Automatic learning of gesture recognition model using SOM and SVM

Masaki Oshita and Takefumi Matsunaga

Kyushu Institute of Technology
680-4 Kawazu, Iizuka, Fukuoka, 820-8502, Japan
oshita@ces.kyutech.ac.jp, matsunaga@cg.ces.kyutech.ac.jp

Abstract. In this paper, we propose an automatic learning method for gesture recognition. We combine two different pattern recognition techniques: the Self-Organizing Map (SOM) and Support Vector Machine (SVM). First, we apply the SOM to divide the sample data into phases and construct a state machine. Next, we apply the SVM to learn the transition conditions between nodes. An independent SVM is constructed for each node. Of the various pattern recognition techniques for multi-dimensional data, the SOM is suitable for categorizing data into groups, and thus it is used in the first process. On the other hand, the SVM is suitable for partitioning the feature space into regions belonging to each class, and thus it is used in the second process. Our approach is unique and effective for multi-dimensional and time-varying gesture recognition. The proposed method is a general gesture recognition method that can handle any kinds of input data from any input device. In the experiment presented in this paper, we used two Nintendo Wii Remote controllers, with three-dimensional acceleration sensors, as input devices. The proposed method successfully learned the recognition models of several gestures.

Keywords: gesture recognition, automatic learning, SOM, SVM.

1 Introduction

Gesture recognition has many potential applications, such as gaming interfaces (for example Nintendo Wii, Microsoft Kinect, Sony Move), control interfaces for robots [1], and electronic devices (for example TVs, lights, air conditioning) [12], severance systems [5], and so on. In general, gesture recognition determines what kind of action the user is performing from various input signals such as body position, velocity and orientation. The input signals can be acquired from various devices, such as sensor-embedded remote controllers, pressure sensors and cameras.

There are many pattern recognition techniques. However, the problem with gesture recognition is that the inputs usually consist of multi-dimensional and time-varying data. If the input data were either multi-dimensional or time-varying, existing pattern recognition techniques could easily be applied. For example, the Support Vector Machine (SVM) and neural networks including the Self-Organizing Map (SOM) work well on multi-dimensional data, while the Hidden Markov Model (HMM) and Dynamic Programming (DP) are well-suited to time-varying data. Owing to the

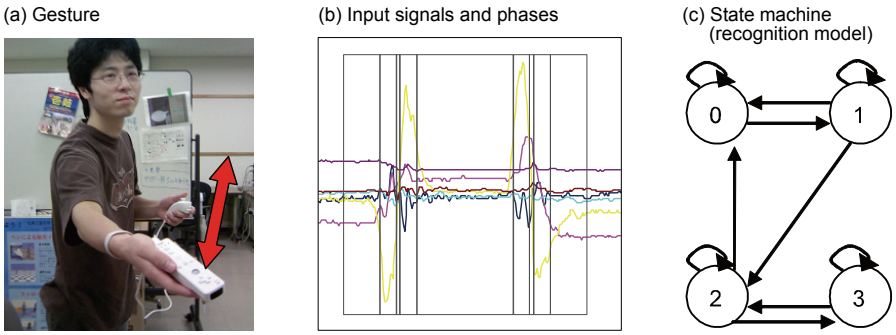

Fig. 1. Examples of gesture recognition using Wii remotes. (a) Performing a gesture. (b) Input signals and phases. The horizontal axis represents time. Each line represents one of the input values (in this case, accelerations), and each block a phase of the gesture. (c) State machine (recognition model). Each state represents a phase of the gesture and has conditions for transitions to connected states.

combination of multi-dimensional and time-varying data, gesture recognition introduces many problems.

A common approach to handling such data is to divide each gesture into short phases, and then to recognize each phase using a pattern recognition technique for multi-dimensional data [9]. For example, the gesture of raising and lowering a hand would contain phases such as stationary, moving upward, moving downward. By introducing a state machine, representing each phase as a state, and using a pattern recognition technique to determine the state transitions, we can construct a recognition model (Fig. 1). By preparing multiple recognition models for all gestures used in the application, the system can recognize any user gesture. The problem with this approach is that the state machine must be constructed manually. The designer must divide a gesture into several phases based on changes in the input signals. This requires much experience, effort, and trial and error processes.

