Sign Language Translator using Microsoft Kinect XBOX 360™

Daniel Martínez Capilla (dani89mc@gmail.com)
Department of Electrical Engineering and Computer Science - Compute Vision Lab
University of Tennessee (Knoxville - USA)
Supervised by: Dr. Hairong Qi and Dr. Fabrice Meriaudeau

***Abstract*— June 2012 - Sign language is the basic alternative communication method between deaf people and several dictionaries of words have been defined to make this communication possible. The goal of the project consists of developing an automatic sign language translator so that a computer will output the corresponding word to a sign executed by a deaf user in front of a camera. Several works have been proposed previously and they mostly make use of probabilistic models such as Hidden Markov Models or Artificial Neural Networks classifiers. In this thesis, the Microsoft Kinect XBOX 360™ is proposed to solve the problem of sign language translation. By using the tracking capability of this RGB-D camera, a meaningful 8-dimensional descriptor for every frame is introduced here. In addition, an efficient Nearest Neighbor DTW and Nearest Group DTW is developed for fast comparison between signs. With the proposed descriptors and classifiers combined with the use of the Microsoft Kinect XBOX 360™, the system has the potential to provide a computationally efficient design without sacrificing the recognition accuracy compared to other similar projects. The project does not focus on a particular official dictionary of signs because the objective consists of evaluation the efficiency of the approach for sign recognition purpose. For a dictionary of 14 homemade signs, the introduced system achieves an accuracy of 95.238%.**

I. INTRODUCTION

Unlike other animals, humans have been endowed by nature with the voice capability that allows them to interact and communicate with each other. Hence, the spoken language becomes one of the main attributes of humanity. Unfortunately, not everybody possesses this capability due to the lack of one sense, i.e. hearing. Sign language is the basic alternative communication method between deaf people and several dictionaries of words or single letters have been defined to make this communication possible.

The goal of the thesis consists of developing an automatic Sign Language Translator using the data provided by the Microsoft Kinect XBOX 360™camera. An input sign done by user is recorded by the camera and after processing the raw image, the translator will provide the correspondent word/letter in the spoken language as output. You can easily understand the goal by seeing Fig 1.

A. Thesis proposal

Several works about Sign Language Translators have been introduced before and Gesture Recognition has always been an active research area. A wide number of authors have tried to find new ways to solve this problem and almost all the time they end up using complex implementations based on statistical descriptors that increase the complexity of computation.

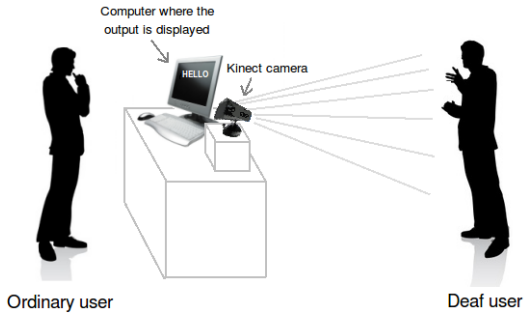


Fig. 1: Goal of the system. A deaf user is making a sign and the system outputs the corresponding word over the screen of the computer so that the ordinary user will understand him

In a project such as this with a time frame of only 3 months, the constraints are an issue. This required the setting up of a suitable and feasible goal of the project from the beginning. The aim of the project is to make the Sign Language Translator work in the simplest possible way and leave it open for future improvements. Starting from a basic implementation and improve it as much as possible until the best possible accuracy of system will be reached.

