Real-Time Gesture Recognition Usind 3D Images

Osama Dweik
Master of Informatics
Deanship of Graduate Studies and Scientific Research
Palestine Polytechnic University
Hebron - Palestine
Email: osamdweik@yahoo.com

Hashem Tamimi
Department of Information Technology
College of Administrative Sciences and Informatics
Palestine Polytechnic University
Hebron - Palestine
Email: htamimi@ppu.edu

***Abstract*—KINECT has been recently introduced in the market as a low cost 3D acquisition device, so it will be interesting to discover the power of this device when we use it for gesture recognition. In this work, we propose an real-time gesture recognition system using 3D sensor that transforms gestures into a set of useful words using different machine learning algorithms and taking into consideration temporal features. A depth image, which is provided by KINECT, will be used to construct a skeleton of the human body. We have used Nearest Neighbor (NN) with different distance formulas, Self Organizing Map (SOM) and Hidden Markov Model (HMM) for recognition. The result of this work shows that using HMM, we obtain recognition accuracy around 95 percent, and using NN algorithm with Spearman distance we obtain around 90 percent and around 60 percent of accuracy using the SOM algorithm. All three algorithms works in real-time, we have used 10 fold cross validation.**

I. Introduction

Gesture recognition is defined as interpreting human gestures using mathematical algorithms. Gestures can originate from any bodily motion or state [5]. Current focuses in the field include emotion recognition from the face, hand gesture recognition or even body gestures. Many approaches have been made using cameras and computer vision algorithms to interpret sign language. However, the identification and recognition of human behaviors is also the subject of gesture recognition techniques.

One of the important benefits of gesture recognition is that it can provide the help and assistance for people with disabilities such as the deaf or any one who can not use his voice to communicate or make gestures using his/her body. If we can provide them a way to communicate with other people by interpreting their sign language into spoken words, or even those who do not know sing language, gestures will be very important.

Due to the availability and low cost of the 3D cameras nowadays in the market, we have focused on the human body motion and gestures. This type of recognition collects information about the human body and translates motion into meaningful words humans can understand and work with.

Generally, if we can recognize and translate the body motions collected by any type of sensor as an input for a computer to useful and meaningful words, it will affect and improve the use of technology by making it more easy and usable for many people who have physical limitation.

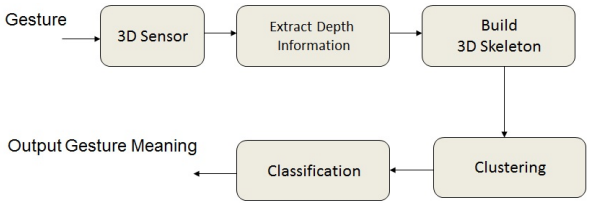

Fig. 1: General block diagram for the system

The use of gesture recognition as an input device for the computer may eliminate the need of some other ordinary input devices such as keyboard, joystick or mouse specially when there is a physical difficulty the user may face. It can be used for many applications and perform in a more speedy and accurate manner.

Gesture recognition may be used in many applications such as communication strategy. This can be used by the computer for gestures to recognize, understand, actions to perform and even words to translate and speak. This type of gestures may focus on a whole body motion or facial expressions or even hand movement. Many types of sensors may be used to apply the communication types for this purpose.

In this work, we need the following phases to make the system work (see Figure 1 :

1) Data collection using KINECT as a 3D imaging device.
2) Data processing and normalization.
3) Transforming the processed information into 3D skeleton.
4) Understanding the gesture in the skeleton using machine learning approach.
5) Take in consideration the context of this gesture.
6) Translate the gesture meaning to the user.

With data processing stage some studies use the hidden markove model (HMM) like [3], [6], [7] who used both HMM and neural networks. [1] uses fuzzy and neural network algorithms and uses nearest neighbor and furthers neighbor classifiers to predict 3D positions of body joints from single depth image and no temporal data.