The purpose of this research is to develop an automatic learning method for gesture recognition. We combine two different pattern recognition techniques: the Self-Organizing Map (SOM) [7] and Support Vector Machine (SVM) [1]. First, we apply the SOM to divide the sample data into phases and construct a state machine. Next, we apply the SVM to learn the transition conditions between nodes. An independent SVM is constructed for each node. Of the various pattern recognition techniques for multi-dimensional data, the SOM is suitable for categorizing data into groups, and thus it is used in the first process. On the other hand, the SVM is suitable for partitioning the feature space into regions belonging to each class, and thus it is used in the second process. Our approach is unique and effective for multi-dimensional and time-varying gesture recognition.

The proposed method is a general gesture recognition method that can handle any kinds of input data from any input device. In the experiment presented in this paper, we used two Nintendo Wii Remote controllers, with three-dimensional acceleration sensors, as input devices. The proposed method successfully learned the recognition models of several gestures.

The rest of this paper is organized as follows. In Section 2, we review related works. In Section 3, an overview of our method is presented, while Section 4 describes our method in detail. Experimental results and a discussion are presented in Section 5. Finally, Section 6 concludes the paper.

2 Related Work

There are many studies on gesture recognition techniques. As explained earlier, the problem is how to deal with multi-dimensional and time-varying input data.

The Hidden Markov Model (HMM) is a popular method for recognizing time-varying data. It represents a recognition model as a network of nodes that produce some symbols and transition probabilities between nodes. Since it handles input signals as a series of discrete symbols, in order to employ an HMM, the input signals must be reduced to a series of discrete symbols. This causes difficulty in handling multi-dimensional signals and prevents accurate recognition. Toyokura et al. [11] used manually specified conditions for discrimination and applied their approach to sign language recognition. Iuchi et al. [5] used an SOM for discretization. Liang et al. [8] used Principal Component Analysis (PCA) to discretize accelerations of multiple body parts.

Another method for recognizing time-varying data is dynamic programming. By matching the trajectory from the input signals and the sample trajectory from a gesture, the system can determine whether the gesture has been executed. However, to apply this approach, the trajectories must be projected onto a low-dimensional feature space. Moreover, an appropriate threshold must be specified to determine whether two trajectories are similar. Billon et al. [1] employed PCA for dimension reduction, while Yoshioka et al. [14] used manual discretization for each element of the feature vector.

Using a state machine is also a viable approach. Matsunaga et al. [9] used an SVM to learn the conditions for transitions, whereas Oshita [12] used manually described fuzzy-based rules. However, as explained above, the state machine must be created manually.

Even though some of these methods were designed for automatic learning of gesture recognition, certain parts or parameters need to be designed or specified for each gesture by a designer. On the contrary, our method realizes automatic learning.

3 System Overview

Our system consists of a learning and a recognition process. The learning process is an offline process, whereas the recognition process is an online process. The learning process takes input signals of sample data performed by a user as input, while the recognition process takes the current feature vector from the input device at each frame as input.

Each instance of the input signals from an input device is represented as a feature vector. Typically the dimension of the feature vector is between a few and several tens. In the experiment in this paper, we used six-dimensional feature vectors.

3.1 Recognition model

Our recognition model is represented as a state machine (Fig. 1(c)). Each state in the state machine represents a phase of the gesture. As explained in Section 1, gestures are expressed by time-varying data and the conditions of the feature vector for a gesture vary from time to time. Therefore, we divide a gesture into phases.

Each state of the state machine has a connection to other states and its own SVM to determine to which state to transit based on the current feature vector.

The recognition model also has an initial state and a recognition path. The recognition path is a series of states. As the system transits through states according to the inputs, forming the recognition path, it determines that the gesture has been performed. In addition, motion speed can also be calculated from the intervals of the transitions [9].

Each recognition model recognizes only one type of gesture. Since a system usually uses multiple gestures, multiple recognition models must be constructed and processed for recognition in parallel. Each recognition model is trained independently for a specific gesture.