Sign Language Translation task is highly influenced by its linguistics (see [1] for further information). The syntax and morphology of the Sign Language play a very important role and the order of the words or the non-manual components (i.e. lip movements, facial expression, etc.) can drastically change the meaning of a sign. These facts make the translation process even more complex. The Sign Language Translator will be capable of satisfying the following goals:

- Use data provided by the Microsoft Kinect XBOX 360™camera.
- Recognize a list of basic signs. This list will contain key words such as the ones from Table I. Using these words, the deaf user will be able to transmit what he/she needs and the communication between deaf and ordinary users will become possible (see again Fig 1). Considering the data that the Microsoft Kinect XBOX 360™provides, the signs are homemade rather than belonging to an official sign language because the main goal of this project is to make a system capable of working with a wide number of meaningful words. If the work is focused on a specific official sign language,

the selection of these basic meaningful words will be hard since sometimes the difference between them resides in characteristics that this project is not taking into account (i.e: finger positions, lip movements,etc.).

Dictionary of Signs	
am/are	doctor
have	hello
hot	hungry
I	love
phone	play
question	sick
want	you

TABLE I: Dictionary of default signs of the system.

- Design an interactive user interface so that the user will be able to run the application without any previous knowledge.
- The system must work on real time and give an instantaneous output once the sign is executed.
- Allow the user to auto-train the dictionary (training dataset) by adding new words.

II. BACKGROUND

A. Related work

Several reviews on Human Gesture Recognition have been presented before in [2], [3], and [4]. They mostly utilized 2D information and only a minority of them worked with depth data (3D).

Yang Quan et al. defined in [5] a Basic Sign Language Recognition system that is able to translate a sequence of signs into the commonly used speech language and vice versa. The system was thought to be installed in public places such as airports and hospitals and the dictionary of words contains specific signs that allow the deaf user to transmit what he/she needs. This sign language/speech bidirectional translation (from signs to speech and from speech to signs) focused on the Chinese Manual Alphabet where every single sign belongs to a single letter from the alphabet. Two kinds of data were used: vector of hand gestures and vector of lip actions. In order to characterize these vectors, they used the Normalized Moment of Inertia (NMI)[6] algorithm and Hu moments [7]. As said before, they combined the hand gestures recognition with the lips movement reader in order to make the system more robust. By using a multi-futures SVMs classifier trained with a linear kernel, the 30 letters from the Chinese manual alphabet were recognized with an average accuracy of 95.55%. Starner et al. [8] used a view-based approach with a single camera to extract 2D features as the input of HMM for continuous American SLR. They got a word accuracy of 92% or in recognizing the sentences with 40 different signs. Other projects made use of custom-made gloves, where every finger contained a different color. In [9], Akmeliawati et al. introduced a sign language translation using Colour Segmentation as feature extraction and a Neural Network as a classifier. They either could detect numbers (1-9), letters (A-Z) and up to 11 words (e.g. beautiful, close, driving, he, his, etc). In the case of the numbers and letters, they defined a 10-array vector that contains x and y offsets that belonged to the distance between each finger and the centroid of the hand. For the dictionary of words, they avoided the details (position of the fingers) and they focused only on the tracking of the centroid of the hand. Hence, the sequence that belonged to the position of the centroid at the different frames defined the model of the sign. Finally, three different Neural Networks were used as classifiers. The average accuracy obtained was 96.668% and the specific accuracy for the dictionary of words was 95%. The Center for Accessible Technology in Sign (CATS) is a joint project between the Atlanta Area School for the Deaf and the Georgia Institute of Technology. They developed a system called CopyCat [10] as a practice tool for deaf children to help them to improve their working memory and sign language skills. The system required an ASL phrase verification to enable interaction. The important citation here is the project that they are developing today. They are working on a Kinect-based ASL recognition system. In fact, it was after being in contact with this company when the brake on the project's goal was put. Although they did not provide the details of their implementation, they are using the GT2K gesture recognition toolkit and they also use Hidden Markov Models. They are trying to build a system capable to recognize the whole ASL dictionary. In [11] Jonathan C. Hall also demonstrated how HMM-based gesture recognition was a good solution when dealing with 3D data (i.e. joint coordinates). A physical gesture can be understood as a Markov chain where the true states of the model cannot be directly observed. This type o Markov model is called a Hidden Markov Model (HMM). In order to reduce the real gesture data to a workable number, K-means was used to cluster the coordinates of every sign. In [12], Vogler et al. introduced the parallel Hidden Markov Model-based method. They used 3D data as the input of the recognition framework. These data was either collected with 3D computer vision methods or with a magnetic tracking system such as the Ascenion Technologies Motion Star system. They showed how to apply this framework in practice with successful results using a 22-sign-vocabulary. The reported best accuracy is 95.83%.

