Hindawi Publishing Corporation
Advances in Human-Computer Interaction
Volume 2015, Article ID 785349, 7 pages
http://dx.doi.org/10.1155/2015/785349

Research Article

Dynamic Arm Gesture Recognition Using Spherical Angle Features and Hidden Markov Models

Hyesuk Kim and Incheol Kim

Department of Compute Science, Kyonggi University, San 94-6, Yiui-Dong, Youngtong-Gu, Suwon-Si, Gyeonggi-Do 443-760, Republic of Korea

Correspondence should be addressed to Incheol Kim; kic@kgu.ac.kr

Received 7 June 2015; Accepted 21 October 2015

Academic Editor: Thomas Mandl

We introduce a vision-based arm gesture recognition (AGR) system using Kinect. The AGR system learns the discrete Hidden Markov Model (HMM), an effective probabilistic graph model for gesture recognition, from the dynamic pose of the arm joints provided by the Kinect API. Because Kinect's viewpoint and the subject's arm length can substantially affect the estimated 3D pose of each joint, it is difficult to recognize gestures reliably with these features. The proposed system performs the feature transformation that changes the 3D Cartesian coordinates of each joint into the 2D spherical angles of the corresponding arm part to obtain view-invariant and more discriminative features. We confirmed high recognition performance of the proposed AGR system through experiments with two different datasets.

1. Introduction

Gestures are a powerful human-to-human communication modality and the expressiveness of gestures also allows for the altering of perceptions in human-computer interaction [1]. Vision-based gesture recognition technology can be applied to multiple fields including human-robot interaction [2], computer game [3], sign language understanding for the hearing-impaired [4], and other fields [5–7]. Kinect, recently released by Microsoft, provides not only RGB images but also depth images at a low cost. Therefore, with the release of low-cost 3D sensors like Kinect, the dynamic gesture recognition technology has gained increased attention.

In this paper, we introduce a Kinect-based arm gesture recognition (AGR) system design. The AGR system learns the discrete Hidden Markov Model (HMM), an effective probabilistic graphical model for gesture recognition, from the dynamic pose of the arm joints provided by the Kinect API. Because the variance of Kinect's viewpoints and the different length of the subject arms can significantly affect the estimated 3D pose of each arm joint, it is difficult to recognize gestures reliably with these features. In order to overcome this problem and obtain view-invariant features, the AGR system performs the feature transformation that changes the 3D Cartesian coordinates of each joint into the 2D spherical angles of the corresponding arm part. For evaluating the performance of the AGR system, we conduct experiments with two different datasets and then introduce the results.

2. Related Works

Major approaches to vision-based gesture recognition include support vector machine (SVM), Dynamic Time Warping (DTW), and Hidden Markov Models (HMMs). The work of [8] uses discrete HMMs to analyze spatial and temporal patterns of 2D hand gestures. In this work, a gesture is described as a spatial-temporal sequence of feature vectors that consist of the direction of hand movement. For each gesture, one HMM is built to learn the temporal characteristics of gesture signals. Gesture spotting is the task of locating the start point and the end point of a gesture pattern. In this work, an additional HMM is built as a threshold model for gesture spotting.

In the work of [9], a method is presented to recognize 3D human body gestures from sequences of depth images. It uses 3D motion trail model (3D-MTM) to explicitly represent the dynamics and statics of gestures in 3D space. And then, the Histogram of Oriented Gradient (HOG) feature vector

