SINERGI Vol. 22, No.2, June 2018: 91-100
DOAJ:doaj.org/toc/2460-1217
DOI:doi.org/10.22441/sinergi.2018.2.004 .

DESIGNING TRANSLATION TOOL: BETWEEN SIGN LANGUAGE TO SPOKEN TEXT ON KINECT TIME SERIES DATA USING DYNAMIC TIME WARPING

Zico Pratama Putra[1], Mila Desi Anasanti [2], Bagus Priambodo[3]
[1] School of Electronic Engineering and Computer Science, Queen Mary University of London
[2]Imperial College London, London, UK
[3]Information System, Faculty of Computer Science, Universitas Mercu Buana
Jl. Raya Meruya Selatan, Kembangan, Jakarta 11650
Email: z.putra@qmul.ac.uk m.anasanti15@imperial.ac.uk bagus.priambodo@mercubuana.ac.id

Abstract -- *The gesture is one of the most natural and expressive methods for the hearing impaired. Most researchers, however, focus on either static gestures, postures or a small group of dynamic gestures due to the complexity of dynamic gestures. We propose the Kinect Translation Tool to recognize the user's gesture. As a result, the Kinect Translation Tool can be used for bilateral communication with the deaf community. Since real-time detection of a large number of dynamic gestures is taken into account, some efficient algorithms and models are required. The dynamic time warping algorithm is used here to detect and translate the gesture. Kinect Sign Language should translate sign language into written and spoken words. Conversely, people can reply directly with their spoken word, which is converted into literal text together with the animated 3D sign language gestures. The user study, which included several prototypes of the user interface, was carried out with the observation of ten participants who had to gesture and spell the phrases in American Sign Language (ASL). The speech recognition tests for simple phrases have therefore shown good results. The system also recognized the participant's gesture very well during the test. The study suggested that a natural user interface with Microsoft Kinect could be interpreted as a sign language translator for the hearing impaired.*

Keywords: *Human-Computer Interaction: Natural User Interface; Speech Recognition; 3D Animation; Sign Language*

Received: April 13, 2018 *Revised: May 23, 2018* *Accepted: May 24, 2018*

INTRODUCTION

Sign language is a visual communication used as the primary means of communication for the hearing impaired, using hand movements, arms, body and facial expressions to communicate using sign language. Many technical solutions have been proposed and implemented for the translation of sign language into text. Text and speech can also be translated into sign language for easier bilateral communication. Because of the nature of sign language communication, it was difficult for non-doves to understand it. As a result, hearing impaired people have difficulties communicating with the customer service counters, e.g., in train stations, shops, and other public areas (Hore et al., 2017).

Before the release of Kinect Camera, which is capable of creating the depth of images, sign language research focused on hand color processing to preserve the hand shape (Cerezo, 2011). The study used a standard webcam camera that can only capture colors of light and dark skin. This method has a disadvantage when different users need different algorithms due to differences in shape and color of the hands.

Some researchers have overcome the problem of skin color by suggesting colored gloves or marker (Akmeliawati, Ooi, & Kuang, 2007; Buchmann, Violich, Billinghurst, & Cockburn, 2004; Kyatanavar & Futane, 2012; Uebersax, Gall, Van Den Bergh, & Van Gool, 2011). This method was unfortunately not very appealing. Other weaknesses are the algorithms used in this method, which focus only on the palm of the user's hand, while in the real situation the sign language is not only dependent on the movement of the palms. In some cases, sign language even refers to hand and head gestures.

The latest advanced technologies of Kinect enable developers and researchers to overcome previous limitations. With the ability to create a depth image, the Kinect allows the use of sign language more naturally. Kinect Auslan offers a range of software modules for the development of applications that perform the recognition of Australian Sign Language with Microsoft Kinect (Auslan, 2012).

Wassner creates a gesture recognition program using Kinect with NN (Neural Network). FANN (Fast Artificial Neural Network) library is used for the recognition system (Wassner, 2011). At the moment, only two words LSF (French Sign Language) are recognized – "hello" and "sorry." They recorded a series of gestures that used to train the signs, i.e., to teach the program to recognize gestures. Researchers at the College of Computing, Georgia Institute of Technology come with American Sign Language recognition for educational games for deaf children. Using the Hidden Markov Model algorithm, they collected 1000 American Sign Language (ASL) phrases for the systems (Zafrulla, Brashear, Starner, Hamilton, & Presti, 2011). There is also research which used multiple regression (Priambodo & Ahmad, 2017). Adriansyah analyzed the goal-seeking behaviors based on Particle Swarm Fuzzy Controller (2015), Fitrianah et al., explored feature extraction in prediction (2015), and Aswari and Diana (2016) used Bezier Curve Method.