B. Other considerations

In this project, the Microsoft Kinect XBOX 360^{TM} is used as the main device for the data collection. Although at the beginning Microsoft did not release any drivers to enable the Kinect to be used with a personal computer, its statement was later modified and they said that the USB port used to connect the device to the XBOX was left "intentionally open". Since then, a few Open Source Drivers, SDKs, and APIs have arisen. Between them, OpenNI/NITE was selected since it contains the main functionalities that will allow to track the joint positions used in this project (see [1] for further information and details).

III. METHODOLOGY

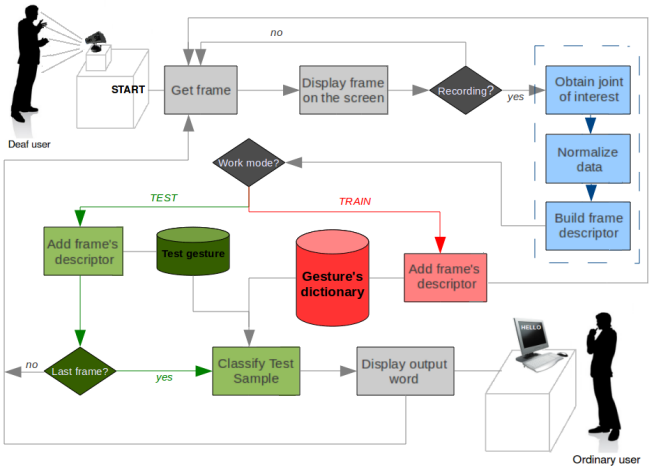


Fig. 2: Flux diagram of the system. Blocks executed when a new frame is captured.

Consider the block diagram from Fig 2 . The deaf user is in front of the camera doing a sign or getting ready to do so. With a frame rate of 20fps, a new frame is obtained and the video stream is updated with the skeleton of the user overlapped onto it. At that point, if the user wants to record a sequence (otherwise, the system asks the camera to get the next frame), three main blocks are executed: the first block consists of obtaining the data of the joints of interest (JoI) required for the frame descriptor, the second block consists of normalizing these data, and the third one consists of building the frame descriptor. Then, if the working mode is set to `TRAINING` (meaning that the user is adding a new sign to the training set), the frame descriptor is added to the correspondent file of the dictionary. Otherwise, if the mode is set to `TESTING` (meaning that the user wants to translate the sign that is been done), the frame descriptor is added to the current test sample. Then, the system checks if the current frame is the last frame of the sign. After a sign is finished and if the working mode is `TESTING`, the test sign is compared using a classifier with the signs from the dictionary and the corresponding output is displayed so that the ordinary user will know the corresponding word in the spoken language. After that, the system keeps going with the next frame and the flow of the block diagram is repeated again.

A. Joints of interest (JoI)

OpenNI/NITE can track up to 15 joint positions. After carefully studying the signs of the proposed default dictionary for the system, only 4 joints out of the 15 resulted to be significant for the description of a sign: both hands and both elbows. There is no point in tracking others joints such as the shoulders, the knees, the feet, etc. because they remain almost static during the execution of the sign. Adding these joints to the sign descriptor will be the same as adding redundant information. Even though the description step can be done using the four previously mentioned joints, some other joints are also required for the normalization and the sign modeling steps. These are the **HEAD** and **TORSO** joints. By doing so, the list of tracked joints at every frame is reduced from 15 to six (see the corresponding position of the joints and the notation that will be used from now on in Fig 3).