Kendons gesture continuum suggests that as the quantity of information transmitted by the human voice decreases [2]. The amount of information conveyed by gesturing directly increases. In some cases, they are the only communication

Fig. 2: 2D intensity image from KINECT

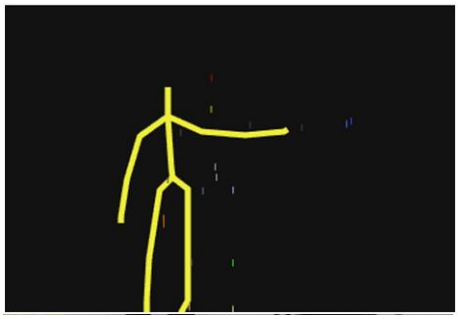

Fig. 3: Detected 3D skeleton

method available (e.g. - some disabled persons). The approache to perfecting gesture recognition vary. Data collection methods include 3D camera images, data gloves, and stereo cameras.

Some researchers used 3D camera image to gain better recognition as [4]. Others try to use different data collection technique like [1] who use Pair of data gloves and [3], [6], [7] use stereo cameras for data acquisition.

Section 2 presents our work and how data was acquired, collected and processed to reach the recognition of gestures. Section 3 shows the results we archived in our experiments. We have list the results we obtained from using the Nearest Neighbor algorithm, Self Organizing Map and Hidden Markov Model algorithms. Finally in section 4, we summarize our research and mention and suggest future work which could be done to increase the knowledge in this field of study.

II. Gesture Recognition using KINECT

The data acquired from the KINECT sensor contains three streams; 2D intensity image (figure 2), depth Image (figure **??**) and sound stream, in addition to the skeleton joints (figure 3), which we need for the feature extraction phase. The 20 skeleton joints are presented as the 3D special point (X, Y, Z), where X = horizontal axis, Y= vertical axis, Z = distance between joint and the sensor (depth).

A. Scale and transition invariant:

We have made some data enhancement and normalization for each joint. This data was collected to make joints invariant for user position from the KINECT, starting from the Hip joints and making it the reference joint. We have also made two adjustments, scaling and translation.

All joints X and Y values centered in the original with reference to the reference joint position by subtracting its position from the Hip reference joint position, and after that, scaling is done by multiplying each joint X and Y positions with the Z joint position. So, no matter how far the user is from the KINECT sensor, it will be invariant with distances between joints. Equation1 and equation 2 are used for this task.

$$x'_i = (x_i z_i) + y_{ref} \tag{1}$$

$$y'_i = (y_i z_i) + y_{ref} \tag{2}$$

Where x' and y' are the new normalized x and y positions after scale and transition, and y_{ref} is the y value of the Hip joint. By obtaining this information, we get a vector V of length 60 values, formed as $V'_1 = X'_1, Y'_1, Z'_1, X'_2, Y'_2, Z'_2, X'_3, \cdots etc$, this vector contains all normalized joints positions referenced with the Hip joint position.

After that, we start recording data for training and testing. The recording process is done by capturing 5 frames for every gesture with interval of 8 frames per second; so we will get 5 frames; 1.29 seconds (frame rate is 31 frames per second), in each frame we will have V vector describing the joints positions. Based to previous experiments we find that the difference in motion between frames with the high frame rate will not be large, because it is based on the human body movement and joint positions.

In our test we have capture 11 different gestures (see figure **??**), each was captured 10 times. For the 11 captured gesture we will have 2200 feature vector to be used for training and testing. Every gesture has been recorded 10 times, each time 5 frames are captured, and each frame forms a vector of 60 values.

As a result we will have 5 sequenced vectors $S = (V_1, V_2, V_3, V_4, V_5)$ denoted by S, which describes the gesture, or movement done by the user.

In this phase we took each captured frame joints and applied the previously described steps to get the frame cluster value. Every 5 clusters are passed to the whole 11 Hidden Markov Models and each will provide a value of likelihood. This shows how much this sequence belongs to this Hidden Markov Model.

We stored the 2200 vector into a file, and by using Matlab, we have used different machine learning techniques to build the gesture recognition phase. These techniques are : Nearest Neighbor (NN), Self Organizing Map (SOM) and Hidden Markov Model (HMM) with K-Mean clustering.