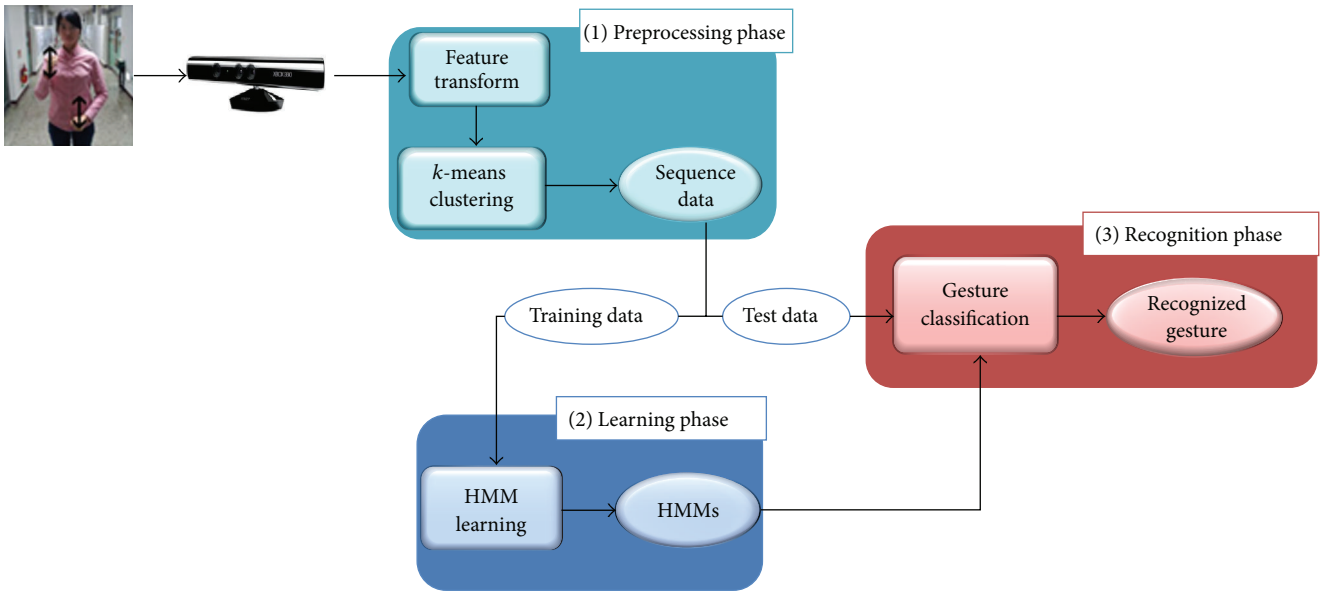


Figure 1: Overall process of arm gesture recognition.

is extracted from the 3D-MTM as the representation of a gesture sequence. In this work, SVM is adopted to classify the gestures.

Dynamic Time Warping (DTW) is a template matching algorithm and is one of the techniques used in gesture recognition. To recognize a gesture, DTW warps a time sequence of joint positions to reference time sequences and produces a similarity value. However, all body joints are not equally important in computing the similarity of two sequences. In the work of [10], a weighted DTW method is proposed that weights joints by optimizing a discriminant ratio.

Affinity propagation (AP) is a clustering algorithm which, unlike all other clustering techniques, simultaneously considers all data points as potential exemplars and recursively transmits real-valued messages until a good set of exemplars and clusters emerge. The work of [11] proposes a gesture recognition system based on a single 3-axis accelerometer. In order to improve the efficiency of DTW-based gesture recognition, the system reduces the number of exemplars for each gesture class by employing AP and DTW algorithms (AP + DTW) in the training phase. As a result, the output of the training phase is a finite set of exemplars, one for each class.

3. Arm Gesture Recognition System

3.1. System Overview. The arm gesture recognition system proposed herein admits the input stream of 3D position coordinates of the arm joints from depth images of Kinect when a user makes one of the predefined dynamic gestures with his arms in front of Kinect. And then, using the learned HMMs, the system determines what gesture is being made.

The overall process of the AGR system consists of three steps: the preprocessing step, learning step, and recognition step, as shown in Figure 1. The preprocessing step extracts the feature vectors, which of each consists of the 3D position coordinates of the arm joints including the shoulders, elbows, and wrists, from Kinect's depth images using Kinect API at 20 frames per second. The joint coordinate features are then transformed into the joint angle features that are insensitive to the length of the arms and Kinect's view. Then, the transformed high-dimensional real-number feature vectors are replaced with the index number of the corresponding cluster through k-means clustering. At the final stage of preprocessing, the training or testing sequence data are constructed for each gesture by collecting series of such index numbers.

In the learning step, multiple training sequence data are used to learn the HMM for each gesture. The HMM used in our system is a discrete and left-right model that allows state transition only from left to right and one of k different observations for each state. The algorithm used for learning each HMM is the Baum-Welch algorithm, an EM (Expectation and Maximization) algorithm. Finally, in the recognition step, the HMM for each gesture is applied to the testing sequence data that are input in real-time through Kinect to determine the maximum log likelihood gesture.

3.2. Preprocessing. Figure 2 shows the process of preprocessing in the arm gesture recognition system. The Kinect API from Microsoft presents the position of the arm joints including two shoulders, elbows, and wrists of the subject as 3D Cartesian coordinate vectors in the form of (x, y, z) from the depth images. For example, the 3D position of the right shoulder joint j_{RS} is given as a vector (x_{RS}, y_{RS}, z_{RS}). As mentioned before, this 3D position of each arm joint in Cartesian coordinate system, the origin of which $(0, 0, 0)$ is placed on the center of Kinect, is sensitive to the variance of Kinect's viewpoints and the different length of the subject arms. For this reason, In this study, we perform the feature

