AUTOMATIC RECOGNITION OF AMERICAN SIGN LANGUAGE CLASSIFIERS

A Thesis
Presented to
The Academic Faculty

by

Zahoor Zafrulla

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
College of Computing

Georgia Institute of Technology

AUTOMATIC RECOGNITION OF AMERICAN SIGN LANGUAGE CLASSIFIERS

This dissertation is dedicated to my little princess Nazafreen.

ACKNOWLEDGEMENTS

First and foremost I would like to thank my family, my parents, my sisters and my loving wife Nazneen for all the support that made it possible to finish this work. Many thanks to my advisor Dr. Thad Starner who has been very patient throughout and provided encouragement when I most needed it. Thanks to my Ph.D. committee for all the advise and support, particularly Dr. Harley Hamilton whose expertise as an American Sign Language linguist was vital to the CopyCat project. Thanks to current and past colleagues from the CCG group for all the lively discussions that sparked new ideas that strengthened this dissertation. I want to specially thank Himanshu Sahni, Abdelkareem Bedri and Pavleen Thukral without whose help the experiments in this dissertation would not be completed in time.

The CopyCat project would not be successful without the efforts of Dr. Helene Brashear, Peter Presti and Scott Gilliland. Dr. Brashear you have been a great mentor; thank you for all the help in keeping this dissertation on the right track. Peter and Scott, your magical skills in building sophisticated pieces of hardware overnight are unparalleled. Thanks to students and teachers at the Atlanta Area School for the Deaf, Minor Elementary School and Georgia School for the Deaf for your participation in the CopyCat studies.

I offer my sincere thanks to Dr. Robert Hoffmeister, Rachel Benedict, Sarah Fish, Jon Henner and Kelly Kim from Boston University, and Frances Conlin for making the trip to Georgia Tech to help design and collect the ASL classifier dataset.

My heartfelt gratitude to Tavenner Hall for not only being a great source of support to my family during the busiest of times but also for the valuable suggestions that made this a better dissertation.

Finally, thanks to Dr. Gregory Abowd, Dr. Meghan Abowd, Dr. Rosa Arriaga and Dr. Santosh Vempala. My wife and I will forever remember the valuable lessons in life that you all have taught us.

