition algorithm in the former recognition system we have implemented[3] fundamentally uses fuzzy logic, which is also extended to time series problems. However this method has advantage that the system can be implemented intuitively, the ability of learning process is essential factor for action recognition system in the following two reasons. The former reason is difficulty of tuning the performance by hand. The latter is that there is variety of action names we have to target.

3 Recognition System Implementation

3.1 Input and Output

While many other recognition systems[4] use the sequential series of images as the input of the systems, our system uses a motion captured file which contains humans' posture, which is measured by some optical or mechanical motion capturing systems. This enables the system to fetch information of motion easily and robustly, and we can concentrate on designing the recognition process itself.

The output of our system contains some action names in sync with a frame of the input motion, so that the problem of the segmentation of the time direction is not treated in this paper.

The format of the file is BVH[5], a de-facto standard motion file format by Biovision Corporation. BVH files contain the structure of a human as a linked joint model(figure) and the motion of the figure per frame.

The figure used in the proposed system is shown in Fig.1. The hip is defined as the root joint of the figure and has 6 degrees of freedom (DOF). The torso, neck, legs and arms each have 3 DOF. Therefore, the figure has total of 36 DOF.

BVH files can be generated from the data captured by a magnetic motion-capturing system (Motion Star, Ascension Technology) in which the subject wears magnetic sensors fitted to the corresponding joints. The body motion is measured per 33 mili seconds.

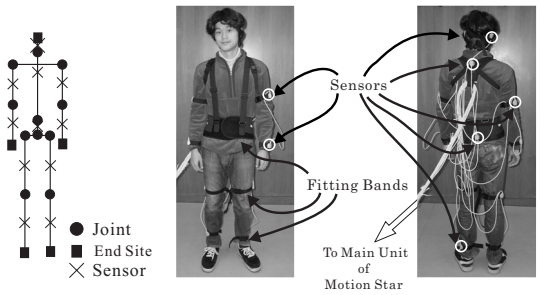

Fig. 1. Human as Link Joint Model and an actor with magnetic sensors

3.2 Recognition System Configuration

Fig.2 shows the processing flow and the structure of our system. In order to realize the simultaneous recognition, the system contains multiple recognition processes, each of which is assigned to the recognition of one action. This primitive recognition process for one action is called as "Action Element Recognizer(AER)".

The process of each AER runs in parallel with the other AERs. For example, AERs which recognize such actions as walking , sitting, and other action run in parallel.

The system collects the results of all processes, and outputs the results of each recognition process per frame. New action can be recognized simply by adding that recognition process to the system.

As noted above, an AER recognizes one assigned action. In practice, an AER which recognizes walking action discriminates whether someone walking or not walking. Thus the result of one AER process is independent of the others and all over performance of the system relies on the performance of each AERs.

3.3 AER Configuration

Time series of human motion is utilized as input of each AER. An AER outputs the recognition result whether the assigned action occurs or not per frame in sync with the input motion. An AER of our former system outputs a matching value whose range is 0 to 1 by utilizing fuzzy logic, in order that the output of the AER makes the system have the ability of the unclarity as same as humans. But this causes difficulties on preparing and inputting recognition results by human.

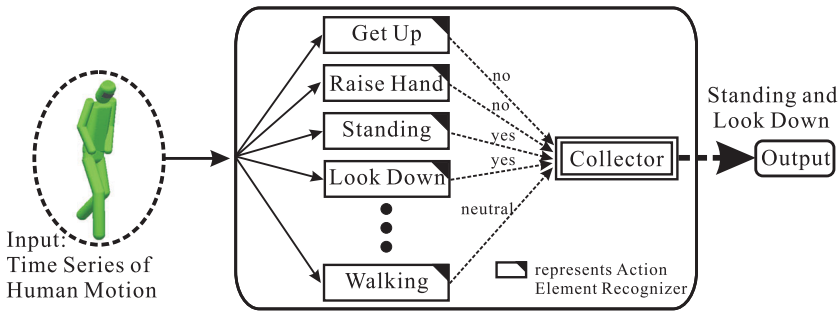

Fig. 2. Configuration of recognition system

Hence, in the newly proposed AER, the output of AER consists of multiple classes which represent not only decisive but also inexplicit result, so that human can easily make the databases which contain humans recognition result. Concretely speaking, the number of the category is three. One is the category named as "yes" which represents that the assigned action clearly occurs. Another is named as "no" which represents that the opposite meaning of "yes". The last one represents the unclarity of recognition result called as "neutral". Thus, the proposed AER is a multiple class (three classes) classifier.