Fig. 3: Used joints.

B. Normalization of the data

1) Invariant to the user's position: The normalization must take into account the position of the user. The deaf user can be at different positions of the room and consequently the data must be stored accordingly to that position. As shown in Fig 4, a slight variation in depth can cause a considerable variation of the X and Y values. The distances between one joint and another one can drastically vary depending on the position of the user.

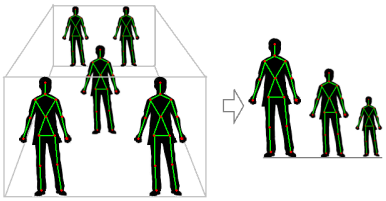
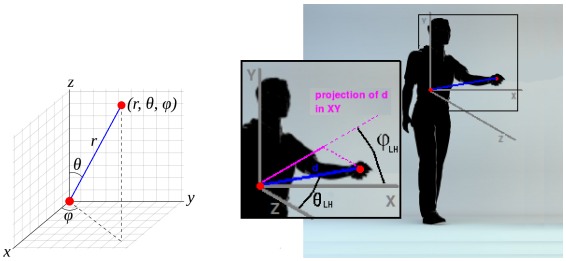


Fig. 4: Normalization required for the position of the user.

Instead of directly storing the Cartesian coordinates X,Y, and Z (which can be obtained using OpenNI/NITE), the proposal consists in normalizing all the joint coordinates with respect to the position of the `TORSO`. This position remains always constant along the sign frames and is the right one to be used to make the system position-invariant. Instead of using the Cartesian coordinates X,Y, and Z, the spherical coordinates considering `TORSO` as the origin are stored.

In mathematics, a spherical coordinate system is a coordinate system for three-dimensional space where the position of a point is specified by three numbers: the radial distance of that point from a fixed origin (r), its polar angle measured from a fixed zenith direction (θ), and the azimuth angle of its orthogonal projection on a reference plane that passes through the origin and is orthogonal to the zenith, measured from a fixed reference direction on that plane (φ). Fig 5(a) shows these three numbers or values and Fig 5(b) shows the correspondence of these three values in the system.

The radial distance r will be expressed by d and defines a vector between the `TORSO` and the correspondent joint. (θ



(a) Definition of (r, θ, φ) as commonly used in physics: radial distance r, polar angle θ, and azimuthal angle φ. [?]

(b) Equivalence of these values in the system. Example case of the LH joint

Fig. 5: Use of the spherical coordinates.

and φ) are the angles that describe the direction of this 3D vector.

Given the set of joints $J = \{LE, RE, LH, RH\}$ and considering T as the TORSO, the set of distances $D = \{d_{LE}, d_{RE}, d_{LH}, d_{RH}\}$, and the sets of orientations $\Theta = \{\theta_{LE}, \theta_{RE}, \theta_{LH}, \theta_{RH}\}$ and $\Phi = \{\varphi_{LE}, \varphi_{RE}, \varphi_{LH}, \varphi_{RH}\}$ are defined as follows:

$$\sum_{i=1}^{n} D(i) = \sqrt{(J(i)_x - T_x)^2 + (J(i)_y - T_y)^2 + (T_z - J(i)_z)^2} \quad (1)$$

$$\sum_{i=1}^{n} \Theta(i) = atan2\left(\sqrt{(J(i)_x - T_x)^2 + (J(i)_y - T_y)^2}\ ,\ (T_z - J(i)_z)\right) \quad (2)$$

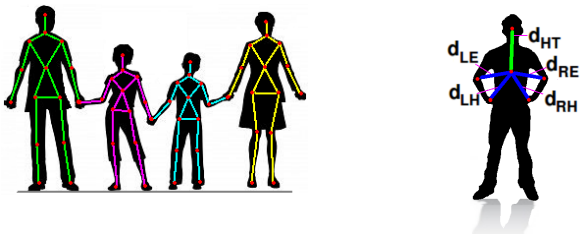
$$\sum_{i=1}^{n} \Phi(i) = atan2\left((J(i)_y - T_y)\ ,\ (J(i)_x - T_x)\right) \quad (3)$$

where n is the number of joints from J.