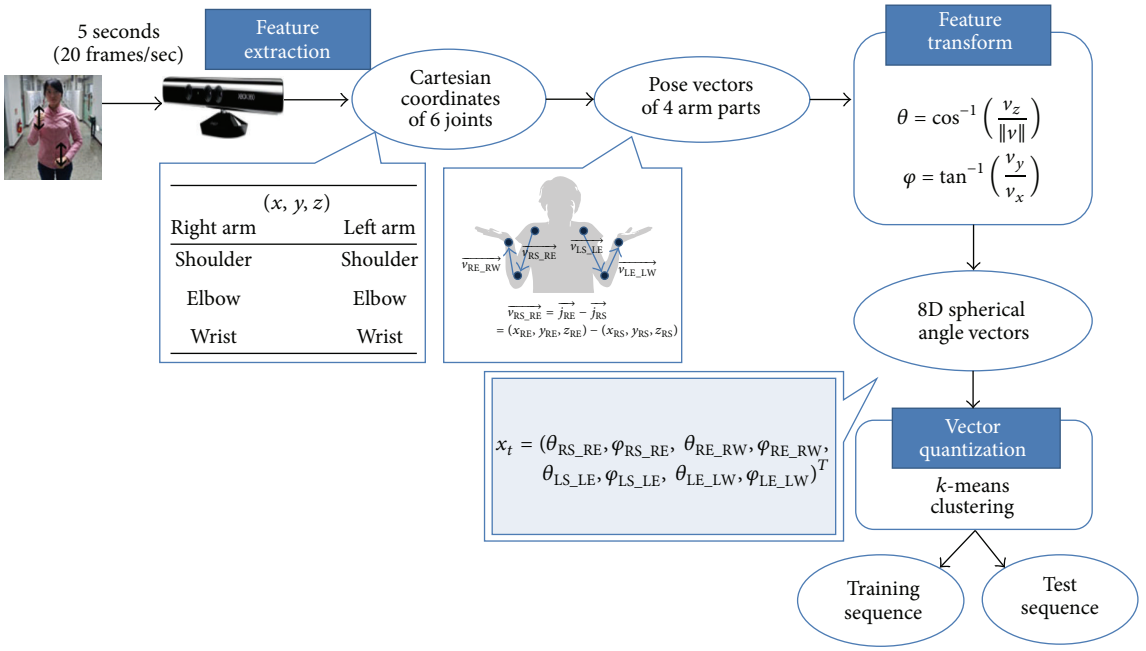
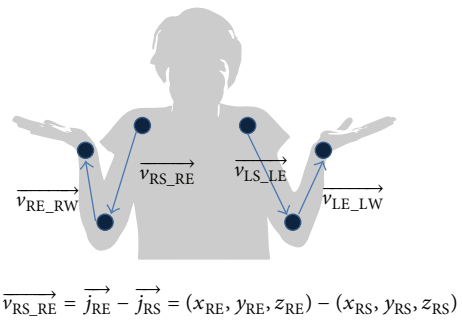


FIGURE 2: The preprocessing step.



$\overrightarrow{v_{RS\_RE}} = \overrightarrow{j_{RE}} - \overrightarrow{j_{RS}} = (x_{RE}, y_{RE}, z_{RE}) - (x_{RS}, y_{RS}, z_{RS})$

FIGURE 3: Pose vectors of four arm parts.

transformation that changes the 3D Cartesian coordinate vectors representing the joint positions, j_{RS}, j_{RE}, j_{RW}, j_{LS}, j_{LE}, and j_{LW}, into the set of 2D spherical angle vectors (θ, φ) representing the directions of the corresponding arm parts.

First, we consider the four parts of both arms: the upper and the lower part of the right arm and the upper and the lower part of the left arm. Real-time pose vectors of these four arm parts, v_{RS_RE}, v_{RE_RW}, v_{LS_LE}, and v_{LE_LW}, are derived from 3D position vectors of six arm joints, j_{RS}, j_{RE}, j_{RW}, j_{LS}, j_{LE}, and j_{LW}, as shown in Figure 3. For example, the pose vector of the upper part of the right arm, v_{RS_RE}, is computed from two 3D position vectors of the right shoulder joint and the right elbow joint, j_{RS} and j_{RE}.

Next, the 3D pose vectors of arm parts in the form of $v = (v_x, v_y, v_z)$ are transformed into the set of 2D spherical angle vectors in the form of (θ, φ), as shown in Figure 4.

