# Sign Language Recognition using Microsoft Kinect

Anant Agarwal
Department of Computer Science and Engineering
Jaypee Institute of Information Technology
Noida, India – 201307
me@anantagarwal.in

Manish K Thakur
Department of Computer Science and Engineering
Jaypee Institute of Information Technology
Noida, India – 201307
mthakur.jiit@gmail.com

**Abstract— In last decade lot of efforts had been made by research community to create sign language recognition system which provide a medium of communication for differently-abled people and their machine translations help others having trouble in understanding such sign languages. Computer vision and machine learning can be collectively applied to create such systems. In this paper, we present a sign language recognition system which makes use of depth images that were captured using a Microsoft Kinect® camera. Using computer vision algorithms, we develop a characteristic depth and motion profile for each sign language gesture. The feature matrix thus generated was trained using a multi-class SVM classifier and the final results were compared with existing techniques. The dataset used is of sign language gestures for the digits 0-9.**

***Keywords—gesture recognition, machine learning, computer vision, support vector machine, kernel, sign language recognition***

## I. INTRODUCTION

Over the past few years, we have seen the rapid development of gaming consoles like Microsoft Kinect, Nintendo Wii, and Sony PlayStation etc. The ability of these consoles to capture body movement information through 3D depth sensors has made them very popular not just among general public but in academia as well [5]. A very major use of these sensor technologies is in gesture recognition and in tracking body movement [9]. A popular application of gesture recognition is in the area of Sign Language Recognition which would provide the differently-abled with a platform to express their views.

A vast amount of research has already be done in the area of recognizing Sign Languages mostly using techniques like body part tracking, neural networks, Hidden Markov Models, skeleton detection etc. [9],[10,[11].

Other prominent techniques include utilizing the motion history information associated with gestures, motion capturing gloves and computer vision combined with different colored gloves [17], [18].

In our alternate approach, we make use of the Microsoft Kinect depth sensor since it provides a major advantage which essentially is the fact that they make the entire application environment independent. Depth sensors such Microsoft Kinect work independent of environmental constraints like low lightening, noisy input etc. and are quite inexpensive [1], [16]. They simply capture the entire gesture as a sequence of depth frame of varying intensity. This advantage provided by the Kinect camera validates our approach of exploiting the depth maps of gestures and build a framework using machine learning techniques for such a purpose.

The remainder of the paper is organized as follows. Section II describes the alternate techniques used by researchers to carry out sign language recognition whereas Section III discusses the design scheme that we have proposed to extract maximum possible information from a depth sequence. Section IV describes the experiments we have conducted on the dataset and its analysis and interpretation.

## II. RELATED WORK

As previously mentioned, a lot of research has already been done on gesture recognition using conventional techniques like body part tracking, different color glove based tracking etc. however the use of Kinect like depth sensors has been adopted for gesture recognition in the recent past and it has been growing in popularity ever since. We studied a number of possible design techniques, existing applications running such systems etc. in order to develop a better understanding of the eventual functional requirements and design parameters of our system.

A very robust system for sign language recognition was implemented in [16] using skeleton tracking through the custom applications provided by Microsoft as part of their SDK for Kinect. The authors' dataset consists of about 50 samples from 2 users each and using LIBSVM, they achieve a classification accuracy of about 97%. However, the authors' approach fails in case of sign languages gestures that involve different finger movement since the skeleton tracking is done for the entire hand and not for individual fingers.

Biswas and Basu [1] used a more sophisticated and viable approach for gesture recognition using depth images. This paper has greatly influenced our work since the authors' have made use of low level features to achieve above 90% accuracy on all of the 8 gestures sequences they had considered in their dataset. However, this technique makes use of only low level features and doesn't quite explore high level features like optical flow characteristics.

The authors' in [15] have presented a number of techniques like hand shape analysis, human detection and human body part detection for the Japanese Sign Language.

They apply their approach to recognize sentence structures and claim to have a high recognition rate.

In short, the use of depth sensing technology is fast gaining acceptance and recognition. Systems such as [17],[13],[2] and [5] are other alternate techniques that have been implemented before, have reasonably good performance and have made some or the other changes in the feature set or the training algorithm [13] to improve upon the short-comings in previous works.

## III. DESIGN

Our entire approach is divided into two major feature capture modules [1]. One module focuses on building the depth profile of the gesture whereas the other module focuses on building the motion profile of the gesture. Through our analysis of existing techniques and approaches, we found that both depth and motion profiles are essential to provide good accuracy and contribute equally in developing robust sign language recognition systems [16],[1],[15],[9],[2].