In general, the framework of the multiple class classifier is summarized as following three types. One of the methods consists of "one vs. one" classifier. This requires three binary classifiers. Another one is built by "one vs. the other" classifier. This method also requires three binary classifiers. The last one is categorized into K nearest neighbors method.

In spite of the above technique, the proposed AER requires two binary classifiers and outputs the integrated result of two binary values. Concretely speaking, the one binary classifier judges whether "yes" or not-"yes", the other judges whether "no" or not "no". The reason that we have adopted this composition is that "neutral" category rarely happens in some actions. In other words, human can explicitly discriminate motion as sitting down when watching someone standing then sitting.

Binary Classifier in AER In brief, each of two binary classifiers in an AER recognizes input human movement by utilizing some templates which represent time series of human motions acquired in advance. In other words, the binary classifier utilizes template matching technique. Concretely speaking, the output is calculated based on the similarities between the input motion and the templates with humans reference.

The system utilizes the expressions of action described by human, in order that the AER fully takes advantage of the excellent power of feature selection and extraction by human. More specifically, the system selects the input motion and transforms it into adequate features for AER.

Meanwhile, the expressions of action by human is divided into three categories in terms of conversion way from the input motion to the features. The three categories are listed as follows.

- **Status**
  For example, the phrase that the position of the head is high is categorized into this type. In this type of expression, the input motion is normalized by some reference value, such as body height. Finally, this normalized value is converted to the input feature whose range is from 0 to 1 by some scaling factors. This process makes the system easily deal with variant types of the input motion, such as angle, velocity and height.
- **Transition**
  This represents the transition of the status of some motion with time. For example, the phrase that the height of the head goes down is categorized into this. In this type, after the input motion is converted to the normalized value in the same manner as "Status ", the converted value for a certain period is selected as the input feature. For example, 13 samples of height of his/her head, which are sampled from a certain time during 2 seconds at 6[Hz], are selected as the input feature.
- **Iteration**
  This represents the repetition of some event occurs in input motion. For example, the phrase that each foot contacting on ground is categorized into this. This process converts input motion to the input feature by the frequency domain in order to strengthen the iteration of motion.

In general, a recognition system changes the recognition algorithm by considering the difference of the treatment of the input motion in terms of time series. The several categories of expressions are generated by the difference of the treatment of the input motion in terms of time series. Thus, the problem that the way of recognition differs by the type of the expressions arises. Concretely speaking, the different types of the expressions cause the corresponding calculation method of template matching between the input motion and the templates. At worst case, the way of the learning process becomes different by the types of AER.

In order to reduce such difference caused by several types of expressions and unify the method of recognition and learning in all the AERs, the binary classifier in the proposed AER utilizes the kernel technique[6]. The kernel serves as the function which mathematically calculates similarity between one sample and another sample and is drawn attention in the field of statistical machine learning community. As a result, the binary classifier in the proposed AER is kernel classifier and the kernel technique enables the binary classifier to deal with the three types of expressions uniformly.

Binary Classifier as Kernel Classifier From here, a kernel classifier is introduced as the binary classifier in the AER. We denote by $\boldsymbol{x}$ the time series of input motion. $D = \{\boldsymbol{x}_i, y_i\}_{i=1}^{l}$ are the input-output pairs in total l frames, where $\boldsymbol{x}_i$ represents i th frame sample motion and its corresponding reference

binary(e.g. "yes" or not-"yes") signal by y_i. We can write by α_i the co-efficiencies whose value is proportional to importance of the templates. Similarity value calculated by Kernel K between the input motion and one templates is represented by $K(\boldsymbol{x}_i, \boldsymbol{x})$. By utilizing these notations, the mapping between the input and the output of the binary classifier in the AER f can be written as

$$f(\boldsymbol{x}) = \mathrm{sgn}\left(\sum_{i=1}^{l} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b\right). \tag{1}$$