3.2 Learning process

The learning process takes input signals of sample data performed by a user as input. A single sample data contains a series of feature vectors obtained during the performance of the gesture. To train a recognition model, a number of sets of sample data are required. For example, in our experiments, we used 10 sets of sample data. Since 16 feature vectors per second are recorded in our system and the length of a typical gesture is between a few seconds and 10 seconds, each sample data consists of a series of about 100 to 200 feature vectors.

Firstly, the system applies an SOM to categorize the feature vectors into a small number of groups (units). A state of the state machine is constructed from each group (unit). We use an SVM to represent the conditions for transitions in each node. Details of the learning process are explained in the next section.

3.3 Recognition process

During the recognition process, the system takes the current feature vector from the input device at each frame as input. Based on the current feature vector, the system determines to which state to transit next or whether to stay in the current state. The system determines if a particular gesture has been performed based on the history of the state transitions and the recognition path as explained in Section 3.1.

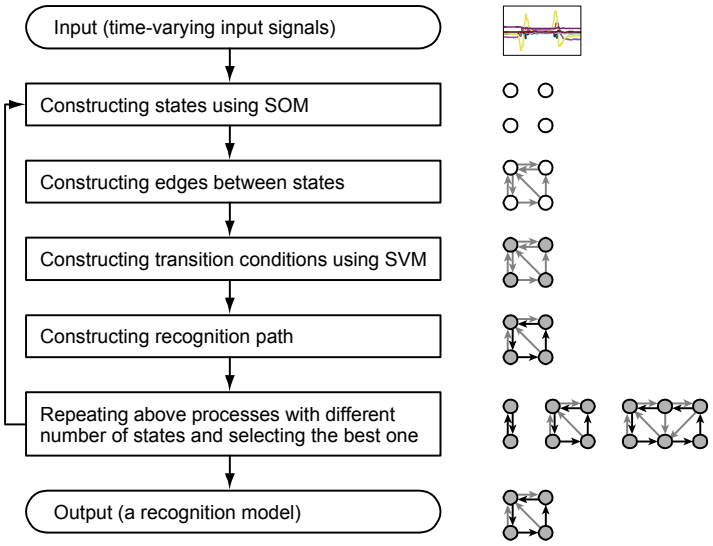

Fig. 2. Flow chart and data structures for the learning process.

4 Automatic Learning of Recognition Model

In this section, we explain our automatic learning method, which involves constructing states using an SOM, edges between states, transition conditions using an SVM, and finally the recognition path (Fig. 2). In the following subsections, we explain each of these processes.

4.1 Constructing states using an SOM

We use an SOM to construct the states of the state machine. The SOM is a pattern recognition technique incorporating unsupervised learning and a type of neural network. It is widely used in many applications for classifying various kinds of data such as images, voices, medical data, genes, etc. The SOM is suitable for categorizing sample data. It maps high-dimensional data into a number of units, which are usually arranged as a two dimensional grid. As result, it groups similar data into the same or a nearby unit.

We simply apply an SOM to all feature vectors from the sample data and then use each unit as a state in the state machine. Each state contains the feature vectors belonging to the corresponding unit and these are used in the following processes. Units with no data are ignored and no states are constructed from them.

In general, when applying an SOM, the number of units must be specified manually. If the number is too small, dissimilar data may be mapped to the same unit.

On the other hand, if the number is too high, similar data may be mapped to different units causing too many states to be created in our system. This can cause recognition errors. In many applications, the number of units is chosen empirically. However, in our case, the number depends on the type of input and the gesture to be recognized. Therefore, we introduce automatic selection of the number of units. Our system constructs several state machines (learning models) with different numbers of units. It then evaluates these using the sample data. Finally, the best state machine that minimizes the following criterion is chosen.

$$e = e_1 + k\,e_2 \tag{1}$$

where e_1 is the error rate that correct inputs are not recognized, e_2 is the error rate that incorrect inputs are recognized, and k is a parameter denoting the weights, specified by the user, of e_1 and e_2. The appropriate value of k depends on the application. For example, in an application such as a gaming interface, a small e_1 is important and a small value is specified for k. On the other hand, in an application such as a security system, a small e_2 is important and thus, a large value is specified for k.