For the purpose of feature selection, we have used the techniques described in [1]. We found the technique to be fair to both depth and motion information and the technique captures the characteristic nature of the gestures in a very comprehensible and exploitable manner.

Our entire processing was on a stream of depth images and the feature matrix was generated using the collective information from all the frames of the image stream; thus all the consequent operations discussed below are on a single video frame which is treated as a single image as shown in Figure 1. Accordingly, we work on deriving a feature vector for each frame and collectively use all the feature vectors pertaining to a particular sign language number gesture sequence for classification purpose.

We first applied a Gaussian blur filter and then eroded the image to remove unwanted noise to generate a clear depth map as shown in Figure 1. The parameters of the blurring and erosion filters were hand tuned for each of the two datasets that we have used in our experiments. Next in order to remove the background, we followed the technique as proposed in [14] which uses an iterative threshold selection technique for background subtraction. We plot a depth histogram of the image by using a 256 bin vector for each intensity value of the depth image as shown in Figure 2(a); which had been converted into a single channel grey-scale image.

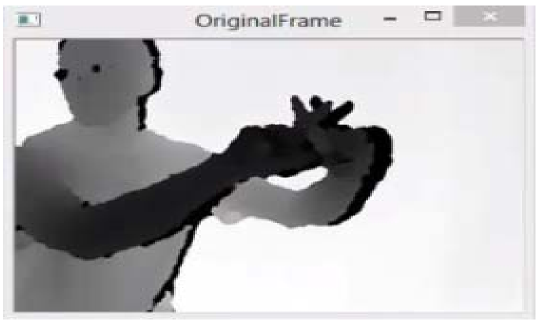

**Figure 1.** Sample Depth Image

As mentioned in [1], the first valley following the largest peak is the foreground and accordingly we extract this foreground for further processing. Since the peaks of the histogram were uneven and it was difficult to capture the first peak, we had to apply a median blur filter to smoothen the edges as shown in Figure 2(b); thus making the first largest peak prominent. Upon extraction of the foreground, we plotted an equalized depth histogram of the foreground and found it to be characteristic for each gesture [1].

A characteristic equalized depth histogram is shown in Figure 2(c). Further development of the depth profile is all based on this extracted foreground whereas the motion profile is still built on the original video image.

Once we have the foreground image, we start building the depth profile in parallel to the motion profile. Using one thread, we captured the depth characteristics of the gesture by applying an 8x8 grid mask on the image.

For each grid cell, we created a 10 bin vector (0-10, 11-20, 21-30, 31-40, 41-50, 51-70, 71-100,101-120,121-155,155-255) and accordingly updated the count of each bin by counting the number of pixels lying within the defined range of the particular grid cell.

For each cell in the 8x8 grid, a separate distinct bin vector was maintained which was further normalized for scalability issues. The consensus of all bin vectors of a particular frame constituted the depth profile of that particular frame.