TABLE OF CONTENTS

DEDICATION . . . . **iii**

ACKNOWLEDGEMENTS . . . . **iv**

LIST OF TABLES . . . . **viii**

LIST OF FIGURES . . . . **x**

SUMMARY . . . . **xiv**

I INTRODUCTION **1**

1.1 ASL Classifiers 2

1.2 Thesis Statement 2

II RELATED WORK **13**

2.1 Language modeling in ASL recognition 13

2.2 ASL verification and sign spotting 14

2.3 Speech utterance verification 14

2.4 Hand tracking and detection 15

2.4.1 Sensor-based methods 15

2.4.2 Appearance-based methods 15

2.4.3 Depth-based methods 16

2.5 3D hand pose estimation using depth data 17

2.6 CopyCat Project 18

2.6.1 The CopyCat game 18

2.6.2 Hardware Changes 20

2.6.3 CopyCat Datasets 21

2.6.4 ASL sentence recognition with Gwinnett dataset 22

III VERIFICATION OF ASL SENTENCES **24**

3.1 Experimental Setup 25

3.2 Verification 27

3.2.1 Human Performance 27

3.2.2 Machine Performance 27

3.3 Live deployment of CopyCat 30

3.4 Summary 31

IV A TWO-PASS TECHNIQUE FOR IMPROVED VERIFICATION PERFORMANCE 33

4.1 Verification 35

4.2 Experiments 36

4.2.1 Forward and reverse pass training and testing 37

4.3 Results and Discussion 39

4.4 Summary 44

V ASL VERIFICATION WITH KINECT 46

5.1 Data Collection 46

5.1.1 Kinect Data Collection 47

5.1.2 CopyCat Adult Data Collection using the CopyCat Sensor Platform 50

5.1.3 Data Collection Summary 51

5.2 Experimental Design 51

5.2.1 Features 52

5.2.2 Training and Testing 54

5.3 Results 55

5.3.1 Baseline results from experiments on the Gwinnett dataset 55

5.3.2 Kinect Data 56

5.3.3 CopyCat Adult Data 57

5.3.4 Discussion 58

5.4 Summary 61

VI ASL CLASSIFIER RECOGNITION 63

6.1 Classifier-Kinect dataset 64

6.1.1 Design 64

6.1.2 Data collection 66

6.1.3 Data Labelling 69

6.1.4 Findings 71

6.2 Methodology for recognition of ASL classifier sentences in the Classifier-Kinect dataset 75

6.3 Handshape classification to identify classifier handshapes 76

6.3.1 Feature extraction 77

6.3.2 Experimental results 77

6.3.3 Findings 79

6.3.4 Handshape classification with regenerated depth images 81

6.3.5 Discussion 84

6.4 Training hidden Markov models to recognize noun signs and verb-of-movement classifier predicates 85

6.4.1 Feature extraction 85

6.5 Sentence recognition results on the Classifier-Kinect dataset 86

6.5.1 Discussion 88

VII IMPROVING CLASSIFIER SENTENCE RECOGNITION WITH BETTER HAND DETECTION USING DOMAIN-DRIVEN RANDOM FOREST REGRESSION 90

7.1 Ground truth data for hand locations 91

7.1.1 Skeleton Tracking Errors 92

7.2 Hand detection using domain driven random forest regression 97

7.2.1 Feature Extraction 97

7.3 Training 99

7.4 Experimental Results 102

7.4.1 Comparing domain driven random forest regression (DDRFR) with Microsoft Kinect Skeleton Tracker (MKST). 103

7.5 Discussion 106

VIII CONCLUSION AND FUTURE WORK 108

REFERENCES 111

LIST OF TABLES

1 Research contributions towards development of an American Sign Language verification infrastructure and a method to recognize selected classifier-based grammatical structures of American Sign Language. 5

2 CopyCat game vocabulary 20

3 CopyCat datasets collected for this dissertation 23

4 Computer vision and accelerometer features extracted from the signing data. 28

5 Average sentence verification accuracy on the Gwinnett dataset from 50 cross validation trials using a 90-10 split. 29

6 Training and testing split for signer-independent cross-validation. 39

7 Examples of one-sign-error sentences. Errors are shown in bold. 40

8 Head-to-Head wins comparing verification criteria, $V^{(1)}$ vs $V^{(3)}$ and $V^{(2)}$ vs $V^{(4)}$. 40

9 True positive cases for verification criteria $V^{(2)}$ and $V^{(4)}$. 44

10 Error-free samples collected versus sensor failures for all data collected. . . 51

11 Description of body pose features generated using OpenNI framework (Kinect data). 52

12 Experiments using Gaussian mixture models for the Gwinnett dataset show a maximum sentence recognition accuracy of 67.0% and a maximum sentence verification rate of 82.0% [114]. 55

13 Sentence recognition and verification results for the pruned CC-Kinect-seated dataset (CopyCat Kinect Platform). 56

14 Sentence recognition and verification results for the pruned CC-Kinect-standing dataset (CopyCat Kinect Platform). 56

15 Sentence recognition and verification results for the pruned CC-Adult dataset (CopyCat Sensor Platform). 57

16 Comparison between three configurations of the CopyCat game apparatus based on ratings for Recognition/Verification Accuracy, Robustness, Aesthetics and Comfort, and User Interaction (a rating of X means further investigation required). 58

17 Summary of sentence recognition and verification results on the CopyCat Kinect Platform datasets and the CopyCat Sensor Platform datasets. . . . 59

18 Comparison of sentence recognition and verification results on the unpruned CC-Kinect-seated, CC-Kinect-standing and CC-Adult datasets. . 60

19 Classifiers, classifier handshapes and nouns in the Classifier-Kinect dataset. (images © 2004, www.Lifeprint.com. Used by permission.) 67

20 Verb-of-movement classifier predicates in the Classifier-Kinect dataset. . . . 67

21 Number of signed instances across signers for each variation in sentences that have the classifier predicate MEETS. 74