The parameter b depicts the bias value and the function $\mathrm{sgn}(\cdot)$ is a step function where the relation of input-output is as follows.

$$\mathrm{sgn}(t) = \begin{cases} +1 \text{ if } t > 0 \\ -1 \text{ if otherwise} \end{cases} \tag{2}$$

Deriving Kernels: Combination of Kernel Values Per Expression The proposed method derives the kernel value in the classifier as the products of all the kernel values corresponding to the similarity in each expression. Concretely speaking, the kernel value which corresponds to the similarities in j th expression $K_j(\cdot, \cdot)$ can be written as

$$K_j(\varphi_j(\boldsymbol{x}_i^{(j)}), \varphi_j(\boldsymbol{x}^{(j)})), \tag{3}$$

where $\boldsymbol{x}^{(j)}$ denotes the selected input motion in the j th expression, and $\varphi_j(\cdot)$ represents the converter from the selected input motion to the input feature.

When the numbers of the expressions in the target action is d, the kernel value in the target action $K(\cdot, \cdot)$ can be written as

$$K(\boldsymbol{x}_i, \boldsymbol{x}) = \prod_{j=1}^{d} K_j(\varphi_j(\boldsymbol{x}_i^{(j)}), \varphi_j(\boldsymbol{x}^{(j)})). \tag{4}$$

Example: Deriving Kernel in Walking The derivation process of the kernel value in walking is as follows. The relations between expressions, selected motions from the expressions and the type of expressions are listed in the Table1. The image of the derivation process is depicted in the Fig.3.

Table 1. Example: Information for Recognizing Walking

Expressions	**Selected Motions**	**Type of Expressions**
Position of Head is High	The height of the head	Status
Moving forward	Forward velocity	Status
Each foot contacting ground	Relative height between right and left legs	Iteration

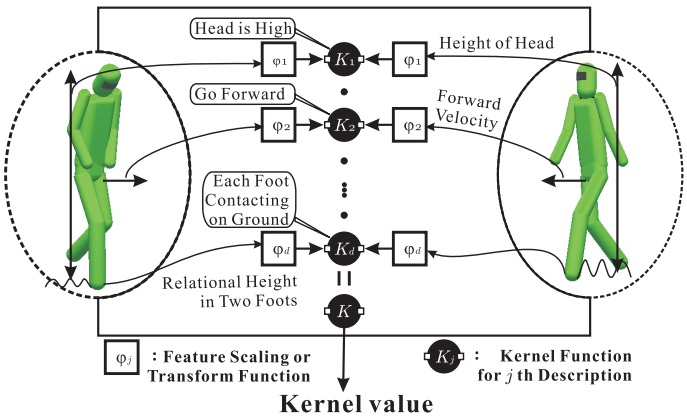

Fig. 3. Example: Derivation process of the kernel value in the binary classifier of the walking AER.

4 Learning Process of AER

4.1 General Outline of Learning Process

The learning process in the AER tunes the co-efficiencies(α_i, b) of the binary classifiers from the training data, which contain the input-output pairs. What we have to pay attention to in learning process are listed as follows.

- Generalization ability
- Incremental(Online) learning ability
- Model selection in terms of kernel
- Computational effort of learning and recognition process

Even though there are several types of learning process for kernel classifier proposed in the statistical machine learning community, such as Gaussian Process[7], Relevance Vector Machine[8] and Support Vector Machine[9], the Support Vector Machine(SVM) is utilized as the basic learning scheme of our proposed method. This is because no kernel classifiers except SVM can optimize its performance incrementally by utilizing the same criterion as the batch process. In the following sections, the detailed explanations of the learning process are described. Firstly, the batch SVM learning process is introduced as the basic scheme of our learning process. Secondly the online learning algorithm based on SVM is described. Finally, the kernel parameters optimization technique which serves as the part of the model selection problems is explained.