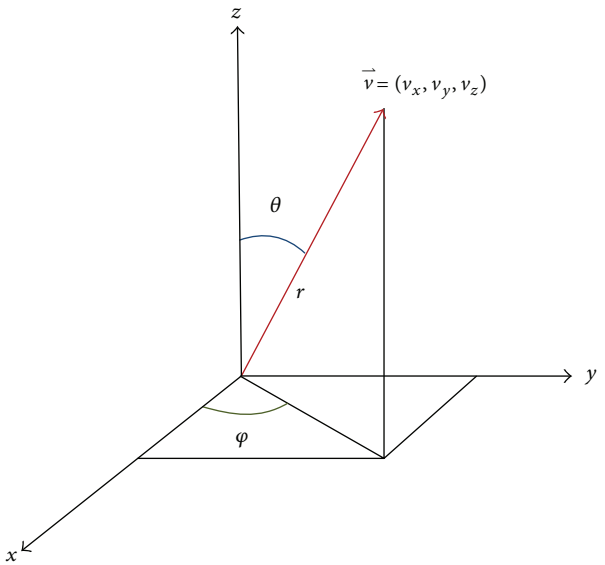


FIGURE 4: Transformation of pose vector (v_x, v_y, v_z) into spherical angle vector (θ, φ).

The transformation is performed based upon

$$\theta = \cos^{-1}\left(\frac{v_z}{\|v\|}\right), \qquad \varphi = \tan^{-1}\left(\frac{v_y}{v_x}\right), \tag{1}$$

where θ and φ are the polar and the azimuthal angle of the pose vector v, respectively. Note that the radial distance

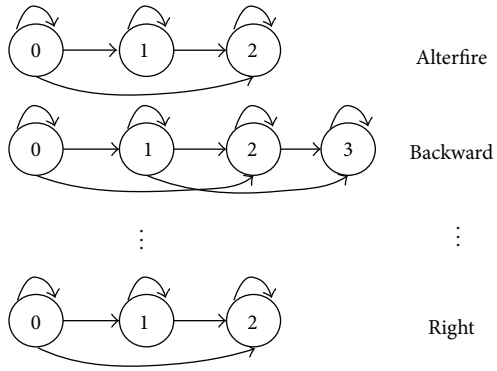


FIGURE 5: Hidden Markov Models (HMMs) learned for arm gesture recognition.

$r = \|v\|$ is omitted in the spherical angle representation. This means that the resulting feature set excludes features sensitive to the length of arm parts but instead includes only the direction of them. The final feature vector X_t of time t is built by combining 2D spherical angle vectors of 4 arm parts into a single vector, as described by (2). For example, $\theta_{\text{RS\_RE}}$ represents the polar angle of the upper part of the right arm. Through feature transformation, therefore, we obtain the stream of 8D spherical angle vectors to be used to train or test gesture models

$$X_t = (\theta_{\text{RS\_RE}}, \varphi_{\text{RS\_RE}}, \theta_{\text{RE\_RW}}, \varphi_{\text{RE\_RW}}, \theta_{\text{LS\_LE}}, \varphi_{\text{LS\_LE}}, \theta_{\text{LE\_LW}}, \varphi_{\text{LE\_LW}}). \quad (2)$$

However, if the transformed high-dimensional feature vectors were applied directly to the HMM-based gesture recognition, it would overload the computational complexity on model learning and also delay the response time of the gesture recognition. To resolve this problem, in the preprocessing step of the proposed system, the eight-dimensional (8D) real-number feature vectors are clustered into k number of clusters through k-means clustering. And then, whenever each vector is met from the stream of 8D feature vectors, it is replaced with the index number of the cluster that this vector belongs to. At the final stage of preprocessing, the training or testing sequence data are constructed for each gesture by collecting series of such index numbers.

3.3. Model Learning. In the learning step, the training dataset for each gesture is used to learn the Hidden Markov Model (HMM) for this gesture. The HMM is a generative probabilistic graphical model, in which the target system to be modeled is assumed to be a Markov Process. This model contains the observation variable and the hidden state variable. The HMM assumes that the conditional probability distribution of the hidden state variable x_t at time t depends only on the value of the hidden variable x_{t-1}. Similarly, it assumes that the value of the observation variable y_t only depends on the value of the hidden variable x_t. The HMM used in our system is a discrete and left-right model that allows state transition only from left to right, as shown in Figure 5.

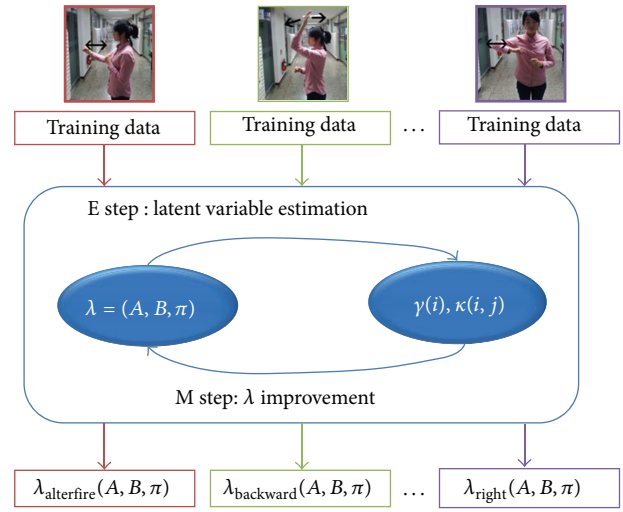


FIGURE 6: Baum-Welch algorithm for learning HMMs.