22 Percentage accuracy of handshape classification for each signer. 79

23 Confusion matrix for classifier handshape classification. (The CL- prefix has been omitted from column names due to lack of space). 79

24 Percentage accuracy of handshape classification for each signer after regeneration of depth data. 83

25 Confusion matrix for classifier handshape classification after regeneration of depth data. (The CL- prefix has been omitted from column names due to lack of space). 83

26 Features for training hidden Markov models to recognize noun signs and verb-of-movement classifier predicates. 86

27 Sentence recognition accuracy. Experimental results on the Classifer-Kinect dataset using two different techniques each for hand detection and handshape classification. 88

28 Comparison of hand detection %error - DDRFR vs MKST. 103

29 Comparison of sentence recognition accuracy DDRFR vs MKST. 103

30 Occupancy ratio in feature space for each signer. 106

LIST OF FIGURES

1 ASL classifier example with accompanying depth map images recorded with a Kinect camera. 3

2 CopyCat apparatus: (a) Flash game (b) gloves with accelerometers (c) kiosk (d) camera view 19

3 Word lattice for the CopyCat game language that follows a sequential sign-based grammatical structure. 25

4 Signing variations for the ASL signs GREEN and CAT observed in the Gwinnett dataset. 26

5 Diagrammatic representation of features extracted using computer vision techniques superimposed on the image of a signer. 26

6 The ROC curve for sentence verification on the Gwinnett dataset using up to two gaussian mixture components in the HMMs. 29

7 Pre and Post test scores showing educational effect of the CopyCat game in a deployment using a live sentence verifier. 31

8 Signing disfluencies that were observed in the Gwinnett dataset: sneezing, scratching, thinking, and interjecting. 32

9 Strict sign-based reverse grammars for the graphic in Figure 2a. a) unmodified b) expanded to accommodate false starts. 34

10 Procedure for applying the verification criteria $V^{(3)}$ or $V^{(4)}$ given in equations 8 and 9 respectively, and choosing segmentations for the signs in ASL sentence $\boldsymbol{P}$. 37

11 Distribution of simulated sign errors for each student. 39

12 ROC plots for signer-independent cross-validation. 41

13 Comparison of verification accuracy for verification criteria $V^{(2)}$ and $V^{(4)}$. (a) Accuracy vs False Alarms (b) Accuracy vs False Rejects (c) Accuracy vs β. 42

14 Parameters for computation of boundary distance. 44

15 Comparison of mean boundary error, after alignment, between verification criteria $V^{(2)}$ and $V^{(4)}$. 45

16 User interface for the sentence repetition task. Video on the left shows a tutorial for the phrase. The right side shows the raw video (bottom) and the video annotated with the skeleton tracking (top). The bottom buttons allow the signer to navigate the system. 48

17 Participant pose for data collection. (a) Seated (b) Standing. 49

18 Tracking failure with BED and ALLIGATOR, respectively (Kinect data). Note that the lines should correspond with the signer's arm positions but do not. 50

19 Visualization of upper body skeletal joints generated using OpenNI framework (Kinect data). 53

20 Procedure for computing log-likelihood thresholds to perform verification. . 55

21 Comparison of verification accuracy between CopyCat Sensor Platform (CC-Adult dataset) and the CopyCat Kinect Platform (CC-Kinect-seated dataset), with and without data collection errors, for different values of the threshold. 60

22 The typical sentence structure for the Classifier-Kinect dataset. (LH - left hand, RH - right hand, do - direct object, subj - subject) 65

23 Gloss-based visual breakdown of the ASL classifier sentence CAR BUMPS-INTO BOY. (do - direct object, subj - subject). 65

24 CL-3 and modified CL-3 convey different meanings. (a) BUMPS-INTO (b) BUMPS-INTO-CRUSHED. 68

25 Data capture configuration to collect signing examples of ASL classifier sentences. 68

26 Screenshot of the segment labelling tool for the Classifier-Kinect dataset. The top-left and top-right panels allow for the selection of the start and the end frame of the segment. The panel in the middle is a custom video player that allows for the segment to be viewed from start frame to end frame at different speeds. The Snippets panel shows the currently labelled segments. Shown above: selection of snippet BOY-RIGHT. 69