B. Gesture recognition using Self Organizing Map (SOM):

We have used the Self Organizing Map clustering technique to make gesture recognition, we provide it with the 11 gestures

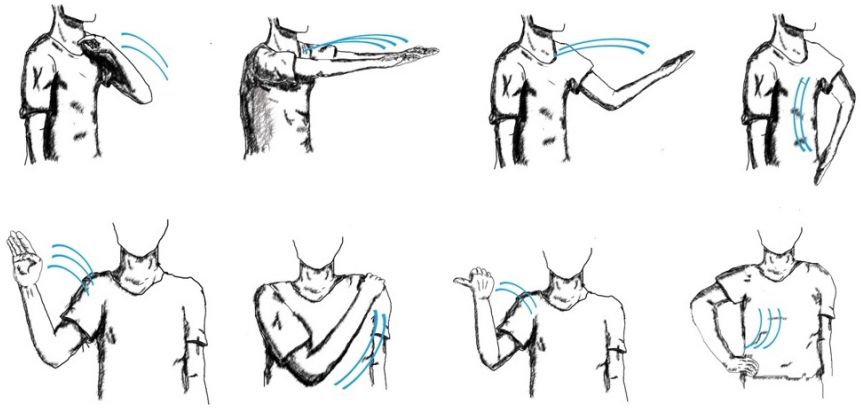

Fig. 4: Sample gestures

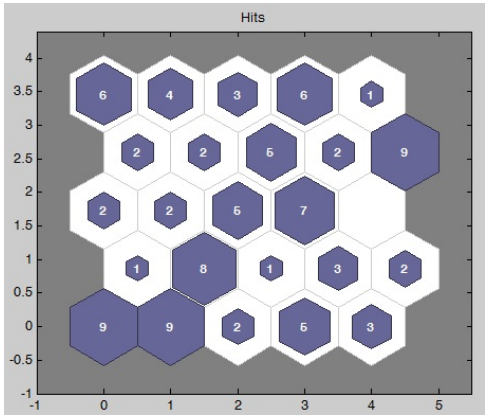

Fig. 5: SOM Clustering

vectors, after the data collection and normalization phase, the feature vectors passed to the SOM to be clustered and classified in order to output the gesture meaning or class. For validation we have used the KFold with $K = 10$.

1) Data preparing:: The normalized data which was collected by the KINECT sensor was prepared to be used for the Self Organizing Map (SOM). Each frame represented by 60 values contains the 20 joints positions in the 3D space. Furthermore, each five frames where reshaped together in sequential order to present the feature vector for the SOM with length 1200 value. We have split the data into two sets: training set and testing set.

2) Training phase:: Two dimensional SOM was created and trained for this training data so each vector represents one gesture of five sequenced frames. Figure 5 shows the clustering results and how the gestures were clustered. We can notice that some clusters has more than one gesture, nearby gestures and far gestures positions in the map.

The decision of which cluster contains which gesture is based on majority function, we have applied a search technique with voting and counting clusters gesture, so the cluster is labeled based on the majority of gestures classified to belong to that cluster.

3) Testing Phase:: After the training process, and using the KFold technique we have classified the testing set using the SOM. Results in details is presented in the next section.

Algorithm:

Algorithm 1 GR using SOM algorithm

KFold with $K = 10$, to split the data into training set and testing set.
Initialise S_{train} and S_{test} sequences for both training and testing sets, where $S = (f_1, f_2, f_3, f_4, f_5)$, ,$f$:frame contains 20 points.
Label each sequence S_{train} with its gesture label L_j.
Train the SOM on S_{train}.
For each sequence in S_{test}, find S_{testi} cell index C_j.
Find C_j majority gestures labels L_j and label S_{testi} with L_j.

C. Gesture recognition using Nearest Neighbor (NN):

To apply the Nearest Neighbor algorithm for predicting the class and recognizing gestures we have prepared the data to be entered to the algorithm and tested the predicted gestures labels. For validation we have used the KFold with $K = 3$.

1) Data preparing: We have used the Nearest Neighbor algorithm to build gesture recognition process. We provide it with 11 gestures vectors. After the data collection and normalization phase, the feature vectors passed to the nearest neighbor algorithm to be classified.

2) Apply Nearest neighbor algorithm: Various types and methods for distances can be applied in the nearest neighbor algorithm. We have noticed that best results produced from the Spearman and Correlation distance calculation methods with accuracy up to 90 percent, some methods where not suitable and produced less than 40 percent of accuracy.

D. Gesture recognition using hidden markov model (HMM):

We have used the Hidden Markov Model and built 11 models for each gesture.