Hazari (2017) has studied the use of hand gestures on Kinect devices and suggested some guidelines to make sign language more accessible. Shanableh (2017) has developed an Android mobile app for real-time bilateral Arabic sign language translation that utilizes simple methods for feature extraction. Further work is being done to translate American sign language using a cross-correlation coefficient (Joshi, Sierra, & Arzuaga, 2017).

The Microsoft Kinect is a remote device for the Xbox 360 game console that allows hands-free control of the game console via image processing. The system includes an RGB camera, an infrared projector and a camera for depth perception, a series of microphones for voice commands and a motor for setting the sensor position.

Microsoft Kinect's allows researchers to access the previously complicated and expensive gesture recognition technology with a single tool.

In this study, the software was developed for the Kinect Software Development Kit (SDK) 1.7 Framework. Understanding the interaction of the individual layers of the Kinect SDK is essential. On the lowest level, the Kinect SDK delivers the drivers required to generate the image and audio data from the hardware devices. The abstract layer of Kinect Sign Language is shown in Figure 1.

The Kinect drivers installed as part of the SDK control the streaming of audio and video (color, depth, and skeleton) from Kinect sensors. Kinect processes the audio and video components for skeleton tracking, audio, color and depth imaging.

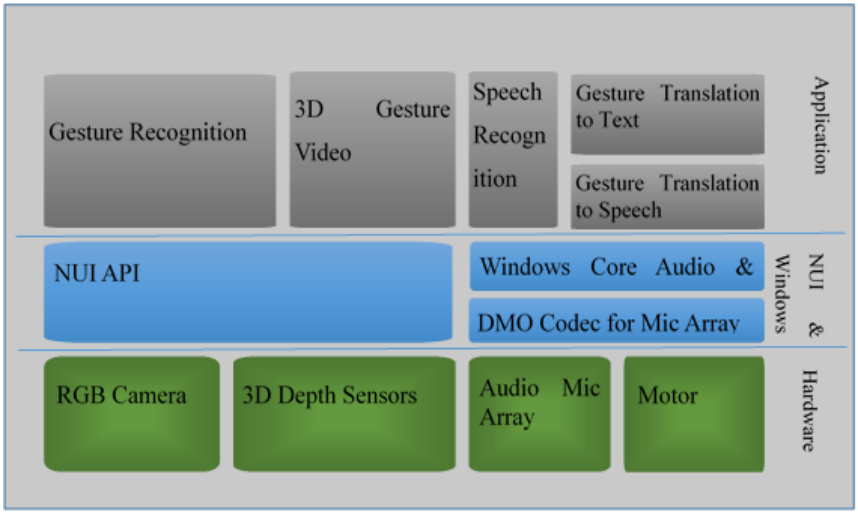

Figure 1. Overview of Kinect Sign Language software layer

The software runs on top of the Kinect SDK framework to extract user gesture information from NUI (Microsoft.Kinect.dll) and provide a comparison method for gesture recognition. The predefined gesture data was delivered through gesture recognition, which can be accessed directly. Gesture translation accesses the audio, speech and media programming interfaces (APIs) from Windows 7 (Microsoft.speech.dll) to process the information obtained from the gesture data. Such data is processed as information that needs to be translated into text and language.

It tweaks the user's voice by accessing the Windows Audio API and translating it into text. We used the results from the text data to execute the 3D animation in the desired language.

The objective of this research is to build intelligent devices that can make a gesture from the first-hand movement to the end position. While the Kinect SDK can track gestures, it cannot record and save them for further processing. Accordingly, we implement the Dynamic Time Warping (DTW) algorithm, which can log and learn targeted gestures for next grouping in some databases depicting sign language. The Dynamic Time Warping (DTW) algorithm was first introduced in the 1960s by Bellman (1959) and extensively researched in the 1970s by Myers (1980) for speech recognition applications. According to Sakoe and Chiba (1978), dynamic time warping is a method for calculating the similarity between two-time series, which can vary in time and speed.

DTW is used in many areas, including handwriting and online signature matching Tappert et al. (1990), computer vision and computer animation by Müller (2007), protein sequence alignment and chemical engineering (Vial et al., 2009).

METHOD

The DTW algorithm is an example of a classification problem in supervised learning. Classification refers to the prediction of a discrete value output. If classification problems occur, it turns out that the algorithm can have more than two values for the two possible value outputs. As a practical example, we can use three set of sign data and the design should able to distinguish the first, second and the third set. First is the "Happy" sign and second is the "Hello" sign, third is the "Good" sign. However, this would also be a classification problem, as this other discrete set of values matches the output "no sign" or "happy" or "hello" or "good".