2) Invariant to user's size: Given a sign, its description must be the same no matter if the user is tall or short and the translator must be able to output the right word in every case. Although the way that the dictionary is built allows it to have several samples for the same sign (meaning that we can have the same sign described for different user's sizes), there is no way to add the samples for all the possible user's sizes to the dictionary. Otherwise the classification process will become slower and less accurate.

The user's size problem is shown in Fig 6 (a). The distance from one joint to another changes significantly depending on the user's size (the distances for the users in the middle are smaller than the distances for the users at the sides).

After the normalization of the user's position, every joint is expressed by its relative distance d to the TORSO joint and the two angles θ and φ that describe the orientation of this distance. The proposal shown in Fig 6(b) consists of normalizing all the relative distances d by the factor that is defined by the distance between the HEAD and the TORSO joints (d_{HT}). This factor tells about the size of the user and all the distances D can be normalized accordingly to this value.



(a) Different user sizes

(b) Set of distances D

Fig. 6: Normalization required for the user sizes.

Given the set of distances $D = \{d_{LE}, d_{RE}, d_{LH}, d_{RH}\}$, the normalized set of distances D_{norm} is obtained as follows:

$$\sum_{i=1}^{n} D_{norm}(i) = \frac{D(i)}{d_{HT}} \quad (4)$$

where n is the number of distances from D and d_{HT} is the HEAD–TORSO distance (the green segment from image image 6(b)). There is no need to normalize the angles θ and φ since they are expressing the direction and the direction remains the same after the normalization.

C. Sign descriptor

Once the JoI data are obtained and normalized, the next step is building a descriptor for each sign. The descriptor must be able to describe a sign in a way that this descriptor will be unique and sufficiently different from the other descriptors of the dictionary. After the first evaluation of the system, the results showed that the feature θ does not provide any meaningful information. That is why the final 8-dimensional descriptor contains for every frame, the spherical coordinates d and φ for each of the four joints (see Fig 7.

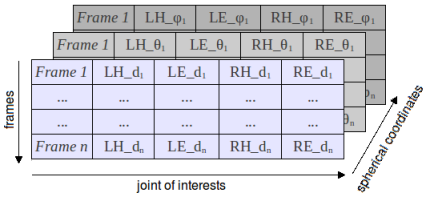


Fig. 7: Sign descriptor based on the spherical coordinates values for every joint.

D. Classifier

The classifier is the function that will output the corresponding word of the spoken language once the deaf user inputs a sign. Given an input sequence of frames, the classifier will match it with the closest sequence of frames (sign) from the default dictionary. The problem here is that the two compared sequence do not share the same lenght (even the same sign always contains more frames because of the velocity at which the user execute it). Two different classifiers are developed.

1) NG-DTW classifier: The first proposal is named as Nearest-Group classifier with the Dynamic Time Warping (DTW) algorithm as a cost function. It is a modified version of the well-known Nearest Neighbor classifier with the Dynamic Time Warping (DTW) algorithm as a cost function (see III-D.3). Given a sample test, it is matched with most similar group of signs samples from the dictionary. The most similar group is the one with the smallest mean similarity coefficient after averaging the DTW distances of the samples that belong to a same group. Fig 8 shows the idea of this classifier. In that case, the DTW similarity coefficients are carried out for a given test. Then, the mean value for every group is found. As can be seen, the average similarity for the group "doctor" is lower than the others and that is why the test sample is matched with the class "doctor".