4.2 Constructing edges between states

Once the states of the state machine have been created, our system determines possible transitions for each state. If a feature vector of state A is followed by a feature vector of state B in the input sample data, an edge from state A to state B is created. In addition, each state has an edge to itself and an edge to the initial state, as the user could terminate a gesture at any time and start from the beginning.

4.3 Constructing transition conditions using an SVM

We use an SVM to learn the transition conditions for each state. The SVM is a popular pattern recognition technique with supervised learning. Since it divides the feature space for each class, the SVM can handle unknown data well, although it is not suited to grouping sample data. This is the reason that we combine two pattern recognition techniques, the SOM and SVM, with different characteristics.

To construct an SVM for each state, the sample data belonging to the state and those belonging to adjacent (possible transition) states are used as training data. In our implementation, we employ LibSVM [3], which supports multi-class recognition.

Once the SVM has been trained, the transition to the next state is determined based on the current feature vector. The transition could be to remain in the same state or to move to one of the adjacent states.

4.4 Constructing recognition path

A state machine is constructed from the above processes. Finally, the recognition path is determined. At this point, we have an initial state, to which the initial feature

vectors belong, and a terminal state, to which the terminal feature vectors belong. However, even if the system transits from the initial state to the terminal state, it does not necessarily mean that the particular gesture has been performed. Generally, a state machine is a next work of states. Therefore, there can be many paths from the initial state to the terminal state and not all paths mean that the gesture has been performed.

Our system determines a correct path for gesture recognition based on the input sample data. A sample data is represented by a series of states. The system stores the recognition path and uses it for recognition. When there are different recognition paths for different sample data, all recognition paths are used in the recognition process.

5 Experiment and Discussion

We have tested our method with various gestures. In this experiment, we used Wii Remote controllers as input devices. With these controllers, the user has one spatial acceleration sensor for each hand. An acceleration sensor detects three directional (X, Y and Z) accelerations. Therefore, the feature vector has a dimension of six. We used two kinds of gestures in the experiments. A simple gesture (Gesture A) consists of raising and then lowering the right hand while keeping the left hand still. A complex gesture (Gesture B) consists of pushing both hands forward, swinging both hands to the right, swinging both hands to the left, and extending both hands outwards. Note that Wii Remote controllers have only an acceleration sensor. Therefore, even these simple gestures are difficult to recognize, because the positions of the controllers cannot be used. We used 10 sets of sample data for training each recognition mode; that is, an individual performed each gesture 10 times and the input signals were recorded for the learning process.

As explained in Section 4.1, to determine the appropriate number of units for the SOM, the system repeats the learning and recognition processes with a different number of units: two, four, and six. Both gestures (A and B) achieved the best recognition rate with four units. Therefore, four units (four states) were used in the following experiment.

To evaluate the effectiveness of our automatic learning method, especially the effectiveness of using the SOM, we compared the results of our method and a manual method. In the manual method, a participant who is a graduate student with basic knowledge of gesture recognition manually labeled each of about 2,000 feature vectors of input sample data by checking the images on the screen. For the number of states, we used the number determined automatically by our method. Once the categorization was done, we used our automatic method for the later processes, since manually specifying transition conditions requires much trial and error and is not practical.

Results of the above experiment are shown in Table 1. Obviously, the manual categorization required a great deal of time compared to our automatic method, especially for complex gestures. On the other hand, the manual method showed slightly better results. This is because the classification using an SOM does not work well in the presence of noise. When a human finds noise in a series of feature vectors,

he/she is able to categorize these according to the previous and following feature vectors. However, since our method simply categorizes all feature vectors independently, a sudden transition to a wrong node could happen as a result of the noise. To solve this problem, an additional process to correct the categorization error could be introduced.

Table 1. Comparison between the proposed automatic method and manual method

Method / Gesture	Time for learning [sec]	Recognition rate [%]
Manual / Gesture A	510	100
Automatic (Proposed) / Gesture A	4	98
Manual / Gesture B	5100	84
Automatic (Proposed) / Gesture B	8	80

6 Conclusion

In this paper, we proposed a method for automatically learning a recognition model by incorporating both an SOM and an SVM. We also showed the efficiency thereof.