In this work, the hidden state variable of the HMM is allowed to have 3~5 different states depending on the complexity of the corresponding gesture. However, the observation variable of every HMM can have only the fixed number of k different values, due to the vector quantization using k-means clustering in the preprocessing step as explained before. A HMM has three parameters: the initial state probability distribution π, the state transition probability distribution A, and the observation probability distribution B. In the learning step, the system finds the optimal parameters of the HMM for each gesture using the training datasets. The Baum-Welch algorithm shown in Figure 6 is used to learn the HMM for each gesture. The Baum-Welch algorithm is an EM (Expectation and Maximization) learning algorithm and learns the optimal parameters of the HMM for each gesture by repeating the E step, the estimation of the hidden variables $\gamma(i)$ and $\kappa(i, j)$, and the M step, the improvement of the model parameters $\lambda = (A, B, \pi)$.

3.4. Gesture Recognition. In the gesture recognition step, the learned HMMs for individual gestures are applied to the testing sequence data to compute the log likelihood of the testing data for each HMM. Then, the best HMM with the maximum log likelihood is found to recognize the testing data as the corresponding gesture, as formulated in

$$g^* = \arg\max_{g_k \in G} \left\{ P\left(X \mid \lambda_{g_k}\right) \right\}. \quad (3)$$

Equation (3) uses the trained HMM with parameters λ_{gk} for each gesture $g_k \in G$, to compute the log likelihood $P(X \mid \lambda_{gk})$ of the testing data X, to find the best matching HMM with the maximum log likelihood, and to recognize it as the corresponding gesture g^*.

4. Experiments

Based on the design explained in the previous section, we implemented an arm gesture recognition system (AGR) on

Figure 7: Screenshot of the UT game.

Table 1: Experimental result for Kyonggi dataset.

Gestures	# of clusters		
	10	20	30
Alterfire	9/10	10/10	10/10
Backward	10/10	10/10	10/10
Fire	9/10	8/10	10/10
Forward	10/10	10/10	10/10
Left	8/10	9/10	10/10
Nextweapon	10/10	10/10	10/10
Preweapon	10/10	10/10	10/10
Right	9/10	10/10	10/10
Accuracy (%)	93.7	96.2	100

Windows 7 using Java and Matlab. We tested the performance of the AGR system with the Kyonggi dataset, a dataset we collected for this study, and the Cornell dataset [12], an open source dataset for research.

The Kyonggi dataset includes eight dynamic arm gestures to be used when playing the Unreal Tournament (UT) game [13] as shown in Figure 7. The gestures included in the Kyonggi dataset are presented in Figure 8. The Kyonggi dataset contains a total of 240 gesture data that were collected from three test subjects who made eight gestures, ten times each, at 1.5~2 m from Kinect. The gesture data were change sequences of the 3D positions (x, y, and z) of the six joints (right/left shoulders, right/left elbows, and right/left wrists) that were collected from Jnect API that taped the gestures for five seconds at 20 frames per second.

The Cornell dataset includes fifteen static arm gestures and fifteen dynamic arm gestures. The dynamic gestures included in the Cornell dataset are presented in Figure 9. The Cornell dataset contains a total of 900 gesture data mainly for aircraft guidance. For our experiments, we used 450 data representing the dynamic arm gestures. The Kyonggi dataset and the Cornell dataset were divided into 20 pieces of training data and 10 pieces of test data for each gesture in our experiments.

Table 1 shows the test results of the proposed gesture recognition system with the Kyonggi dataset. The number of clusters used in k-means clustering to discretize high-dimensional feature vectors may affect the performance of the HMM-based gesture recognition system. Thus, the recognition accuracy of the proposed system was measured for each gesture when the number of clusters k was 10, 20, and 30. As a result, our proposed system showed high accuracies, ranging from 93.7% to 100%. In addition, as the number of clusters k increased, the recognition accuracy improved.