Algorithm 2 GR using NN algorithm

Split the data into training set and testing set.
Initialize S_{train} and S_{test} sequences for both training and testing sets, where $S = (f_1, f_2, f_3, f_4, f_5)$ where f:frame, each has 20 points.
Label each sequence S_{train} with its gesture label.
For each sequence in S_{test}, find S_{testi} cell index C_j.
Find distance between S_i and S_train points.
Find minimum distance for S_j and label S_{testi} with L_j.

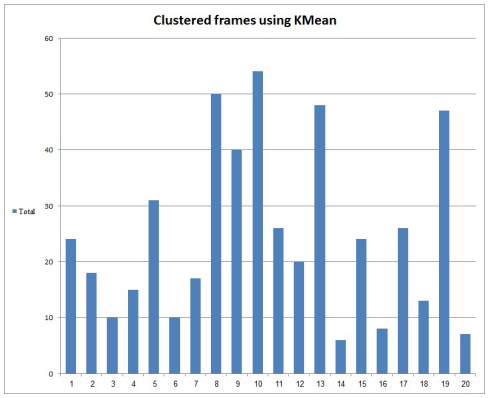

Fig. 6: Frames distribution over KMean clusters

1) Data Preparing:: In order to use the HMM, we first use a clustering technique to reduce the dimensionality of the feature vector. We have tried to use the KMean and the Fuzzy CMean clustering algorithms. The number of clusters were selected experimentally.

Each frame represented by 60 values contains the 20 joints positions were passed to the KMean clustering algorithm to find the frames cluster as a value of 1-20 based on the number of clusters we have. Figure 6 shows the distribution of the frames over the K-Mean clusters. Each five sequenced frames provide one vector for the HMM. The data was split randomly to training and testing sets and we must mention here that the data which was used for clustering is the training set and doesn't included the testing set.

2) Training Phase:: We have created 11 Hidden Markov Models. Each model to be trained for one gesture. Each training gesture data were inputted for its model as a sequence of five values. Each value represent the frame cluster. The HMM emission and transition values were initiated randomly. We have used the Baum-Welch algorithm with tolerance 1e-6. As a result, we had 11 trained hidden markov models, each has emotion and transition values to be used in the featured testing phase.ve.

a) Vectors reading and clustering:: In this phase we use the KFold for splitting the data into two sets; testing set and training set. Then we pass all V of the training set to KMean clustering and compare it with Fuzzy CMean results for number of clusters from 1 to 30. The result was that KMean clustering is diverge when $K = 20$.

We select K = 20, in order to get every vector presented with a value from 1 to 20, depending on the class it belongs to, and store that class centers in order to use it for a new data and the testing set. The S vector will now be presented as 5 values. So, the representation of the vector is reduced from 60 values to one single value between 1 and 20.

For example: $S = (1, 4, 12, 3, 2)$

After that we store all center values generated by the K-Mean algorithm to be used to find new and test data clusters.

Next we start the HMM process by building 11 models. Each model will lead to one single gesture. All training S, which belongs to each HMM, are passed to be trained on.

After the learning process for the hidden Markov models using Matlab, we were able to save the output Emission and Transition matrices to be loaded into an online system, which rebuilds the hidden Markov models depending on the values emission, transition and labels as they were exported from Matlab models.

3) Testing Phase:: After the training process and using the K-Fold technique we have clustered the testing set using the HMM. Algorithm:

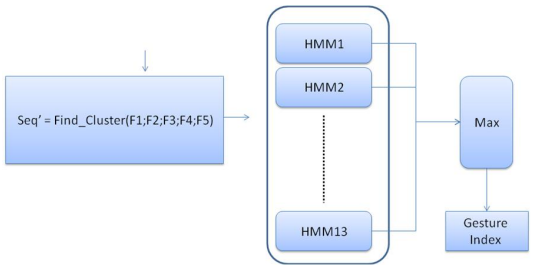

Fig. 7: Proposed hidden markov models

Algorithm 3 GR Using HMM algorithm

1: KFold with $K = 10$, to split the data into training set and testing set.
2: Initialize Raw_{train} and Raw_{test} sequences for both training and testing sets.
3: Apply KMean Clustering on Raw_{train} to produce S_{train}.
4: Find S_test by assign each value to the closest cluster center from the KMean.
5: Reshape $S = (f_1, f_2, f_3, f_4, f_5)$, where$f$ is frame, such that each feature vector S contains one frame cluster value.
6: Label each sequence S_{train} with its gesture label L.
7: Train each HMM on its gestures from S_{train}.
8: For each sequence in S_{test}, find Maximum likelihood from HMM's.