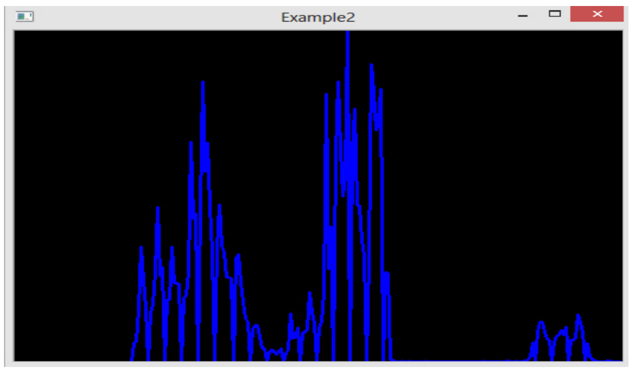

**Figure 2(a).** Depth Histogram

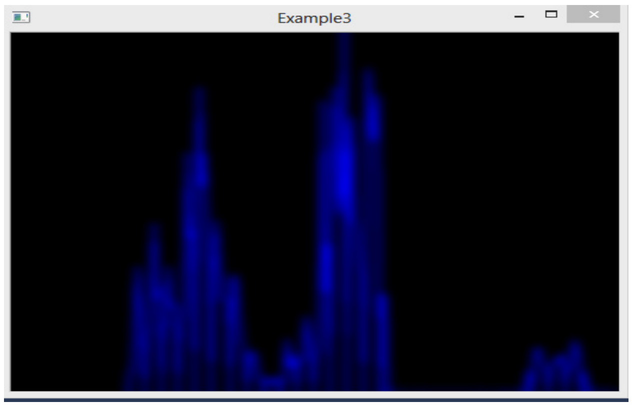

**Figure 2(b).** Eroded depth Histogram

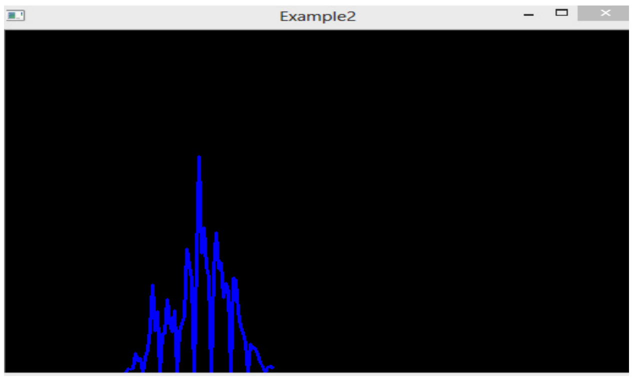

**Figure 2(c).** Histogram of Extracted Foreground

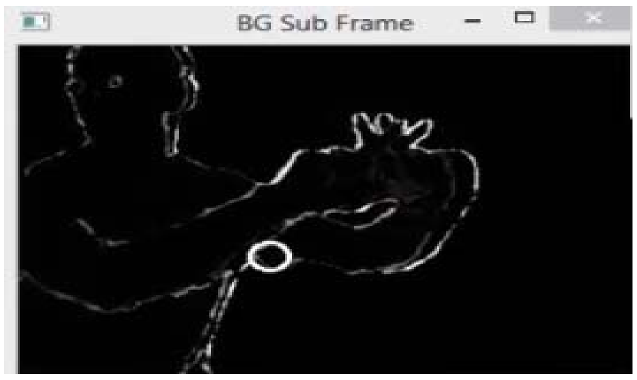

**Figure 3.** Motion Difference Image

Through the other thread, we captured the motion characteristic of the gesture by first taking the difference of two subsequent image frames to reveal the motion history of the gesture through the motion difference image clearly shown in Figure 3. We exploit this motion history by placing our 8x8 grid over the difference image and maintaining the count of the bin vectors for each cell of the masking grid. This was accordingly normalized similar to what we had done for the depth profile. The consensus of all the bin vector of all the cells constituted the motion profile of that particular frame.

Consequently, we repeat this process for each frame of the video for the particular gesture and thus end up generating a feature vector for each frame. The combined array of feature vectors of all the frames in the video sequence allows us to generate our feature matrix for that video sequence or that particular gesture. Accordingly we add label to each feature matrix that are part of the training set. Additional details and descriptions of the dataset, feature matrix generation and the classification module are mentioned in the sections below.

## IV. EXPERIMENT AND ANALYSIS

For the purpose of our study, we chose the Chinese Number Sign Language dataset from the ChaLearn Gesture Dataset (CGD 2011) repository.

We conducted our experiments on two datasets kept under the repository named devel 44 and devel 59. These datasets in particular deal with Chinese Numbers Sign Languages and were chosen randomly from many such already present datasets. Each dataset has about 47 video sequences consisting of not only individual gestures but also sequences of multiple gestures.

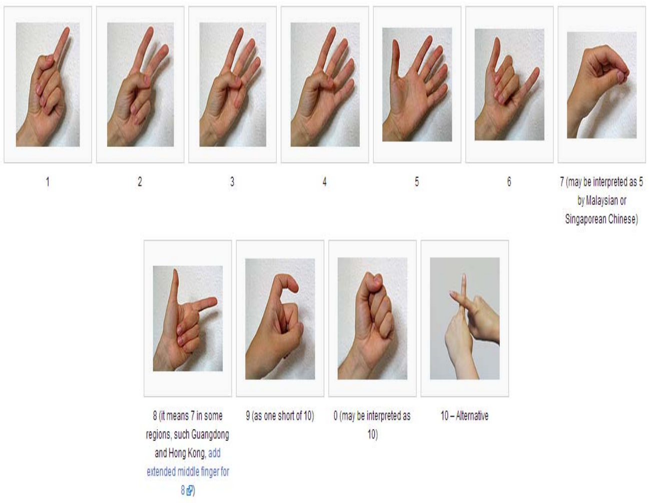
1
2
3
4
5
6
7 (may be interpreted as 5 by Malaysian or Singaporean Chinese)
8 (it means 7 in some regions, such Guangdong and Hong Kong, add extended middle finger for 8)
9 (as one short of 10)
0 (may be interpreted as 10)
10 – Alternative