Figures 10(a) and 10(b) show the average recognition performance of the proposed gesture recognition system on the Kyonggi and Cornell datasets. The bar graphs in the figures represent the average recognition accuracies when the number of clusters k was 10, 20, and 30. As shown in Figure 10(a), the recognition accuracy of the Kyonggi dataset improved as the number of clusters k increased. However, as shown in Figure 10(b), the recognition accuracy of the Cornell dataset did not improve in proportion to the number of clusters. In our experiment, the highest recognition accuracy was 98.2% when the number of clusters k was 20.

Figure 11 shows a comparison of the proposed gesture recognition system, which uses the HMM, with a system (AP + DTW) that uses AP clustering and Dynamic Time Warping (DTW). The AP + DTW system, one of the representative gesture recognition systems, presets a few exemplars that can present each gesture by applying AP (Affinity Propagation) clustering to the given training dataset in the model-learning step.

In the gesture recognition step, the system calculates similarity between the new test data and the exemplar data using Dynamic Time Warping (DTW), recognizing the gesture of the test data. As shown in Figures 11(a) and 11(b), the AP + DTW gesture recognition system showed 86% and 95% accuracies on the Kyonggi dataset and the Cornell dataset, while the proposed HMM-based recognition system showed higher than 99% accuracy on both datasets. That is, the proposed HMM-based system showed recognition performance 4~15% better than the AP + DTW-based recognition system.

5. Conclusions

In this work, we introduced an effective arm gesture recognition system using Kinect. The proposed system performs the feature transformation that changes the 3D Cartesian coordinates of each joint into the 2D spherical angles of the corresponding arm part to obtain view-invariant and more discriminative features. In order to represent the unique temporal pattern of each dynamic gesture, the system uses a discrete HMM which is a kind of probabilistic graphical models. Through experiments with two different datasets, the Kyonggi and Cornell datasets, we confirmed the high recognition performance of the proposed system.

Conflict of Interests

The authors confirm that this paper content has no conflict of interests.

Acknowledgment

This work was supported by the GRRC program of Gyeonggi province.

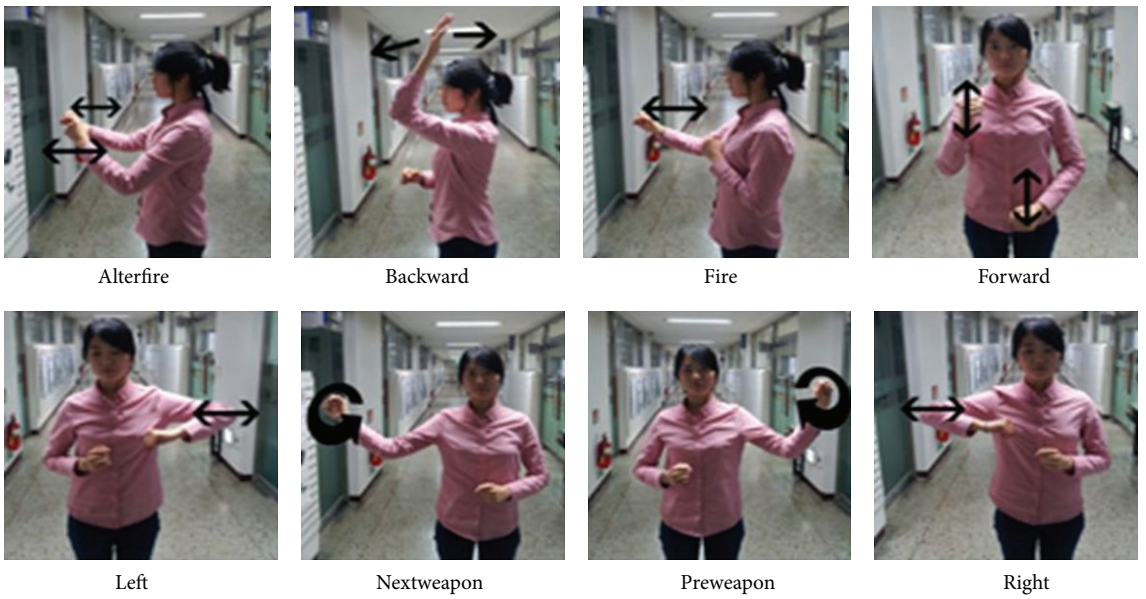


Figure 8: Gestures in Kyonggi dataset.