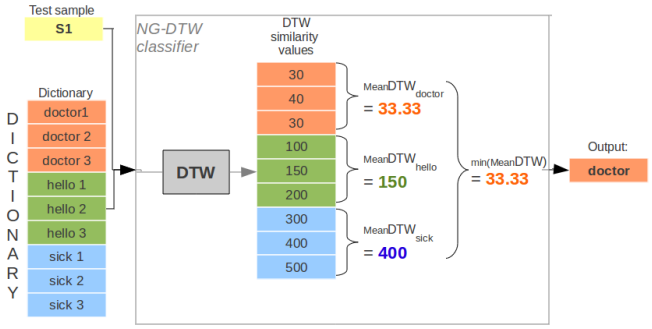


Fig. 8: NG-DTW Classification example for a given test sign. The input sign is classfied as "doctor" because it is the group that contains the smallest mean similarity coefficient.

2) NN-DTW classifier: The second proposal is a modified version of the first one, but instead of matching the test sign with most similar group of signs samples from the dictionary, the test is matched with the most similar single sign sample from the dictionary. In order to find the similarity between the test sign and each of the signs from the training set, the DTW algorithm is used.

3) Dynamic Time Warping algorithm (DTW): Dynamic time warping (DTW) was introduced in 60s and it is an algorithm for measuring similarity between two sequences which may vary in time or speed. For instance, similarities in walking patterns would be detected, even if in one video the person is walking slowly and if in another video he or she is walking more quickly, or even if there are accelerations and decelerations during the course of one observation. By using DTW, a computer will be able to find an optimal match between two given sequences (i.e. signs) with certain restrictions. The sequences are "warped" non-linearly in the time dimension to determine a measure of their similarity independent of certain non-linear variations in the time dimension. In this project, DTW is satisfactory used for gesture/sign recognition purposes, coping in that way with sign executions speeds.

IV. EXPERIMENTS AND RESULTS

In this section, the accuracy of the system for the different implemented approaches and configuration of parameters is analyzed. The default training set contains a total of 70 different samples, which is the result after adding five different samples for each of the 14 signs from the dictionary of words listed in Table I. All these training samples belongs to the same user and are executed at the same position. In order to test the system, a set of test samples is collected. This set contains signs done by four different users that differ in size which are not the same than the one from the dictionary. For every user, three different samples for each sign are added to the set of test samples. This results in a total of 168 testing samples that will be used to find the accuracy of the system.

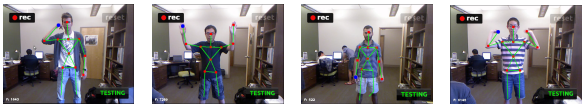


Fig. 9: Different users used to test the system.

Three different approaches are evaluated: (1) *Cartesian + NN-DTW:* The descriptor contains the Cartesian coordinates (X,Y, and Z) of the four used joints and where the user's size normalization is not taken into account (only position normalization is considered). The classifier used is the Nearest Neighbor-DTW. (2) *Spherical + NN-DTW:* The descriptor contains the spherical coordinates of the joints. The user's size and position normalization are taken into account here. The classifier used is also the Nearest Neighbor-DTW. (3) *Spherical + NG-DTW:* The difference of this approach with respect to the second one resides in the classifier. Instead of using the Nearest Neighbor-DTW, the Nearest Group-DTW is used.

Different configurations are also evaluated (i.e. which is the most meaningful features and which is the combination of weights for every joint (being H=`HANDS`, E=`ELBOWS`) that provides the best accuracy of the system). See Table II.

A. Discussions

From Table II, several conclusions can be obtained. There is an important variation between the Cartesian approach and the Spherical ones. Considering that in the Cartesian only the normalization of the user's position is taken into account, the significant difference in accuracy between both approaches is showing the efficiency of the implemented normalization for the user's size. Regarding the weight applied to each joint, the `HANDS` seems to have more importance than the `ELBOWS`. The best accuracies are reached when the `HANDS` have and 80% of weight and the `ELBOWS` a 20%. The reason for this is because the `HANDS` remain more separated with respect to the `TORSO` than the `ELBOWS` during the execution of the sign and consequently are the joints that contain the coordinates that vary more. The last conclusion if about the most meaningful features, which results to be d and φ.