**Figure 4.** Chinese Number Sign Language Gestures

Each dataset also has specific train and test video sequences which we have explicitly used in our experiments. Additionally each dataset comprised of both the depth and RBG video sequences; however for the purpose of our study we only chose the depth video sequences.

As mentioned, we had hand tuned the parameters for the Gaussian and the erosion filter. For each dataset, an 8 unit radius Gaussian blur filter was used and a rectangle was used as the erosion shape.

On each depth video sequence, we apply our mentioned technique to generate a feature vector for each frame of the video sequence; thereby generating a feature matrix for each gesture of the dataset. This is done for both the training as well as the test dataset. Thus for each dataset, we have a single train file that contains the feature matrices of all the 10 sign language number gestures.

Similarly for each dataset, we have multiple test files each consisting of the feature matrices of the single as well the multiple sign language number gesture present in the test video sequences. Once we have the training file ready, we send it to a multi-class SVM classifier in order to generate a model for classification of the test files.

SVM is a training algorithm that learns classification and regressions rules and generates a model that can be used for further classification purpose. The SVM classifier aims to identify a handful of points called the support vectors from the entire dataset which allow us to draw a linear or a non-linear separating surface in the input space of the dataset.

SVM models are typically built with the help of kernel functions which allow us to transform the data into n-dimensional space where a hyper plane can be easily constructed to partition the data. We have used the linear and the RBF kernel separately in our approach.

Our performance metric is the classification accuracy and training time which we achieve by running the test gesture sequences on our trained model.

Our test set consists of individual gesture as well as combination of multiple gestures and thus, computes the classification accuracy with which our trained model is successfully able to identify the gestures present in the our test file.

We compute separate classification accuracy for each test file since we have a total of 3 test files for each dataset. In total we had 3 test runs for each dataset i.e. a total of 6 test runs for the two datasets.

Results obtained from our experiments are mentioned below in Table 1, Table 2, Table 3 and Table 4 respectively. We describe the rationale behind each set of the result below.

Moreover, in order to further optimize our results, we added OpenMP directives to our algorithm in order to leverage the use of multiple cores present in our CPU's. We thus were able to multi-thread the application and gain a slight improvement in performance.

Our analysis is done on a 2.63 GHz Intel Dual Core Processor with 4 GB of RAM running a Ubuntu 12.10 Linux distribution. We used OpenCV 2.3 library for implementing the computer vision algorithms since it has relatively fast implementations of various algorithms.

We also used the LIBSVM library [6],[7],[8] for training and classification of the sign language gestures. Both the libraries are standard and popular in academic setting and thus allowed us to efficiently implement our designs and carry out our experiments in an optimized manner.

**TABLE 1.** CLASSIFICATION ACCURACY

| Gesture Set | Linear Kernel | RBF Kernel |
|---|---|---|
| Gesture Set 1 | 89.63% | 92.313% |
| Gesture Set 2 | 77.59% | 90.83% |
| Gesture Set 1+ Gesture Set 2 | 81.48% | 87.67% |

**TABLE II.** TRAINING TIME

| Gesture Set | Linear Kernel | RBF Kernel |
|---|---|---|
| Gesture Set 1 | 10.53 sec. | 5.74 sec. |
| Gesture Set 2 | 6.07 sec. | 4.99 sec. |
| Gesture Set 1+ Gesture Set 2 | 12.77 sec. | 8.75 sec. |

**TABLE III.** TRAINING TIME FOR SINGLE AND DOUBLE THREADED APPLICATION OVER THE LINEAR KERNEL

| Gesture Set | Single Thread | Double Thread |
|---|---|---|
| Gesture Set 1 | 12.76 | 10.53 |
| Gesture Set 2 | 7.51 | 6.01 |
| Gesture Set 1+ Gesture Set 2 | 14.27 | 12.77 |

**TABLE IV.** TRAINING TIME COMPARISON FOR SINGLE AND DOUBLE THREADED APPLICATION OVER THE RBF KERNEL

| Gesture Set | Single Thread | Double Thread |
|---|---|---|
| Gesture Set 1 | 6.37 | 5.74 |
| Gesture Set 2 | 5.25 | 4.99 |
| Gesture Set 1+ Gesture Set 2 | 10.82 | 8.99 |

Our experiments yielded the following results for the classification accuracy as shown in Table 1, the results for the training time ( best possible) in Table 2, the difference in the training time when running the application on a single and double thread using a linear or RBF kernel in Table 3 and Table 4 respectively.