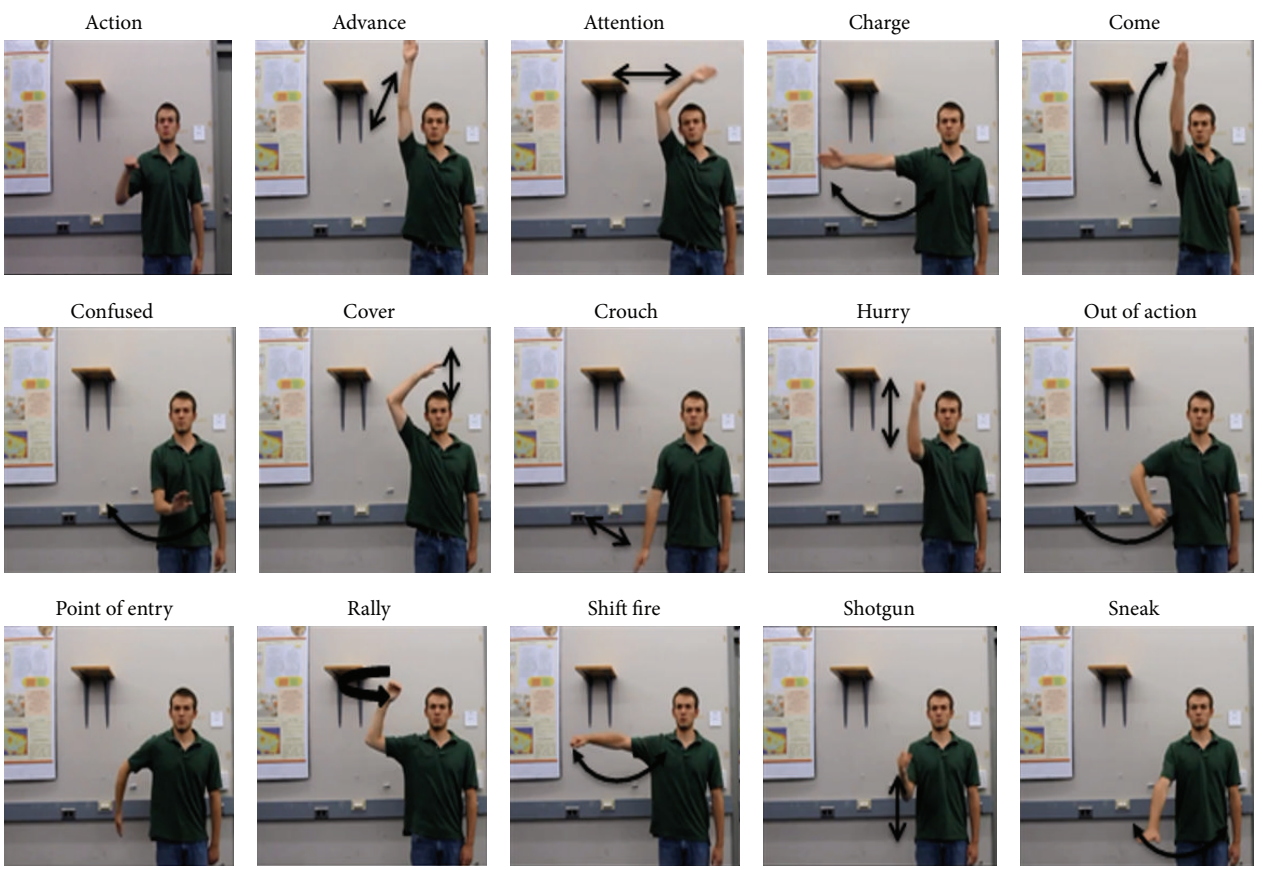
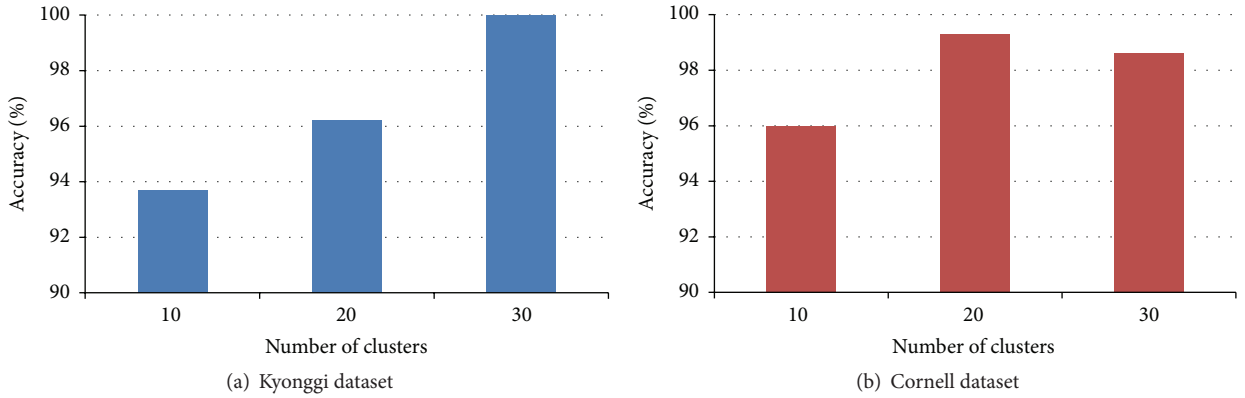
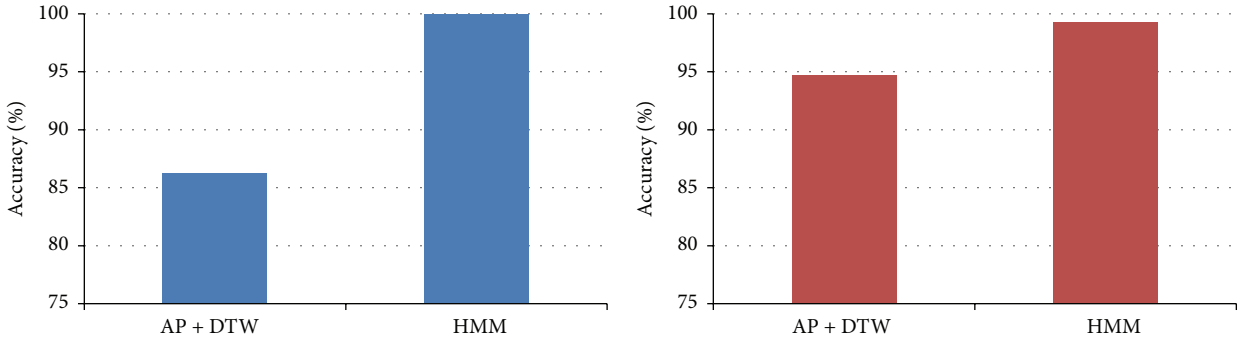