E. Incremental approach

Our approach is based on learning each HMM for one gesture, sampled from a sequence of frames. This approach is incremental because when we need to add a gesture for the system to be identified, we need to add a new HMM and pass the sequence of that gesture. This will not affect previous learned models or the clustering phase because it will have

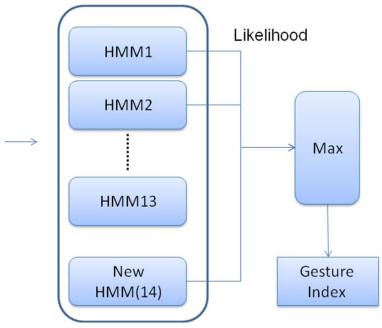

Fig. 8: Incremental approach

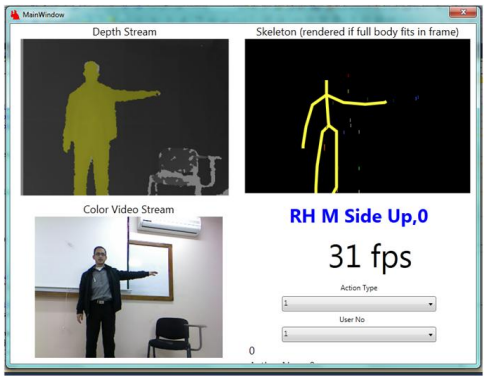

Fig. 9: Final system screen

different sequences to be identified with. Figure 8 shows the process of adding new trained model to the system.

To apply new gesture, we need to provide the system with new sample sequence by recording frames in the same way described previously. Then we find the collected data sequences by finding each frame cluster from the cluster centers we have. After this process we add the new HMM to the system and learn this new model with new sequences to identify.

F. Online gesture recognition using HMM:

In this phase we build a complete system that translate gestures captured by the KINECT sensor and outputs the gesture meaning. The system use the output of the hidden markov model emotion and transmission matrices. It also use the clusters centers generated by the K-Mean clustering from the Matlab to find the new captured frame clusters. The application was built using Microsoft Visual Studio 2010 and Microsoft KINECT SDK. The programming language is C sharp.

- Building the models: In this phase, the system reads the trained models matrices and builds a model for each gesture. These models are already trained and ready to be used.
- Capturing data:
  Captured data from the KINECT sensor is transferred and translated into skeleton joints, each joint represented by a point in a 3D space. For each frame joints we calculate that joint's position with reference to the Hip joint value. Also we apply the scaling and transformation update (as we mentioned before) on each captured joint to form a feature vector of the frame.
  Then each frame is subtracted from the already defined centers generated by the K-Mean to find the distance between the captured frame and all other cluster centers. The $\min$ distance will be the cluster which this frame belongs to.
  After we label the frame, the system adds this frame to an array and when we have 5 labels it forwards the sequence to be recognized by the 11 HMMs and compare the likelihood values each HMM provides. The maximum likelihood determines the gesture.
- Two enhancements:
  First enhancement is defining a threshold ω for sequences to be accepted or rejected. If the maximum likelihood generated from the 11 HMMs is less than ω will be rejected. In other words if the gesture made by the user does not belong to the set of gestures it knows then the HMMs will produce a low value of likelihood and it will be rejected.
  The model with the highest likelihood is considered as the class this sequence belongs to, the difficulty we faced here was finding the start and end points of the sequence because of the continuity of the recognition process. We have solved this issue by overlapping frames between sequences passed to the model because the point we started the sequence in is not deterministic for the user when he behaves in continues recognition mode.
  The overlapping we design is like a queue, with the LIFO (Last In First Out) strategy, each clustered frame is added to the queue, then the sequence of 5 clustered frame values are passed to the Hidden Markov Models to figure out which model provides us with maximum likelihood. As an example, Figure 9 presents a screenshot of the final system. We can notice the RGB image, depth image, human skeleton and the result generated for the current gesture which is: Right hand moving side up.

III. Results

These results describes the progress we archived in gesture recognition using KINECT sensor. First we will mention the results of the three algorithms we discussed earlier; Self Organizing Map, Nearest Neighbor and Hidden Markov model. We also will justify our use for some parameters like HMM number of states and the number of clusters K for the KMean algorithm.