For example, the study has gesture data set consist of the 2D axis position of the body for specific phrase gesture. In such a data set, the learning algorithm could throw the straight line through the data to try to separate the phrase from the null phrase. Then the learning algorithm can decide to throw the straight line to distinguish the two or three sets of phrases.

The aim of DTW is to compare two time-dependent series $X = (x_1, x_2, \dots, x_n)$ of length $N \in \mathbb{N}$ and $Y = (y_1, y_2, \dots, y_n)$ of length $M \in \mathbb{N}$. These series may be discrete signals (time series) or feature sequences sampled at equidistant points in time. A feature space denoted by F is set. Then:

$$x_n, y_m \in F \text{ for } n \in [1:N] \text{ and } m \in [1:M] \quad (1)$$

To compare two different features $x, y \in F$, one needs a local cost measure, sometimes also referred to as local distance measure, which is defined to be a function:

$$c: F \times F \to \mathbb{R} \geq 0 \quad (2)$$

Whether x and y are similar to each other, $c(x, y)$ is small (low-cost), and vice versa. The algorithm starts by building the *cost matrix* CÂ^^R$^{N?—M}$ representing all pairwise distances between X and Y (see Figure 2 and Figure 3). In Fig. 3 it could be understood that the Cost matrix of the two real-valued sequences X and Y using the Manhattan distance as local cost measure c. Regions of low-cost are indicated by dark colors and regions of high cost are indicated by light colors.

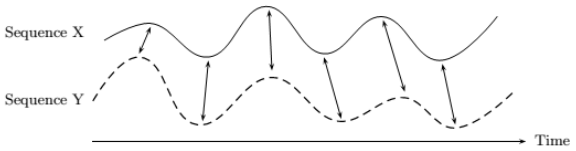

Figure 2. Raw time series, arrows show the desirable points of alignment (Müller, 2007)

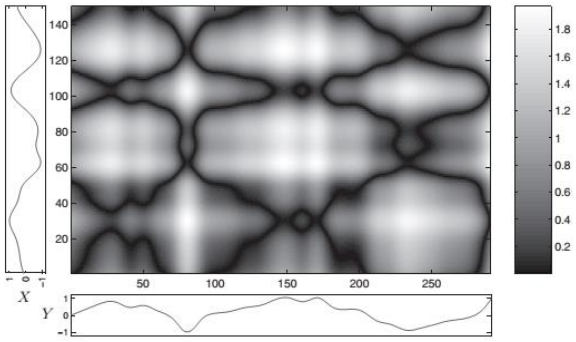

Figure 3. The optimal warping path aligning time series

The local cost matrix for the alignment of two sequences X and Y, using the Manhattan distance, take the sum of the absolute values of the differences of the coordinates:

$$C \in \mathbb{R}^{N \times M} : C_{i,j} = \|x_i - y_j\|, i \in [1:N], j \in [1:N] \quad (3)$$

When the local cost matrix created, the algorithm finds the alignment path that runs through the low-cost areas “valleys" on the cost matrix. This warping path defines the $c: F \times F \to \mathbb{R} \geq 0$correspondence of an element $x_i \in X$ to $y_j \in Y$ following the boundary condition that assigns first and last elements of X and Y to each other (Müller, 2007).

With the Kinect DTW, gesture recognition is fast, reliable and highly customizable. It supports skeleton tracking and 2D vector gesture

recognition that works with all joints of the upper body (skeletal frame).

There are three main classes which compare the gesture, Dynamic Time Warping nearest neighbor sequence class, class to analyze the data of the skeleton, and Kinect SDK skeletal frame coordinates and converts them into a DTW format.

The DTW Gestures Recognizer class use the Kinect runtime that works as a gesture listener. It recognizes gesture in the given sequence. It will always assume that the gesture ends on the last observation of that sequence. If the distance between the previous observations of each sequence is too high, or if the overall DTW distance between the two sequence is too high, no gesture will be recognized.

A rational number of frames is needed before the system attempts to synchronize gestures with stored sequences. If the system finds a corresponding gesture, it enters the file name as a phrase in the text field and pronounces it as a verbal phrase. Here, Figure 4 describes the flowchart for training the gesture and translation to a phrase.