Figure 9: Gestures in Cornell dataset.



(a) Kyonggi dataset (b) Cornell dataset

Figure 10: Recognition accuracy with different numbers of clusters.



(a) Kyonggi dataset (b) Cornell dataset

Figure 11: Recognition accuracy with different learning models.

References

[1] G. Niezen and G. P. Hancke, "Gesture recognition as ubiquitous input for mobile phones," in *Proceedings of the International Workshop on Devices that Alter Perception (DAP '08)*, September 2008.

[2] M. Peris and K. Fukui, "Both-hand gesture recognition based on KOMSM with volume subspaces for robot teleoperation," in *Proceedings of the IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER '12)*, pp. 191–196, Bangkok, Thailand, May 2012.

[3] Y. Li, "Multi-scenario gesture recognition using Kinect," in *Proceedings of the 17th International Conference on Computer Games (CGAMES '12)*, pp. 126–130, Louisville, Kan, USA, August 2012.

[4] S. Lang, M. Block, and R. Rojas, "Sign language recognition using Kinect," in *Artificial Intelligence and Soft Computing*, vol. 7267 of *Lecture Notes in Computer Science*, pp. 394–402, 2012.

[5] O. Chagnaadorj and J. Tanaka, "Gesture input as an out-of-band channel," *Journal of Information Processing Systems*, vol. 10, no. 1, pp. 92–102, 2014.

[6] J. C. Augusto, V. Callaghan, D. Cook, A. Kameas, and I. Satoh, "Intelligent environments: a manifesto," *Human-Centric Computing and Information Sciences*, vol. 3, article 12, 2013.

[7] C. K. Ng, J. G. Fam, G. K. Ee, and N. K. Noordin, "Finger triggered virtual musical instruments," *Journal of Convergence*, vol. 4, no. 1, pp. 39–46, 2013.

[8] H.-K. Lee and J. H. Kim, "An HMM-Based threshold model approach for gesture recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 10, pp. 961–973, 1999.

[9] B. Liang and L. Zheng, "Three dimensional motion trail model for gesture recognition," in *Proceedings of the International Conference on Computer Vision, Big Data in 3D Computer Vision Workshop*, 2013.

[10] S. Celebi, A. S. Aydin, T. T. Temiz, and T. Arici, "Gesture recognition using skeleton data with weighted dynamic time warping," in *Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP '13)*, Barcelona, Spain, February 2013.

[11] A. Akl and S. Valaee, "Accelerometer-based gesture recognition via dynamic-time warping, affinity propagation, & compressive sensing," in *Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP '10)*, pp. 2270–2273, IEEE, Dallas, Tex, USA, March 2010.

[12] G. Bernstein, N. Lotocky, and D. Gallaghet, "Robot recognition of military gestures," Tech. Rep. CS4758, Term Project, 2011.

[13] L. M. Surhone, M. T. Tennoe, and S. F. Henssonow, *Unreal Tournament*, Betascript Publishing, 2011.