A. Gesture Recognition using Self Organizing Map(SOM) algorithm

Listed below the results for using SOM algorithm, we have apply the SOM using $n \times n$ dimensionality, we have used cross validation with $K = 10$:

We can notice form the table that best results reached when we used 7×7 SOM with accuracy of 75 percent. we can justify

TABLE I: Results using SOM

K	1	2	3	4	5	6	7	8	9	10	Avg
1	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0
3	3	7	11	12	17	21	24	27	31	33	18.6
4	35	38	38	39	43	43	46	49	51	54	43.6
5	56	56	59	63	63	67	67	67	68	68	63.4
6	69	69	69	68	67	67	69	71	71	73	69.3
7	74	74	74	75	77	78	77	76	75	74	75.4
8	75	77	76	74	74	73	73	73	74	74	74.3
9	71	69	67	68	66	67	68	68	67	67	67.8
10	69	71	72	72	72	70	69	69	69	69	70.2

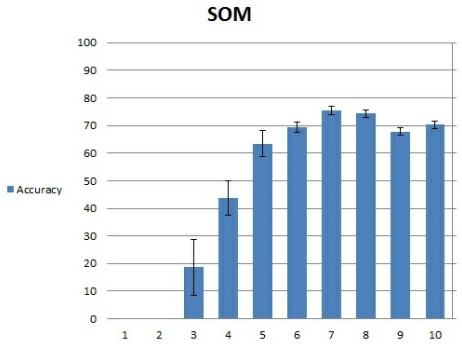

Fig. 10: Self Organizing Map algorithm accuracy using different dimentions

the reason of this result because when the dimensionality of the SOM is very low, multiple gestures with different classes will be placed in the same class due to the similarity between them. On the other hand, if the number of clusters is large it will cluster the gestures into small classes where we need them to be grouped based on their type. This number is so related to the number of gestures we use.

B. Nearest Neighbor algorithm accuracy using different distance methods

The results we obtain from HMM algorithm was better than we obtain from SOM, below the results from using NN algorithm, we have apply the NN using different distance methods, we used cross validation with $K = 10$:

TABLE II: Nearest Neighbor results comparison

Distance method	Average accuracy
Euclidean	73.33
Seuclidean	73.23
Cityblock	61.91
Minkowski	73.33
Chebychev	78.48
Mahalanobis	80.90
Cosine	87.37
Correlation	90.30
Spearman	90.90
Hamming	38.58
Jaccard	38.58

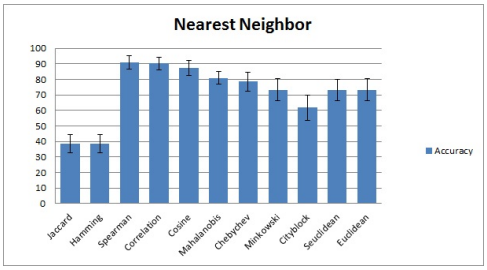

Fig. 11: Nearest Neighbor algorithm accuracy using different distance methods

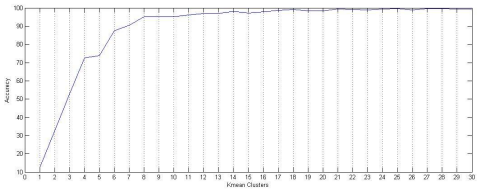

Fig. 12: K-Mean clusters

IV. Gesture Recognition using Hidden Markov Model(HMM) algorithm

The results we obtain from HMM algorithm was better than we obtain from NN and SOM, below the results from using HMM algorithm, we have apply the HMM multiple times with different number of states, we used cross validation with $K = 10$:

The first step in this phase was applying KMean clustering for each frame vector with 20 clusters. That provides a sequence of 5 frames for each gesture. Each gesture was recorded 10 times and values passed to the HMM related to that gesture. Using cross validation with $K = 10$ we have archived an accuracy of 98 Percent.

We need to cluster the frames before the HMM training phase, we have experimentally choose $K = 20$ based on the results we notice in Figure 12.