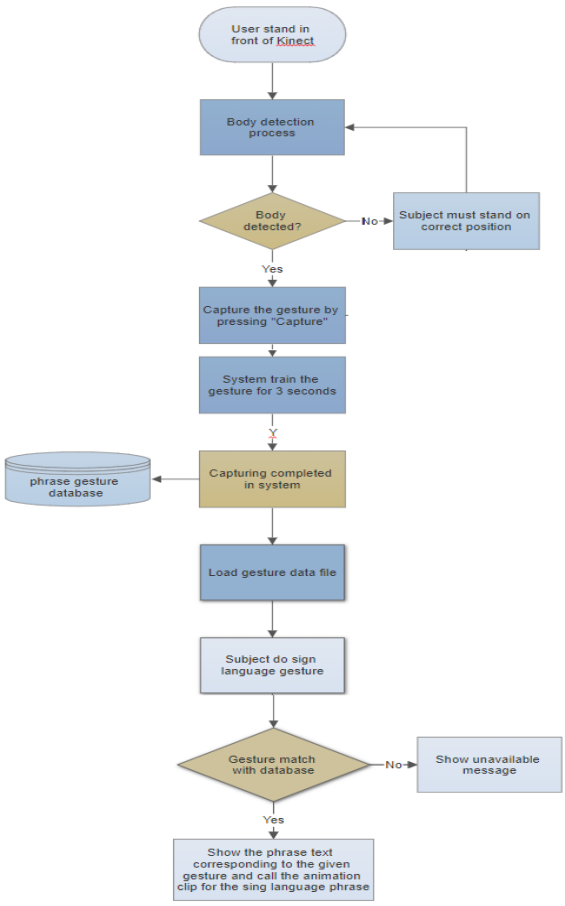

Figure 4. The proposed framework for training and translation method

Database Construction

To create the ASL character database for the phrase and the sentence, we implemented software to capture character gestures. It offers a GUI environment that allows the user to gather the ASL gestures. The sign gesture is used to train the system as shown in Figure 5.

Kinect DTW library is used to create a database of gestures associated with the word or phrase in sign language. They are phrases or simple sentences stored in a text file. The selection of phrases tailored to the needs of a system at airports, terminals or railway stations. Skeleton tracking and 2D vectors support gesture recognition with all joints of the upper body (skeleton frame).

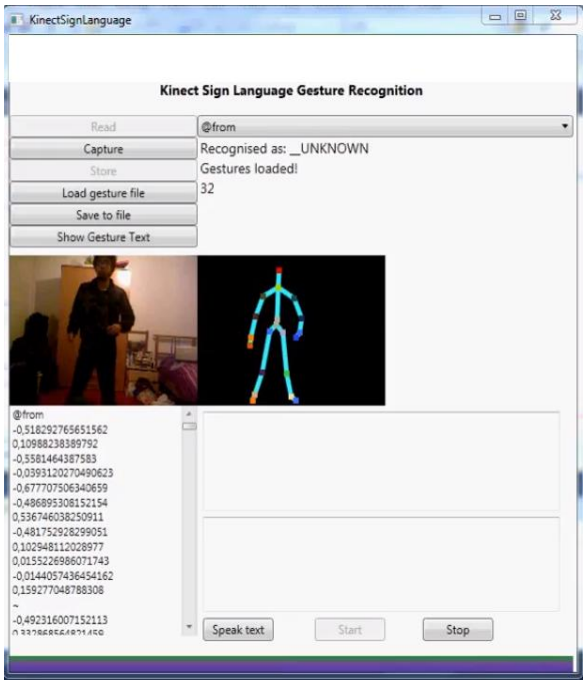

Figure 5. Sign gesture acquisition software interface

Users only have to perform their gestures based on the names of the gestures in the selection field. Then, select the desired gesture name for the recording and click the Capture button. KinectDTW then begins counting for three seconds before the gesture recording is initiated to allow the user to prepare. By default, the system records the gesture up to 32 frames before the user finishes the training. Fifty phrases and sentences were assembled, which are used by the gesture recognition module to translate sentences from hearing impaired people via sign language into text. The text generated from the database query causes the "text-to-speech" module to read the word verbally.

Motion Capture And 3D Animation Work

Part of Kinect Sign Language is a 3D animation video with images of sign language taken by an animated character. For ordinary purposes, both for non-animators and hobbyists, this technique, known as electromechanical motion detection, was less practical for detecting a realistic human movement. Kinect allows developers to use its skeleton tracking capabilities to capture human motion and import it into any motion capture software. To capture and reconstruct the realistic movement of a 3D model, a rig with sensors connected to each joint of real actors performing or imitating an action is the most common method. Animators can use Kinect's skeleton tracking capabilities to capture human motion and import it into any motion capture software. This research uses MikuMikuDance (MMD) software (Higuchi, 2008) combined with Kinect Camera to capture sign language and later process it as an animation clip.

The production phase included the creation of the entire final visual element of the 3D animation project Sign Language:

1. Create Layout

MMD was used to create a layout and an animated character positioned in the environment to capture movement with Kinect. The camera was placed statically in front of the figure. The screen size has been set to 320 pixels long and 240 pixels wide.