27 Add label form to select an appropriate label for the segment. First select handedness and then select one label from one of four categories: Signs, Classifiers, Predicates or Other. 71

28 The five variant sentence structures found in the Classifier-Kinect dataset used in conjunction with the MEETS classifier predicate. (LH - left hand, RH - right hand, do - direct object, subj - subject, MEETS-v1 - originally intended MEETS classifier predicate, MEETS-v2 - variant of the MEETS classifier predicate). 72

29 MEETS-v1: The predicate MEETS as intended in the dataset design where subject moves towards direct-object while direct-object remains stationary. Arrow indicates direction of movement. 73

30 MEETS-v2: The alternate form of predicate MEETS where subject and direct object move towards each other. Arrows indicates direction of movement. 73

31 The classifier predicate JUMPS signed identically for both PERSON and ANIMAL subject nouns. 74

32 Three steps in classifier sentence recognition: 1) segment sentence by identifying landmarks L_1 and L_2, 2) handshape classification and 3) sign and verb-of-movement classifier predicate recognition. (do - direct object, subj - subject). 75

33 Feature extraction for handshape classification. 77

34 Comparison of (a) FENCE and (b) CAR classifier handshapes when used as direct-object (left hand). 80

35 Comparison of depth images of (a) FENCE and (b) CAR classifier handshapes when used as direct-object (left hand). 80

36 Kinect IR dot pattern used to estimate depth. 81

37 Process of regenerating depth values in “holes.” a) color mask of the hand obtained by skin color segmentation b) depth pixels overlayed over color mask c) select an invalid pixel to begin regeneration d) examine local neighboring pixels in four principle directions e) perform linear regression to fit a line to neighboring pixels with valid data f) extrapolate values and compute average g) final result after regeneration. 82

38 Proposed set of classifier handshapes with reduced minimal-pair confusion. (© 2004, www.Lifeprint.com. Used by permission.) 84

39 Three state left-to-right HMM. 85

40 Process of normalizing three-dimensional hand coordinates using X-, Y- and Z-extents computed based on arm lengths. 86

41 ASL classifier sentence recognition pipeline. 87

42 Interrelationship between factors that impact sentence recognition accuracy. An arrow indicates impact on results. 89

43 Labelling ground truth hand locations in the Classifier-Kinect dataset - (a) initial location of hand in color image (b) invalid depth value in the corresponding location in the depth map (c) a better choice is found, with the superimposed depth map, close to the initial selection but with a valid depth value. 92

44 Heat maps of the errors by MKST. (a)-(c) depict average error with blue being almost no error to red, an error of 14 cm or greater. (d)-(f) depict likelihood of errors with blue being low chance of error to red being the locations of most frequent error. 93

45 Perspective view of cross-sections in signing space showing hand tracking errors of the Microsoft Kinect Skeleton Tracker. Cross-sections are shown separated for clarity. 94

46 Examples of tracking errors by the Microsoft Kinect Skeleton Tracker in the Classifier-Knect dataset. 95

47 Process for generating features normalized based on arm lengths. 98

48 Reverse lookup mechanism to obtain the z coordinate of the hand. The predicted x and y location aid a local search to obtain the z coordinate by averaging values in the depth map. 100

49 Plot of ground truth y coordinates of the left hand against the normalized depth value of the point in the depth map closest to the camera (lower X-axis values represent points closer to the camera in the -1.0 direction). . . 101

50 Comparison of hand locations reported by DDRFR to MKST. (ground truth locations are in green, MKST locations in red and DDRFR hand detection is shown in blue). 105

SUMMARY

Automatically recognizing classifier-based grammatical structures of American Sign Language (ASL) is a challenging problem. Classifiers in ASL utilize surrogate hand shapes for people or "classes" of objects and provide information about their location, movement and appearance. In the past researchers have focused on recognition of finger spelling, isolated signs, facial expressions and interrogative words like WH-questions (e.g. Who, What, Where, and When). Challenging problems such as verification of ASL sentences and classifier-based grammatical structures remain relatively underexplored in the field of ASL recognition.