4.2 Learning Scheme based on SVM

Batch Learning Algorithm If some adequate motion samples with humans' judgment which contain "yes" , "neutral" and "no" can be acquired, the batch

learning process can be executed to optimize the weighting and the bias parameters. Concretely speaking, this algorithm optimizes the weighting parameters α_i and the bias parameter b from the motion samples with humans judgment $D = \{\boldsymbol{x}_i, y_i\}_{i=1}^{l}$ and the kernels given prior. This utilizes the criterion so called "margin" which represents the degree of separation between one category and the other. The detailed derivation of this algorithm is described by Vapnik[10].

The optimization algorithm in this learning process is derived from the quadratic programming problem of weighting parameters $\alpha_i,\ i = 1, \ldots, l$, and can be written as

$$\begin{aligned} &\text{minimize}: \quad W(\boldsymbol{\alpha}) \equiv \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \sum_i \alpha_i + b\sum_i \alpha_i y_i \\ &\text{subject to}: \begin{cases} \sum_{i=1}^{l} \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C,\ \forall i = 1, \ldots, l \quad , \end{cases} \end{aligned} \tag{5}$$

where the constant positive number C penalizes the training error in the non-separable case.

This QP problem makes the solution have a number of weighting parameters equal to zero. Since there is a weighting parameter α_i associated to each motion templates $\boldsymbol{x}_i$, only the templates corresponding to non-zero α_i (the "support vectors") will influence the output of the classifier.

Online Learning Algorithm The online learning algorithm we use can be applied, only if binary kernel classifiers optimized in advance are given and the additional motion with human judgment is input. This algorithm originally was proposed by Causwenberghs et al.[11]. It basically uses the Karush-Kuhn-Tucker(KKT) condition in the batch learning of SVM QP problem in order to avoid the over-fitting problems which often occur in many other online learning algorithms. The optimal condition used in this algorithm is denoted as follows, which is derived by KKT optimal condition.

We define by $g(\cdot)$ the derived function of the performance function W in the Eq. (5)

$$g(\boldsymbol{x}_i) \equiv \frac{\partial W}{\partial \alpha_i} \tag{6}$$

$$= y_i f(\boldsymbol{x}_i) - 1 \tag{7}$$

Then, the relation between the weighting parameter α_i and the function $g(\boldsymbol{x}_i)$ should be written as

$$g(\boldsymbol{x}_i) = \begin{cases} > 0 \text{ if } \alpha_i = 0 \\ = 0 \text{ if } 0 < \alpha_i < C \\ < 0 \text{ if } \alpha_i = C \end{cases} \tag{8}$$

If we consider that the optimized binary classifier trained with Data $D = \{\boldsymbol{x}_i, y_i\}_{i=1}^{l}$ is given, we denote by α_i^* the optimal weighting parameters. Nextly,

the new labelled sample denoted by $\boldsymbol{x}_c$ is added to the classifier, then the value of derived function $g(\boldsymbol{x}_c)$ is calculated as the first step of the online learning. If the value of $g(\boldsymbol{x}_c)$ is less than zero, the corresponding weighting parameter should be more than zero, which is derived from Eq. 8. In other words, the additional motion should be one of the support vectors.

In the case that the additional motion has potential to be a support vector, the iterative process is done as follows. The iterative process increases the value of the co-efficiencies α_c in incremental steps to keep the Eq.8 in the other training data, until additional training data satisfies the Eq.8. The change of the other coefiiciencies by α_c growth is derived by equilibrium status based on KKT condition. The detailed information of deriving the equilibrium relations is described by Causwenberghs et al.[11]

Kernel Parameters Optimization It is the fundamental premise that the kernel types and their parameters are priori given in the learning process of any kernel classifiers, and the performance is surely dependent on the kernel types and their parameters. However the SVM achieves more honor than other classical learning algorithms in terms of the performance of many applications, the performance fails to acquire the high accuracy when some kernel types and its parameters are set. In this paper, the kernel parameters optimization scheme is adopted as the part of the solution for these kind of model selection problems.