2. Development Phase

As a precaution against the large file data, the study defined that the sign language lasted a maximum of 2 seconds with the number of render frames at 30 fps (frames per second). If the data was a complete sentence gesture, then the maximum duration was five seconds or 150 images.

3. Modeling

The red costume customer service character was chosen from some character available in the MMD.

4. Rigging

A control rig is applied into an animated character to move the object as shown in Figure 6. The control rig is made up of the character's gesture positions, which range from head, shoulders, elbows, hips, knees, heels, fingers and accessories on the character's body. The control unit corresponds to the joint position, which according to Kinect SDK has detected a total of 22 joints.

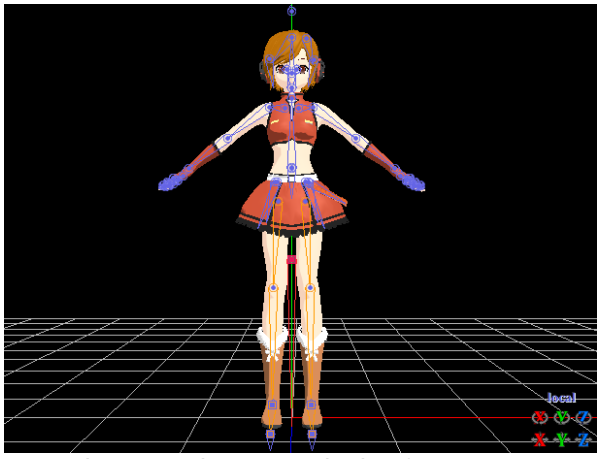

Figure 6. Character Rigging from MMD

MMD Gesture Library Development

The brief steps to create the animation for sign language gesture are explained below:

1. Load the model

Load the model character by clicking the load button in model manipulation, then select the model.

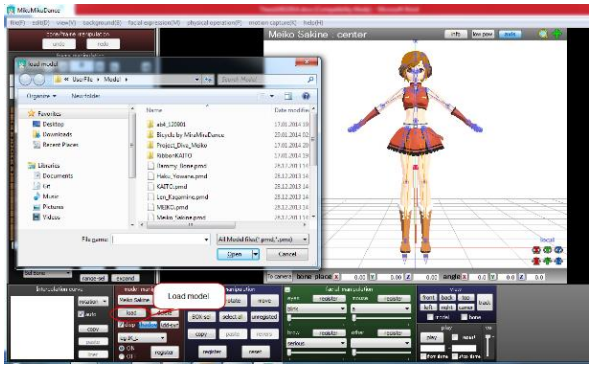

Figure 7. Load the model

2. Set the camera

To get half-body character onscreen, click “To camera” button to change the panel to camera edit mode (Figure 8). The button then turns to “To model”. Next to the button is the camera position in XYZ. Type number 16 in Y-axis. Move to camera panel and change the default view angle from 30 to 15. Press the register button in the camera panel. Then the character should appear in a half-body onscreen.

3. Train the gesture

Open a sign gesture video that already prepared to emulate as a sample shown in Figure 9. Train our self to follow the sign gesture correctly before start capturing.

Figure 8. Set the camera

4. Capture the motion

Connect the Kinect sensor to PC. Begin Kinect mode from motion capture panel, select Kinect (K). After starting the Kinect mode, the shadow man in red will appear on the screen to reflect the infrared of a user body (Figure 10). Then select panel capture (C) to start recording.

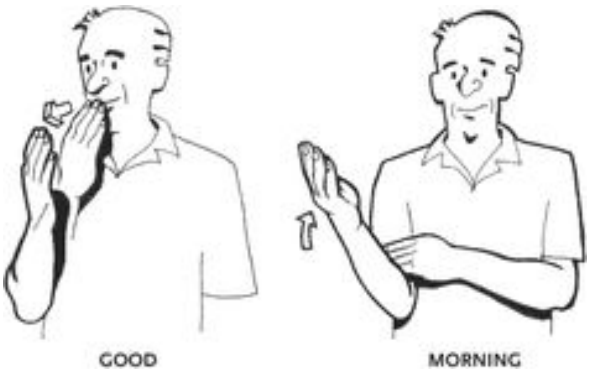

Figure 9. ASL gesture "good morning"

Figure 10. Record the motion using Kinect

5. Past processing

After capturing the sequence, animation data was created in the form of a sign gesture. The gesture that emerged from the motion capture was still rude. Thus, the animation had processed to the *past-production* to fix it. The Figure 11 shows the sample to fix jittering resulted from capturing. This past-processing should be carried out for every frame for around 100 frames for every sign.