One application of recognition of classifiers is creating educational games to help young deaf children acquire language skills. Previous work developed CopyCat, an educational ASL game that requires children to engage in a progressively more difficult expressive signing task as they advance through the game.

We have shown that by leveraging context we can use verification, in place of recognition, to boost machine performance for determining if the signed responses in an expressive signing task, like in the CopyCat game, are correct or incorrect. We have demonstrated that the quality of a machine verifier's ability to identify the boundary of the signs can be improved by using a two-pass technique that combines signed input in both forward and reverse directions. Additionally, we have shown that we can reduce CopyCat's dependency on custom manufactured hardware by using an off-the-shelf Microsoft Kinect depth camera to achieve similar verification performance. Finally, we show how we can extend our ability to recognize sign language by leveraging depth maps to develop a method using improved hand detection and hand shape classification to recognize selected classifier-based grammatical structures of ASL.

CHAPTER I

INTRODUCTION

American Sign Language (ASL) is the primary means of communication for deaf and hard of hearing people in the United States [67]. ASL is a visual language with its own grammatical structure that uses hand, facial and body gestures to convey meaning. Computer science research in ASL recognition began in the early 1990s. In the past, researchers have applied a variety of computer vision and pattern recognition techniques [6, 13, 83, 84], some aided with colored gloves [13], data gloves [30, 43], accelerometers [13] and motion capture systems [95], to recognize ASL finger spellings [27, 53, 74], non-inflected isolated signs [33] and in some cases ASL sentences [13]. The bulk of the research has focused on recognition of finger spelling and isolated signs, but more recently there have been works that have examined recognition of facial expressions [66, 58] and interrogative words like WH-questions (e.g. who, what, where, and when) [59]. Two areas in ASL recognition that are relatively underexplored are 1) verification of sign-based grammatical structures consisting of basic signs strung together to form phrases or sentences and 2) classifier-based grammatical structures, that convey information about the behavior of a noun object.

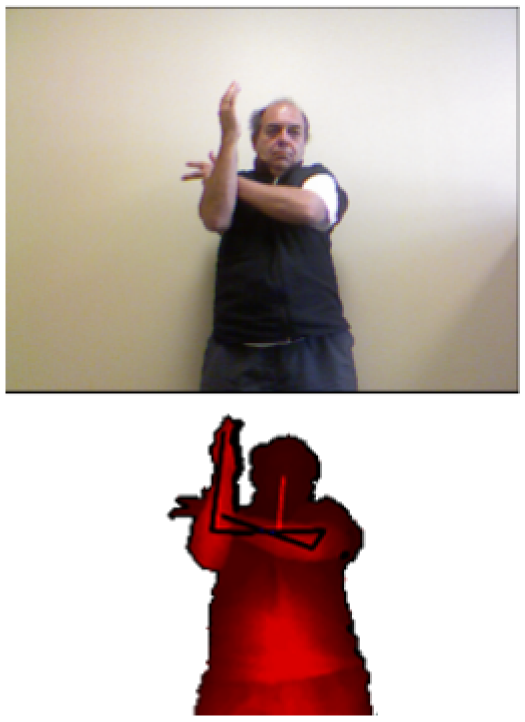
In this dissertation we have shown that by leveraging context we can use verification in place of recognition to boost machine performance for judging the correctness of signed responses in tasks when the required ASL sentence is known beforehand. The advent of inexpensive depth cameras like the Microsoft Kinect has spurred renewed interest among many researchers to employ such cameras for ASL recognition. Previously the high cost, low reliability and difficult calibration issues with depth cameras have discouraged researchers from using them for ASL recognition. We have conducted experiments to show that the Microsoft Kinect can be viable as a hardware platform to collect data to perform ASL recognition. We have shown how we can extend our ability to recognize sign language by leveraging depth maps to develop a method using improved hand detection and handshape classification to recognize selected classifier-based grammatical structures of ASL.