Gesture recognition has many potential applications. Applying our method to various actual applications with higher dimensions remains a future work. For most applications, camera-based sensors are suitable, since users do not have to hold or wear any input devices. However, in order to apply our method to input signals from cameras, a method for extracting feature vectors from images is needed. In addition, when applying our method to very high-dimensional data, for example, motion data acquired using motion capture equipment and which could have more than 40 dimensions, we would need a method for dimension reduction, such as PCA. Further research on these extensions as well as performing further experiments for comparison with alternative methods are also future works.

References

1. Ronan Billon, Alexis Nedelec and Jacques Tisseau, “Gesture recognition in flow based on PCA and using multiagent system”, Proc. of Advances on Computer Entertainment 2008, pp.139-146, 2008.
2. Nello Cristianini and Jhon Shawe-Taylor, “An Introduction to Support Vector Machines”, Cambridge University, 2000.
3. Chih-Chung Chang, Chih-Jen Lin, 2002. LIBSVM: a Library for Support Vector Machines, http://www.csie.ntu.edu.tw/~cjlin/libsvm/index.html, 2001.
4. Kota Irie, Naohiro Wakamura, and Kazunori Umeda, "Construction of an Intelligent Room Based on Gesture Recognition -Operation of Electric Appliances with Hand Gestures-", Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp.193-198, 2004.
5. Hirotaka Iuchi, Sakashi Maeda and Naoyuki Tsuruta, “Gesture Recognition using Self-Organizing Maps and Hidden Markov Model”, IPSJ SIG Notes, Computer Vision and Image Media, Vol. 2001, No. 36, pp. 127-134, 2001.

6. Taichi Joutou, Keiji Yanai, “Web Image Classification with Bag-of-keypoints”, IPSJ SIG Notes, Computer Vision and Image Media, Vol. 2007, No. 42, pp. 201-208, 2007.
7. Teuvo Kohonen, “The self-organizing map”, Proc. of the IEEE. vol. 78, no. 9. pp. 1464-1479, 1990.
8. Xiubo Liang, Qilei Li, Xiang Zhang, Shun Zhang, Weidong Geng, “Performance-driven motion choreographing with accelerometers”, Computer Animation and Virtual Worlds, vol. 20 issue 2-3, pp. 89-99, 2009.
9. Takefumi Matsunaga and Oshita Masaki, “Recognition of Walking Motion Using Support Vector Machine”, Proc. of ISICE 2007, pp. 337-342, 2007.
10. Takuya Nanri, and Nobuyuki Otsu, “Anomaly Detection in Motion Images Containing Multiple Persons”, Proc. of PRMU 2004-77, Vol. 104, No. 291, pp. 583-588, 2004.
11. Kenichi Noma, Mitsuru Nakai, Hiroshi Shimodaira, and Shigeki Sagayama, “Sequential-Pattern Recognition by Support Vector Machines using Dynamic Time-Alignment Kernels”, Technical report of IEICE, Vol. 100, No. 507, pp. 63-68, 2000.
12. Masaki Oshita, “Motion-Capture-Based Avatar Control Framework in Third-Person View Virtual Environments”, ACM SIGCHI International Conference on Advance in Computer Entertainment Technology 2006, 2006.
13. Toshiya Yamada, and Kazunori Umeda, “Improvement of the Method of Operating a Mobile Robot by Gesture Recognition”, JSME Conference on Robots and Mechatronics, Vol. 2001, p. 49, 2001.
14. Taichi Yoshioka, Hisashi Koga, Toshinori Watanabe, Takanori Yokoyama, “Online Automatic Acquisition of Human Motion Models with Voting ”, Technical report of IEICE, Vol. 105, No. 302, pp. 119-124, 2005.
15. Yukitaka Toyokura and Yoshihiko Nankaku et al. “Approach to Japanese Sign Language Word Recognition using Basic Motion HMM”, Proceedings of the Society Conference of IEICE, Vol. 2006, pp. 72, 2006.