A. Time analysis

The time experiments shows that the gestures are recognized in real-time. We have implement the three algorithms mentioned before which are NN, SOM and HMM. Table tblno shows the time results. The computation overhead for using the recognition algorithms on Intel(R) Core(TM)2 Duo CPU E7200 with 2.53GHz:

TABLE III: Results for 11 gestures using HMM

2	100.00	90.90	72.72	100.00	90.90	100.00	100.00
3	100.00	100.00	81.81	90.90	81.81	100.00	90.90
4	100.00	63.63	72.72	100.00	90.90	100.00	63.63
5	90.90	90.90	100.00	90.90	90.90	90.90	81.81
6	90.90	100.00	81.81	81.81	81.81	100.00	90.90
7	81.81	90.90	100.00	81.81	90.90	100.00	81.81
8	81.81	100.00	90.90	90.90	81.81	100.00	90.90
9	81.81	100.00	100.00	81.81	90.90	90.90	81.81
10	90.90	90.90	100.00	90.90	90.90	90.90	100.00

Two states HMM: 1.67 milliseconds. SOM with dimensionality 7×7: 8 milliseconds. NN using Spearman distance: 9.21 milliseconds.

This computation overhead will be added to the frame rate of the sensor (28 to 31) frame per seconds which keeps our work in a real-time manner.

V. Conclusion

We have presented a gesture recognition system using different machine learning algorithms for the KINECT sensor.

First we capture data from the KINECT sensor which was translated into human body skeleton. Then we normalize the joints positions based on reference selected joint by applying transformation and rotation over each recorded joint .The normalization process makes the algorithm invariant for changing in position with reference to the sensor position.

The algorithms we have used were Nearest Neighbor(NN) Algorithm, Self Organizing Map(SOM) and Hidden Markov Model(HMM). Through our experiments we have noticed that the HMM algorithm archives best recognition for gestures with an average of 95 percent accuracy. We have used the K-Mean algorithm for data clustering before providing the data to the HMM. The output of gesture recognition using HMM is a set of models that can be learned and added to the system to gain better usability of the algorithm.

Finally, we built a recognition system using the HMM algorithm, the final system builds the trained models based on the emission and transition matrices. Each gesture has an already trained model and ready to be used. The model with the highest likelihood is considered as the class this sequence belongs to, we also made a threshold for the sequences to be accepted or rejected based on the likelihoods they provide.

We have used different distance methods for NN algorithm and we obtain about 80 percent using Spearman distance. Other distances did not provide good results. Further more SOM algorithm provide about 68 percent of recognition accuracy using 7×7 dimensions.

References

[1] S. C. Chen, K. H. Chang, C. K. Liang, S. W. Lin, T. H. Huang, M. C. Hsieh, C. H. Yang, C. M. Wu, and C. H. Lo. 3d gesture language recognition system. In *4th Kuala Lumpur International Conference on Biomedical Engineering 2008*, volume 21 of *IFMBE Proceedings*, pages 773–777. Springer Berlin Heidelberg, 2008. 10.1007/978-3-540-69139-6-192.

[2] Adam Kendon. Some relationships between body motion and speech. In Aron Seigman and Benjamin Pope, editors, *Studies in Dyadic Communication*, pages 177–216. Pergamon Press, Elmsford, NY, June 1972.

[3] C. Keskin, A. Erkan, and L. Akarun. Real time hand tracking and 3d gesture recognition for interactive interfaces using hmm. In *In Proceedings of International Conference on Artificial Neural Networks*, 2003.

[4] S. Malassiotis, N. Aifanti, and M. G. Strintzis. A gesture recognition system using 3D data. In *3D Data Processing Visualization and Transmission, 2002. Proceedings. First International Symposium on*, pages 190–193, 2002.

[5] Matthias Rehm, Nikolaus Bee, and Elisabeth André. Wave like an egyptian: accelerometer based gesture recognition for culture specific interactions. In *Proceedings of the 22nd British HCI Group Annual Conference on People and Computers: Culture, Creativity, Interaction - Volume 1*, BCS-HCI '08, pages 13–22, Swinton, UK, UK, 2008. British Computer Society.

[6] Andrew D. Wilson, Student Member, Ieee Computer Society, Aaron F. Bobick, and Ieee Computer Society. Parametric hidden markov models for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21:884–900, 1999.

[7] Guangqi Ye, Jason J. Corso, and Gregory D. Hager. Gesture recognition using 3d appearance and motion features. In *Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 10 - Volume 10*, pages 160–, Washington, DC, USA, 2004. IEEE Computer Society.