The graphical representations of the results are shown below in the form of Figure 5, Figure 6, Figure 7 and Figure 8 as standard column charts where the unit of time is seconds and the classification accuracy is measured as the percentage of correct classifications divided by the total number of sample.

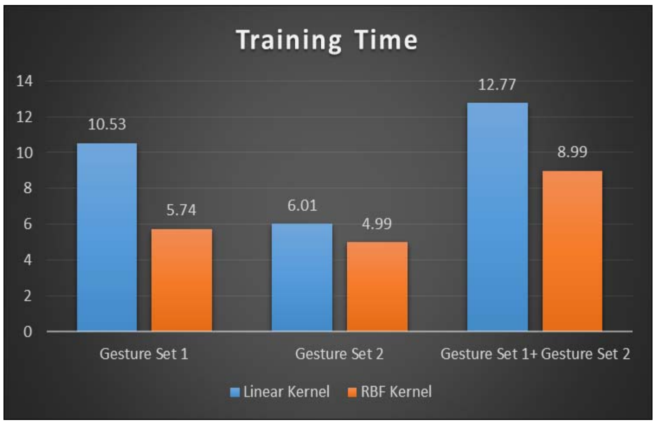

**Figure 5.** Training Time

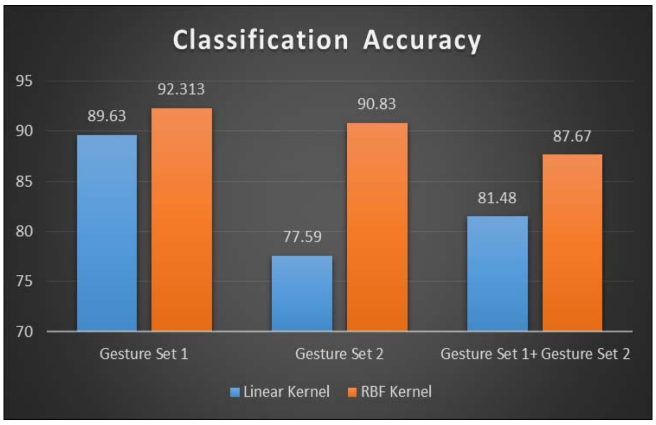

**Figure 6.** Classification Accuracy

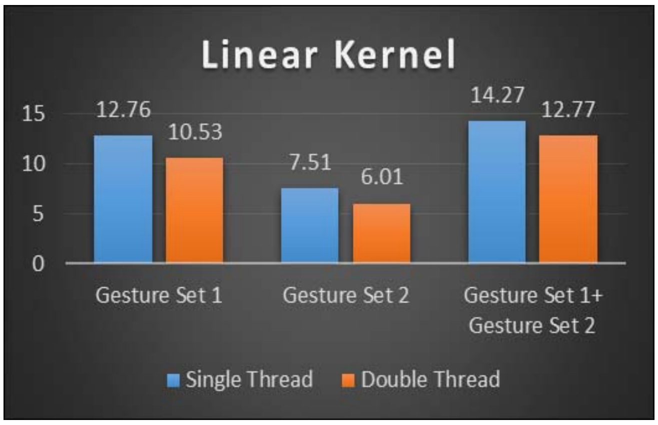

**Figure 7.** Training Time comparison for single and double threaded application over the Linear kernel.

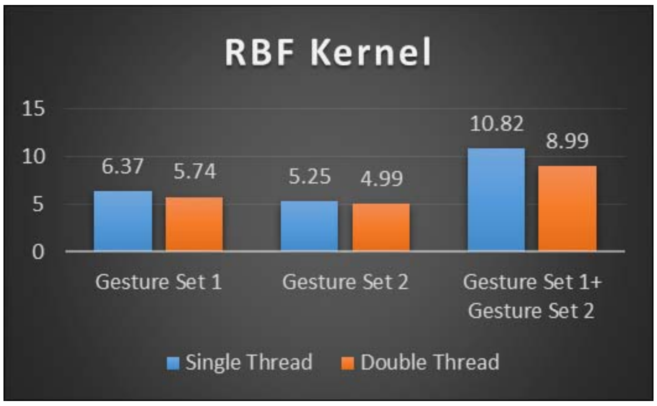

**Figure 8.** Training Time comparison for single and double threaded application over the RBF kernel

## CONCLUSION

Thus, through our experimentation, we clearly showed that an efficient sign language system can be implemented using depth images. Even though our dataset had about 20 sequences in the training set, we can safely assume that the entire process can be easily extended to a larger dataset as accomplished in [1]. We can bring about improvements in our classification accuracy if we explore high level features like optical flow information, motion gradient information etc. We can safely conclude by stating that through some minor pre-processing and inexpensive feature extraction techniques, we have made a sign language recognition system that works faster in comparison to other techniques that are based on tracking, hand shape analysis or ones that are developed using high level features.

## REFERENCES

[1] Biswas K. K. and Basu S. K. , "Gesture recognition using microsoft kinect®", Proceedings of the 5th International Conference on Automation, Robotics and Applications, Dec 6-8, 2011, Wellington, New Zealand, December – 2011, Pages – 100-103

[2] Tang Mathew, "Recognizing hand gestures with microsoft's kinect®", Stanford University, 2011

[3] Bhattacharya Sambit, Czejdo Bogdan Denny and Perez Nick," Research modules for undergraduates in nachine learning for automatic gesture vlassification", Proceedings of the Twenty-Fifth International Florida Artificial Intelligence Research Society Conference , May 23–25, 2012

[4] Li Tian, Putchakayala Prabhat and Wilson Mike, "3D Object detection with kinect" , Cornell University, 2011