Thus, a generalization error of a SVM can be written as the function of the parameter $\boldsymbol{\theta} \in \boldsymbol{R}^{d_\theta}$, the kernel parameters, by $T(\boldsymbol{\theta})$. Finally, we denote by $\boldsymbol{\theta}^*$ the optimized kernel parameters as

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} T(\boldsymbol{\theta}) \tag{9}$$

In order to realize the kernel optimization scheme easily and robustly, the following list is the summary of the consideration.

- Utilizing an effective searching algorithm in the space of kernel parameters.
- Using an accurate estimator of the performance of SVM computed by little amount of calculation

In this paper, the gradient descent algorithm is utilized as the effective search engine. Thus, the general outline of the kernel optimization algorithm is listed as

1. Initialize $\boldsymbol{\theta}$ with some values.
2. Using a batch SVM algorithm, find the optimal co-efficiencies.
3. Update the parameter $\boldsymbol{\theta}$ such that $T(\boldsymbol{\theta})$ is minimized.
   This can be written as

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \epsilon \frac{\partial T(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}. \tag{10}$$

4. Go to step 2 or stop when the minimum of T seems to be at balance. We can write the terminal condition of this step as

$$\left|\frac{\partial T(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right| < \epsilon, \tag{11}$$

where ϵ is some positive constant number.

There are several good estimators for the performance of SVM. In this research, we adopt the "span bound" technique proposed by Chapelle et al.[12]. This is because the gradient descent technique requires the derived function for kernel parameters and co-efficiencies, and we can explicitly write it if estimator is based on span bound. The property of span bound technique has close relationship with the *Leave-One-Out* validation error estimation, whose computational cost is too high but accuracy of estimation gets the excellent quality.

The estimator based on the span bound is defined as

$$T = \frac{1}{l}\sum_{p=1}^{l} \Psi(\alpha_p S_p^2 - 1), \tag{12}$$

where Ψ is sigmoid function which is denoted as

$$\Psi(t) \equiv \frac{1}{1+\exp(-At+B)},\ A > 0,\ B \geq 0. \tag{13}$$

The parameters A and B is derived from the Platt's process[13]. The variable S_p represents the span bound of training data corresponding to p th support vector and is defined as

$$\Lambda_p = \left\{ \sum_{i\neq p, \alpha_p > 0} \lambda_i \phi(\boldsymbol{x}_i), \sum_{i \neq p} \lambda_i = 1 \right\}, \tag{14}$$

$$S_p^2 = d^2(\boldsymbol{x}_{\boldsymbol{p}}, \Lambda_p) = \min_{\boldsymbol{x}\in\Lambda_p} (\boldsymbol{x}_p - \boldsymbol{x})^2, \tag{15}$$

where λ_p represents linear combination of all the support vectors except p th support vector training data. Thus, we explain S_p holds the distance between Λ_p and $\boldsymbol{x}_p$. In practice, the computation of S_p is easy enough as

$$S_p^2 = \frac{1}{(\tilde{K}_{sv}^{-1})_{pp}}, \tag{16}$$

where $\tilde{K}_{sv}$ denotes

$$\tilde{K}_{sv} = \begin{pmatrix} K_{sv} & \mathbf{1} \\ \mathbf{1}^t & 0 \end{pmatrix}. \tag{17}$$

We define by K_{sv} the matrix containing the kernel values corresponding all the support vectors.