6. Rendering

Once the editing process completed, the final step was to render the gesture and save the animation into a single *avi* file format. To begin rendering, click on the panel "File" and select "Render to AVI files (V)". The *avi* video size was set at 320 x 240 pixel as previously defined in layout setting. The frame rates were normally set as default for 30 frames per second (fps).

Figure 11. The motion after capturing has a lot of jittering. Re-shape and edit rigging should be carried out for every frame to fix it. Hand gesture before fix (above) and after being edited by rotating it to the correct position (below)

RESULTS AND DISCUSSION

User Experience Evaluation

Two screens on the user interface represent both target users. The first target user or hearing-impaired people seemed to mirror themselves to make the gesture. The second target user is served by a 3D-animated customer service, which demonstrates an SL based on the spoken text.

Each target user also has its text field. The first target text box is the text from the DTW algorithm, which compares the gestures with the data gestures and displays them in text form. Its function is to show the phrases translated into a voice by the SL gesture. The second target user text box is the work of the Sound to Text algorithm, which extracts the sound into the text as shown in **Error! Reference source not found.**.

Table 1. User Interfaces (UI) preference test result

Subject	UI Preference number
1	3
2	2
3	4
4	2
5	3
6	3
7	4
8	2
9	2
10	4

The position of both the displays and the text box has been set to one of four possible options, which were used to ask questions to each participant for their preferred position (Figure 12 and Figure 13).

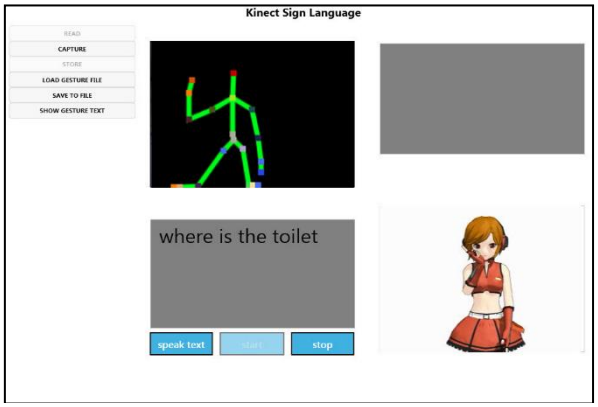

Figure 12. First Interface Option

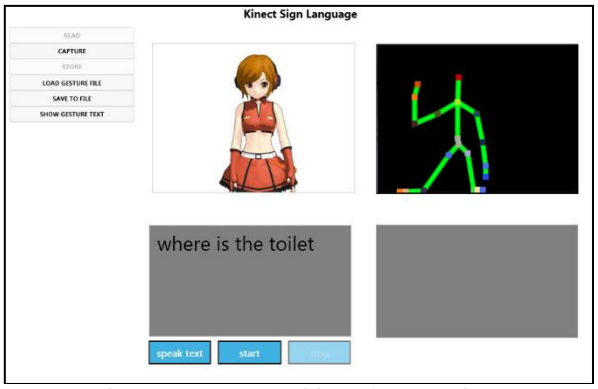

Figure 13. Second interface option

Gesture Evaluation

Each participant was trained on gesturing the ASL "happily" and " good". ASL for "happy" is carried out by turning one or both hands in front of the chest. When the swingarm rises, the palm hardly moves against the chest. Meanwhile, the palm moves away from the chest as the arms swing down. In daily use, this sign worked with only one hand. Participants were asked to play "happily" only with the right hand without the left hand.

ASL for "good" is done by placing the fingers of the right hand against the lips. The left hand is placed with the palm up in front of the stomach. Then the right hand moved towards the left palm. The end position, right palm is pointing up above the left palm.

Participants were requested to use the ASL phrases "Happy" and "Good" with the number of trials as listed in Table 2.

Table 2. Participant Response

	Happy	Good
P1	1	2
P2	2	1
P3	1	1
P4	1	2
P5	2	1
P6	2	1
P7	1	2
P8	1	1
P9	2	1
P10	1	2
Total 1st attempt	**6**	**7**

The test results for both phrases were almost similar. Six participants are rated "happy" on the first attempt and seven participants "good". Regarding complexity, the "happy" sign was more complicated because the hand rotation gesture had to be repeated more than once. However, the system was able to recognize the gesture of the participant with experimental repetition no more than once.

Overall System Evaluation

The level of success may be examined in terms of the user experience. The result has been shown in Fig. 14, that some tasks are accomplished without any difficulty, and others are done with the minor or major issue.

According to some trials during the test, a four-point scoring method has been used for each task as follow:

1 = No issue. The participant successfully finished the task smoothly in the first attempt without any issue.