	Cartesian + NN-DTW			Spherical + NN-DTW			Spherical + NG-DTW		
	H=0.8, E=0.2	H=0.5, E=0.5	H=0.2, E=0.8	H=0.8, E=0.2	H=0.5, E=0.5	H=0.2, E=0.8	H=0.8, E=0.2	H=0.5, E=0.5	H=0.2, E=0.8
x / d	77.381% 77.421%	80.350% 80.396%	77.381% 77.381%	73.810% 73.8095%	78.5714% 78.5714%	77.381% 77.500%	71.429% 71.429%	75.000% 75.000%	72.619% 72.7381%
y / θ	4.762% 4.762%	7.143% 7.024%	8.330% 8.214%	5.357% 5.2381%	8.928% 9.008%	10.714% 10.794%	4.762% 4.643%	5.952% 6.032%	8.330% 8.413%
z / φ	71.429% 71.429%	70.833% 70.952%	68.452% 68.810%	94.048% 94.405%	91.660% 92.143%	88.690% 88.166%	91.071% 91.4286%	87.500% 87.980%	82.143% 82.739%
x,y / d, θ	58.928% 57.857%	72.619% 72.5794%	75.5952% 75.754%	57.143% 57.143%	59.524% 59.524%	51.191% 51.310%	64.286% 64.286%	58.929% 58.929%	44.643% 44.881%
x,z / d, φ	85.119% 85.119%	80.357% 80.357%	74.405% 74.524%	**95.238%** **95.238%**	93.452% 93.452%	86.905% 86.905%	92.262% 92.262%	91.071% 91.071%	83.929% 83.929%
y,z / θ, φ	71.429% 71.429%	70.833% 70.833%	69.048% 69.405%	75.595% 75.952%	70.238% 70.516%	60.714% 61.071%	70.238% 70.595%	66.670% 67.024%	54.762% 55.952%
x,y,z / d, θ, φ	85.119% 85.119%	82.738% 82.738%	75% 75.119%	**94.643%** **94.643%**	91.660% 91.660%	80.952% 81.071%	**94.643%** **94.643%**	91.666% 91.666%	80.952% 81.071%

TABLE II: System accuracies for the different configurations. The left column tells about which feature is used. Consider x,y,z for the Cartesian approach and d, θ, φ for the Spherical approach. In the first row, the title indicates the evaluated approach and the second row expresses the applied weight to each one of the joints (being H=hands and E=elbows). The top value from each cell refers to the Accuracy 1 (positives/total) and the value from the bottom refers to the Accuracy 2 (accumulation of the single accuracies by signs / number of signs).

After evaluating the behavior of θ along the frames and for different signs, it was thought that this angle has a similar behavior always and it is not meaningful to describe a sign. This is the reason why the final descriptor only considers the features d and φ.

By its side, the differences between the accuracies when using the Nearest Neighbor-DTW classifier and the Nearest Group-DTW classifier do not seem to be that important, although the former unexpectedly performs better than the second one (95.238% and 94.643% respectively). Indeed, the second approach was intended to make the classification process more robust, but the results are showing that this guess was not true for the current test samples. In the case of the Nearest Group-DTW classifier, the test sample is matched with the group of signs whose average DTW-distances with the test sample are smaller. This means that if for some unexpected reason one of these DTW-distances is totally different compared with the rest from the same group (e.g. due to some error when collecting the data), the average value will be consequently affected and the classifier will be more prone to misclassification. If the training samples were taken by different users, the Nearest Group-DTW would probably perform better than the Nearest Neighbor-DTW.