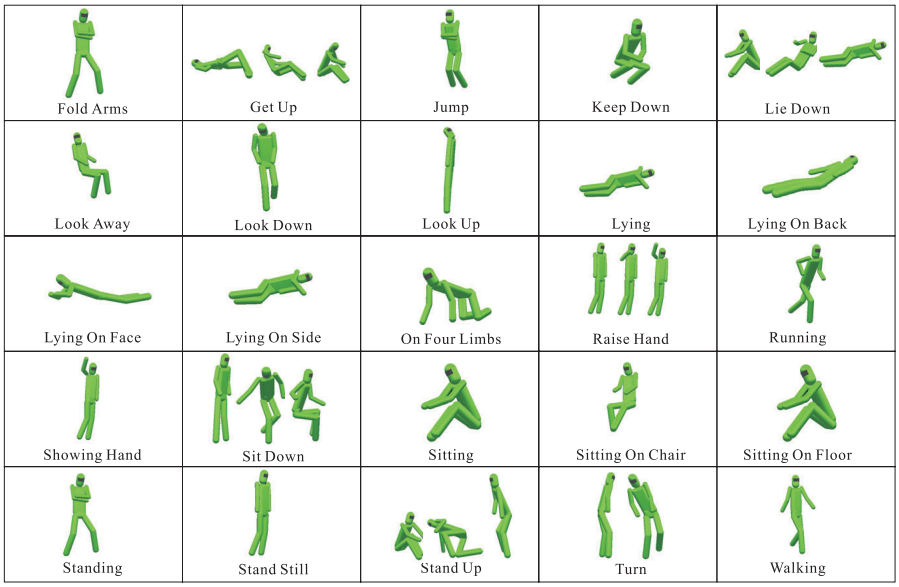

Fig. 4. The 25 daily life actions are listed as the target actions in the experiments

5 Performance Evaluation Experiments

We present some experiments for evaluating the performance of the proposed recognition system optimized by the learning process noted in previous section. The evaluation criterion of all the experiments we execute is based on the average frame-wise accuracy of AERs.

The experiments we conduct can be divided into three categories, which are

- the evaluation of the performance acquired by the batch SVM learning process and comparison with the performance acquired by other classical batch learning algorithms,
- the evaluation of the properties of the SVM based online learning algorithm,
- the evaluation of the properties of the kernel parameter optimization algorithm.

Target Actions and Motion Data For the experiments, we selected 25 actions as the target actions in the experiments which occur indoor and can be measured by the motion capturing system, such as lying and getting up. Fig.4 depicts all names and snapshots of the target actions. In each experiment, we have measured motion data by a magnetic motion capturing system. All the motion data we have used in the experiments are incorporated in ICS Action Database[14]. Each motion data in this database contains a BVH formatted motion captured file and its reference files per each target action. One reference file

contains humans' judgment for the assigned action per frame by three degrees ("yes", "neutral" and "no") .

Concretely, the motion data used in the experiments consists of 5 collections of BVH files. One set contains 25 BVH files and its reference files. In other word, 125 BVH files and 3,125 reference files are used in our experiments. The average span of one BVH file is almost 3 [Sec.] and the total length of all BVH files are 12,126 frames. The actor in all the BVH files is the same male person. Fig.5 depicts thumbnails of one BVH file when the actor stands up. The thumbnails are selected per 10 frames, i.e. 0.33[sec.].

Fig. 5. A BVH file from ICS Action Database where a male stands up from sitting. The thumbnails are selected per 0.33[sec.] from the file. The figures shown in the right-top side of each thumbnail indicate seconds from the starting time.

5.1 Evaluation of Performance by Batch Learning

We evaluated the performance of the system optimized by the batch SVM learning algorithm. The evaluation is done by calculating the average accuracy per frame of all AERs when the test motion data are input to the system. Concretely speaking, we utilize one collection of BVH files as the training data of the system, and four collections as the evaluation data. We define the average accuracy ratio, the proportion of the frames agreeable with the reference and the output.

Here, the prior parameters of SVM utilized in this experiment are described. We utilize Radial Basis Function in all the kernels in all the AERs. We denoted by $\sigma > 0$ the kernel parameter, which is the same value in all kernels. If we define by f_d the total number of the features dimension, which are the products after the selection and conversion to input kernel functions. Then the kernel parameter σ is set to $1.5\sqrt{f_d}$ and the penalizing constant number C is $100\sqrt{f_d}$.

Besides the evaluation of the performance of the proposed system, the comparison with multi layered neural networks optimized by back propagation algorithm (BP) and decision trees automatically generated by C4.5[15] is conducted. All the features utilized in these classifiers are same as the features which enters directly to the kernels of the proposed system.

The neural networks configuration in the experiments contains three layers of neurons. The type of the neurons in the input and the output layer is basically

linear function, while the type in the hidden layer is sigmoid function. The number of the neurons in the hidden layer is experientially decided as $2f_d$ to perform better. As the implementation of C4.5 algorithm, we use the software by Quinlan[16].