2 = Minor issue. The participant successfully finished the task but in the second attempt. They made small gesture mistake but quickly recovered and successful.

3 = Major issue. The participant successfully finished the task but in the third attempt. They struggled and hardly tried to accomplish the task.

4 = Failure / gave up. The participant finished the task but in the fourth attempt or gave up before completing the task.

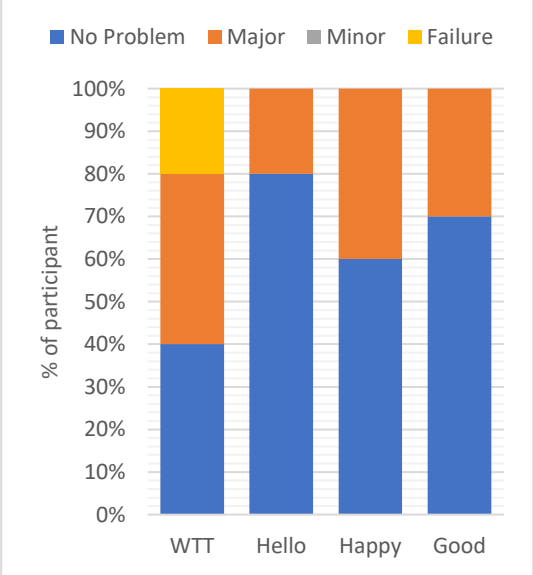

Figure 14. Stacked bar chart showing a different level of success for Kinect Sign Language system

Four of the tasks performed by previous evaluations were measured in a stacked bar chart by scoring the percentage level of success above, as shown in Figure 14. Base on the task completion, the task "hello" from ASR test gained the highest “usability score” with the large percentage of success up to 80%, followed by task "good" and “happy” of the gesture. Instead, the task "WTT" for ASR provided lowest usability score with the highest failure rate up to 20%. However, the success percentage of "Hello" is at acceptable levels.

Discussion

This report examines a sign language translation system based on Microsoft Kinect for Xbox. This system is motivated by the importance of real-time communication between hearing-impaired people with customer service in kiosks such as train stations, airports, shopping centers, banks, etc.

The evaluation resulted in knowledgebase positions and their use with this NUI system. There appears to be a clear difference between the acceptance and a crossed screen compared to parallel screens. Users disliked a crossed display because it sounds confusing, especially when it comes to the relationship between the individual text fields and the displayed destinations.

The change of animation based on the spoken phrase (e.g., when users gesture the SL phrases "happy" and "good") was acceptable to all users. When the phrase is read by Customer Service, a user appears as an animation, along with the conversion of speech to text. In this context, the participants mentioned that the animation would help the deaf to understand the sentence.

CONCLUSIONS

Such an approach can be used to improve the user experience and increase the efficiency of further use as a kiosk. It intends to further enhance the system capability by addition of database to ASL gestures and phrases to store gesture data in SQL format. Also, localization into another language and addition of palm gesture recognition. However, this research has not managed to combine the palm of the hand with the gesture of the hand. Because the Kinect sensor lacks a direct interface to access the palm. Some algorithms can also recognize the gesture of the palms, although performing poorly. This issue is a real challenge, as sign language depends heavily both on the palm and finger gestures. A future research option is the integration of depth detection cameras, which have the special function of recognizing the palms and fingers, such as Leap Motion.

.

REFERENCES

Adriansyah, A., Gunardi, Y., Badaruddin, & Ihsanto, E. (2015). Goal-seeking behavior-based mobile robot using Particle Swarm Fuzzy Controller. *TELKOMNIKA (Telecommunication Computing Electronics and Control).* 13(2): 528-538. http://dx.doi.org/10.12928/TELKOMNIKA.v13i2.1111

Akmeliawati, R., Ooi, M.P.L., & Kuang, Y.C. (2007). Real-Time Malaysian Sign Language

Translation using Colour Segmentation and Neural Network. *2007 IEEE Instrumentation & Measurement Technology Conference (IMTC 2007)*. Warsawa, Polandia. 1-6. http://dx.doi.org/10.1109/IMTC.2007.379311

Aswari, P. and Diana, N.E. (2016). Identifikasi Emosi berdasarkan Action Unit menggunakan Metode Beizer Curve. *SINERGI.* 20(1): 74-80. http://dx.doi.org/10.22441/sinergi.2016.1.010

Auslan, K. (2012). *Auslan Recognition Software for Kinect*. Retrieved from https://code.google.com/p/kinect-auslan/

Bellman, R., & Kalaba, R. (1959). On adaptive control processes. *IRE Transactions on Automatic Control.* 4(2): 1–9. http://dx.doi.org/10.1109/TAC.1959.1104847

Buchmann, V., Violich, S., Billinghurst, M., & Cockburn, A. (2004). FingARtips: gesture based direct manipulation in Augmented Reality. *The 2nd International Conference on Computer graphics and interactive techniques in Austalasia and Southe East Asia - GRAPHITE '04*. New York, USA. 212-221. http://dx.doi.org/10.1145/988834.988871

Cerezo, F. (2011). *3D Hand and Finger Recognition using Kinect*. Project Report. University of Granada. 1–6.