Finally, after evaluating the possible configurations and approaches independently, it is time to consider all of them together and see which is the one that gives the best system accuracy. This configuration is:

{ weights:{`HANDS`=0.8, `ELBOWS`=0.2}, used features: $\{d, \varphi\}$, approach:NN-DTW classifier }

This will be the approach used by default by the system since it is the one providing the best results (95.238%). Only 8 test samples out of 168 are misclassified where 6 out of these 8 belong to the same conflictive sign `hungry`. In [1], a more detailed evaluation of the results is presented.

V. CONCLUSIONS AND FUTURE WORK

The system is working since the best configuration achieves 95.2381% accuracy. By combining the 14 signs from the default dictionary, a total of 12 basic sentences have been defined (the list can be easily increased). These sentences consist of basic sentences such as "I want to see a doctor", "I am sick", "I am hungry", "What is your phone number?", etc. If the system is incorporated in business meetings, hospitals, supermarkets, etc, by using these sentences, the communication between a deaf user and an ordinary user will become possible. Despite the fact that the defined signs do not belong to a specific official Sign Language, the idea of the project was to show that with basic descriptors and classifiers and the use of the Kinect, a wide number of signs could be recognized and the system has the potential to provide a computationally efficient design without sacrificing the recognition accuracy compared to other similar projects.

To make this system work with a real Sign Language (American Sing Language, Spanish Sign Language, etc.), some other features such as the finger position or shape of the hand will have to be considered. The inclusion of a new way to detect the initial frame of a gesture will make the system more automatic. The last future improvement refers to the computational cost. Although the current system works in real time, its computational cost could be improved by reducing the number of dimensions from the descriptors to those that are most meaningful. Principal Component Analysis might me a good solution.

REFERENCES

[1] D.Martinez, "MSc Thesis - Sign Language Translator using Microsoft Kinect XBOX 360™" ,*VIBOT 5.* , Department of Electrical Engineering and Computer Science, Computer Vision Lab, University of Tennessee.

[2] Pavlovic, V.I. and Sharma, R. and Huang, T.S., "Vision-Based Gesture Recognition: A Review", *Pattern Analysis and Machine Intelligence, IEEE Transactions on, 1997.*

[3] Wu, Ying and Huang, Thomas S., "Visual interpretation of hand gestures for human-computer interaction: a review", *Proceedings of the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, 1999*.

[4] D. M. Gavrila, "The Visual Analysis of Human Movement: A Survey", *Computer Vision and Image Understanding, 1999*.

[5] Yang Quan and Peng Jinye, "Application of improved sign language recognition and synthesis technology in IB", *Industrial Electronics and Applications, 2008. ICIEA 2008. 3rd IEEE Conference on*.

[6] Dai Jun, "Gesture Recognition Reach Based on High-Order NMI", *Master Dissertation, ShanHai Maritime University, 2004*.

[7] Yang Quan, "Sign Language Letter Recognition Algorithm Based on 7Hu Invariant Moments", *Master Dissertation, Xi'ans University of Architecture and Technology, 2007*.

[8] Thad Starner and Joshua Weaver and Alex Pentland, "Real-Time American Sign Language Recognition Using Desk and Wearable Computer Based Video", *IEEE Transactions on Pattern Analysis and Machine Intelligence, 1998*.

[9] Akmeliawati, R. and Ooi, M.P.-L. and Ye Chow Kuang, "SReal-Time Malaysian Sign Language Translation using Colour Segmentation and Neural Network", *Instrumentation and Measurement Technology Conference Proceedings, 2007. IMTC 2007. IEEE*.

[10] Zafrulla, Z. and Brashear, H. and Hamilton, H. and Starner, T., "A novel approach to American Sign Language (ASL) phrase verification using reversed signing", *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*.

[11] Jonathan C. Hall, "Gesture Recognition With Kinect Using Hidden Markov Models (HMMs)", http://www.creativedistraction.com/demos/gesture-recognition-kinect-with-hidden-markov-models-hmms/.

[12] Christian Vogler and Dimitris Metaxas, "A framework for recognizing the simultaneous aspects of American Sign Language", *Computer Vision and Image Understanding, 2001*.