The average performance per AER optimized by the batch SVM, BP and C4.5 are shown in Fig.6. By the batch SVM, the mean accuracy of all the target actions per frame reaches 97.1%, meanwhile the accuracy reaches only 87.4% in the neural networks by BP and 94.1% in the C4.5 decision trees.

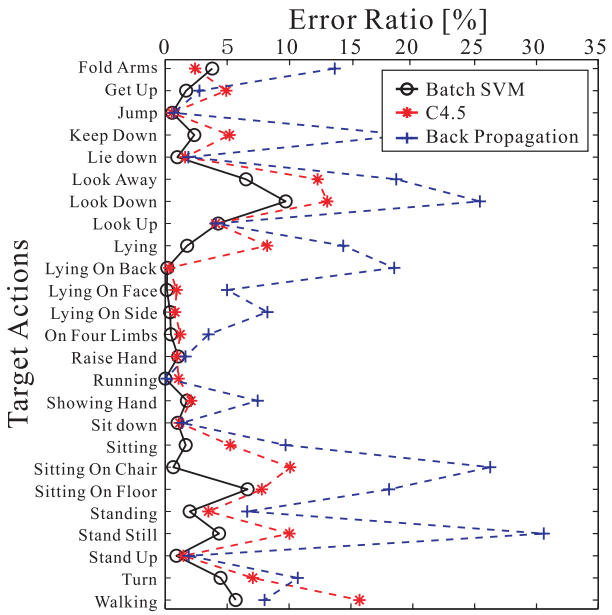

Fig. 6. This shows the error ratio per AER as performance optimized each recognition method.

5.2 Evaluating Properties of Online Learning

In this experiment, we evaluated the property of the online learning algorithm mentioned above. By comparing the performance and some parameters of recognition system acquired by the batch SVM with those by the online SVM, the evaluation is conducted. Concretely speaking, we compare the performance, the number of the support vectors and the sum of all the co-efficiencies of the classifier, which can be written as $\sum_{i=1}^{l} \alpha_i$. The reason why we evaluate the sum of the weighting parameters is that it has the relations with the margin between two classes. The two types AERs to be compared are described as follows. The former is acquired only by the batch learning algorithm from two sets of BVH

files. The latter is acquired by the online SVM with one training data set, after the batch learning process with one collection of training data.

The experimental result shows that the average difference of the performance between two classifiers is 0.1%. The ratio number of support vectors by the batch SVM to the online SVM is 1.13. Thus, we can consider this online learning algorithm has the similar property as the batch learning algorithm.

In fact, the number of the target actions we can actually evaluate in this experiment was 19. This is because the online learning process is broken by the numerical unsteadiness of the iterative steps for the other actions(i.e. the number of them is 6) such as Turn and Look Away. Concretely speaking, the unsteadiness occurs at the inversion of the matrix of which each component represent the product of the kernel value and the two labels. We will implement a recursive updating of the inversion of the matrix which can avoid this unsteadiness, instead of calculating the inversion directly.

5.3 Evaluating Property of Kernel Parameters Optimization

Experiment to evaluate the property of the kernel parameters optimization we implement was conducted. Concretely speaking, the performance and the changes of the number of the support vectors are the criteria of the evaluation. We selected 8 actions such as Folding Arms and Lying in this experiment to satisfy all the conditions listed as follows.

- Actions whose accuracy cannot reach 99%.
- Actions where the types of the expressions consists only of "Status".
- Actions whose classifier does NOT be burdened by the improper reference signal and feature selection from the expressions.

This experiment was conducted as follows. The kernel parameters optimization executes with one collection of the motion set, which is exactly the same as the training set in the experiment of the batch learning. Next, the performance evaluation was done with 4 collections of the motion data. Then we compared the number of the support vectors in these classifiers with that in the classifiers acquired by the experiment of the batch SVM.

The experimental results show that the performance increases only 0.4% while the number of the support vectors dramatically mark a 34.2% decrease.