Fitrianah, D., Praptono, N. H., Hidayanto, A. N., & Arymurthy, A. M. (2015). Feature Exploration for Prediction of Potential Tuna Fishing Zones. *International Journal of Information and Electronics Engineering*. 5(4): 270–274. http://dx.doi.org/10.7763/IJIEE.2015.V5.543

Hazari, S. S., Asaduzzaman, Alam, L., & Goni, N. Al. (2017). Designing a sign language translation system using Kinect motion sensor device. *2017 International Conference on Electrical, Computer and Communication Engineering (ECCE)*. 344–349. http://dx.doi.org/10.1109/ECACE.2017.7912929

Hore S. et al. (2017) Indian Sign Language Recognition Using Optimized Neural Networks. In: Balas V., Jain L., Zhao X. (Eds) *Information Technology and Intelligent Transportation Systems. Advances in Intelligent Systems and Computing*, 455. Springer, Cham. http://dx.doi.org/10.1007/978-3-319-38771-0_54

Higuchi, Y. (2008). *MikuMikuDance*. Retrieved May 19, 2018, from http://www.geocities.jp/higuchuu4/index_e.htm

Joshi, A., Sierra, H., & Arzuaga, E. (2017). American Sign Sign Language Translation Using American Using Edge Edge Detection and Cross Correlation Detection. *2017 IEEE Colombian Conference on Communications and Computing (COLCOM).* Cartagena, Colombia. 1-6. http://dx.doi.org/10.1109/ColComCon.2017.8088212

Kyatanavar, M. R. D., & Futane, P. P. R. (2012). Comparative Study of Sign Language Recognition Systems. *International Journal of Scientific and Research Publications.* 2(6): 1–3.

Müller, M. (2007). Dynamic Time Warping. In *Information Retrieval for Music and Motion.* Springer, Berlin. 69–84. http://dx.doi.org/10.1007/978-3-540-74048-3_4

Myers, C., Rabiner, L., & Rosenberg, A. (1980). Performance Tradeoffs in Dynamic Time Warping Algorithms. *IEEE Transactions on Acoustics, Speech, and Signal Processing*. 28(6): 623–635. http://dx.doi.org/10.1109/TASSP.1980.1163491

Priambodo, B., & Ahmad, A. (2017). Predicting traffic flow based on average speed of neighbouring road using multiple regression. *In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 10645 LNCS: 309–318. http://dx.doi.org/10.1007/978-3-319-70010-6_29

Sakoe, H., & Chiba, S. (1978). Dynamic Programming Algorithm Optimization for Spoken Word Recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*. 26(1): 43–49. http://dx.doi.org/10.1109/TASSP.1978.1163055

Shanableh, T., & Eqab, A. (2017). Android Mobile App for Real-Time Bilateral Arabic Sign Language Translation Using Leap Motion. *2017 International Conference on Electrical and Computing Technologies and Applications (ICECTA)*. Ras Al Khaimah, UAE. 1–5. http://dx.doi.org/10.1109/ICECTA.2017.8251936

Tappert, C. C., Suen, C. Y., & Wakahara, T. (1990). The state of the art in online handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 12(8): 787–808. http://dx.doi.org/10.1109/34.57669

Uebersax, D., Gall, J., Van Den Bergh, M., & Van Gool, L. (2011). Real-time sign language letter and word recognition from depth data. *The IEEE International Conference on Computer*

Vision. Barcelona, Spain. 383–390. http://dx.doi.org/10.1109/ICCVW.2011.6130267

Vial, J. et al. (2009). Combination of dynamic time warping and multivariate analysis for the comparison of comprehensive two-dimensional gas chromatograms. Application to plant extracts. *Journal of Chromatography A*. 1216(14): 2866–2872. http://dx.doi.org/10.1016/j.chroma.2008.09.027

Zafrulla, Z., Brashear, H., Starner, T., Hamilton, H., & Presti, P. (2011). American sign language recognition with the Kinect. *Proceedings of the 13th International Conference on Multimodal Interfaces - ICMI'11*. New York, US. 279-286.http://doi.org/10.1145/2070481.2070532