[5] Zhou Ren, Jingjing Meng, Junsong Yuan and Zhengyou Zhang, "Robust hand gesture recognition with kinect sensor", MM '11 Proceedings of the 19th ACM international conference on Multimedia, May 2011, Pages - 759-760

[6] Chih-Chung Chang and Chih-Jen Lin. 2011. "LIBSVM: A library for support vector machines", ACM Trans. Intell. Syst. Technol. 2, 3, Article 27 (May 2011), 27 pages.

[7] Rong-En Fan, Pai-Hsuen Chen, and Chih-Jen Lin. 2005. "Working Set Selection Using Second Order Information for Training Support Vector Machines", J. Mach. Learn. Res. 6 (December 2005), 1889-1918.

[8] Chan, Tan, Jin, "Local support vector and its efficient algorithm", IEEE Transactions on Knowledge and Data Engineering, Volume 22 Issue 4, April 2010, Pages 537-549

[9] Mitra Sushmita and Acharya Tinku, "Gesture recognition: a survey", IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews, VOL. 37; NO. 3, May 2007

[10] V. A. Oliveria and A.Conci, "Skin detection using HSV colour space", Workshops of Sibgrapi 2009

[11] Benjamin D. Zarit, Boaz J. Super, Francis K. H. Quek, "Comparison of five color models in skin pixel classification", Proceedings of the International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems 2009, Pages 58-63

[12] L'ubor Ladicky and Philip H.S. Torr, "Locally linear support vector machines", International Joint Conference on Neural Networks (IJCNN) 2010, Pages 1-6

[13] Christian Schuldt, Ivan Laptev and Barbara Caputo, "Recognizing human actions: a local SVM approach", The International Joint Conference on Neural Networks 2010, Volume 3, Pages 34- 36

[14] T.W.Ridler, S. Calvard, "Picture thresholding using an iterative selection method", IEEE Transaction Systems, Man and Cybernetics, SMC – 8, page 630-632, 1978

[15] Kikuo Fujimura, Xia Liu, "Sign recognition using depth image streams", 7th International Conference on Automatic Face and Gesture Recognition 2006

[16] Frank Huang and Sandy Huang, "Interpreting american sign language", Stanford University term paper for CS 299.

[17] Hao Zhang, Wen Xiao Du and Haoran Li, "Kinect gesture recognition for interactive system", Stanford University term paper for CS 299

[18] Parton, B.S., "Sign langauge recognition and translation : A multidisciplined approach from the field of artificial intelligence". Journal of Deaf Studies and Deaf Education, 11(1):94-101, 2006