6 Conclusion

We designed and implemented a recognition system for human daily life action. The design principles of our former system were the simultaneous recognition, expressing the unclarity result in recognition, and utilizing expressions of action by human knowledge in order to realize recognition result close to humans recognition result. Meanwhile the proposed system in this paper takes over the above features, the modification of the recognition process to incorporate learning ability to the system is the main progress of this paper.

Motion captured files are utilized as the time series of input motion of the proposed system. In order that the system can output multiple recognition results simultaneously, the proposed system consists of recognition processes(AER) as many as the number of the target actions and the AERs run in parallel with the others.

The AER is designed to output multi-class decision result of one action to realize the unclarity in recognition and contains two binary classifiers. The binary classifier utilizes voting framework as linear combination of similarities between input motions and the sample motion data acquired in advance. By utilizing kernel technique to calculate similarity, the recognition and learning process can be unified to reduce the difference caused by the multi types of expressions of action by human.

The learning scheme we adopted is based on Support Vector Machine, which can executes with the training data(motion and humans judgment). In this paper, the extended learning algorithms of SVM, which can be computed in online situation, and kernel optimization as a part of model selection of kernel classifiers, are applied to the recognition system.

The experimental result showed that the performance of our system is superior to other classical classifier such as multi layered neural network optimized by back propagation algorithm, and decision tree generated by C4.5 algorithm.

It also proved that the property of the online learning are similar to batch SVM algorithm. The result of the evaluation of the property of kernel parameters optimization showed that it effectively reduces the number of Support Vectors.

In the days ahead, we intend to make algorithm to integrate the online learning algorithm and the kernel optimization algorithm. This is because that the parallel execution of two algorithms will have potential to more user friendly automatic optimization process and guarantee the optimality of its performance.

We also have plans to design new kernel function which can more effectively incorporate the expressions of action and to develop new learning algorithm incorporating prior knowledge of action. The expansion of target actions and the construction of knowledge database of action are also important work.

References

1. T. Starner and A. Pentland. Visual Recognition of American Sign Language Using Hidden Markov Models. In *International Workshop on Automatic Face and Gesture Recognition*, pp. 189–194, 1995.
2. A. Wilson and A. Bobick. Parametric Hidden Markov Models for Gesture Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 21, No. 9, pp. 884–900, September 1999.
3. T. Mori, K. Tsujioka, and T. Sato. Human-Like Action Recognition System on Whole Body Motion-captured File. In *Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2066–2073. IEEE/RSJ, 2001.
4. Claudette Cédras and Mubarak Shah. Motion-Based Recognition: A Survey. *Image and Vision Computing*, Vol. 13, No. 2, pp. 129–155, March 1995.
5. BVH File Format. http://www.biovision.com/bvh.html.

6. B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, 2002.
7. D. Mackay. Gaussian Processes A Replacement for Supervised Neural Networks? Lecture notes for a tutorial at NIPS 1997, 1997.
8. M. Tipping. Sparse Bayesian Learning and the Relevance Vector Machine. *Journal of Machine Learning Research*, Vol. 1, pp. 211–244, 2001.
9. C. Cortes and V. Vapnik. Support-Vector Networks. *Machine Learning*, Vol. 20, No. 3, pp. 273–297, 1995.
10. V. Vapnik. *The Nature of Statistical Learning Theory (Statistics for Engineering and Information Science)*. Springer Verlag, 1995.
11. G. Cauwenberghs and T. Poggio. Incremental and Decremental Support Vector Machine Learning. In *Advances in Neural Information Processing 13*, pp. 409–415. MIT Press, 2001.
12. O. Chapelle and V. Vapnik. Model Selection for Support Vector Machine. In S. Solla, T. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems 12*, pp. 230–236, Cambridge, MA, 2000. MIT Press.
13. J. Platt. Probabilities for SV Machines. In A. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pp. 61–74. MIT Press, 1999.
14. ICS Action Database. http://www.ics.t.u-tokyo.ac.jp/action/.
15. J. Quinlan. Induction of Decision Trees. *Machine Learning*, Vol. 1, No. 1, pp. 81–106, 1986.
16. C4.5 Release 8. http://www.cse.unsw.edu.au/ quinlan/c4.5r8.tar.gz.