1.1 ASL Classifiers

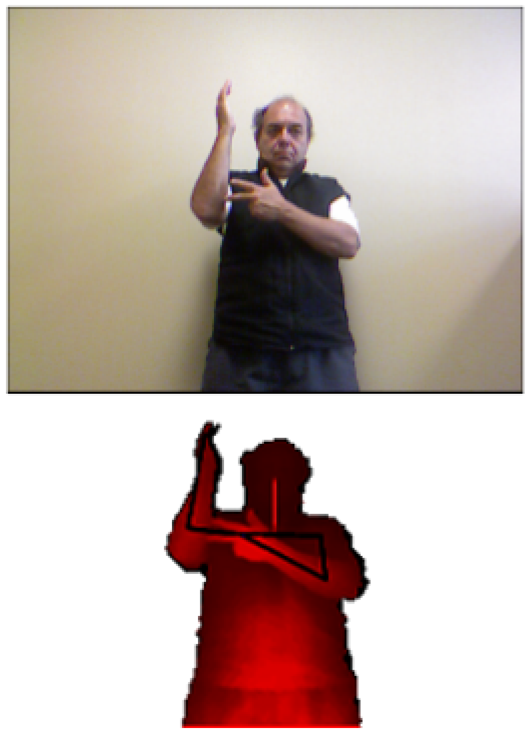
Classifiers in ASL utilize surrogate hand shapes for people or objects and provide information about their location, movement and appearance. If any of these three parameters in the physical construct changes, it conveys a completely new meaning about the person or the object. In contrast, for isolated non-inflected signs a change of location, movement or hand shape results in a new sign (e.g., a change of starting location with the same movement and hand shape in the case of FATHER and MOTHER). The spatial constructions of ASL classifiers take advantage of the fact that visual spatial information is one of the most reliable sensing modality in humans [38]. Studies have shown that deaf signers have enhanced visual spatial processing capabilities in tasks such as mental rotation, visual attention and face recognition [9, 25, 70, 92]. Deafness *per se* is not the cause for this enhanced capability, exposure to a sign language is [71]. An example of classifier use is shown in Figure 1, which shows still images accompanied by corresponding depth maps recorded using a Kinect camera. It demonstrates the use of the TREE classifier and the VEHICLE classifier. It shows three situations, the first being CAR DRIVES BEHIND TREE (Figure 1a), the second being CAR CRASHES INTO TREE (Figure 1b) and the last one shows the CAR DRIVES IN FRONT OF TREE (Figure 1c). The accompanying depth map provides us with enough information to discern the semantic difference between the three situations. The same task becomes extremely challenging if only the RGB image is taken into consideration.

1.2 Thesis Statement

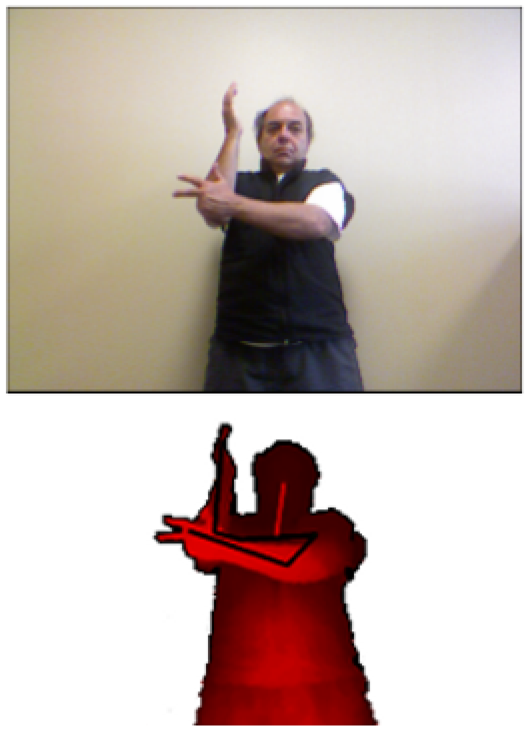
Computer science research in ASL can be broadly classified into the following categories: verification, recognition, translation, generation (synthesis) and natural language processing (NLP). Based on current research we find that there are two distinct tracks researchers follow. The first track has verification and recognition tied in with translation, and the second track ties NLP with generation (or synthesis). ASL synthesis researchers have explored the idea of using computer avatars to depict signing that is generated based on a paragraph of English text, including ASL classifier predicates [39, 40]. However, in ASL



(a) CAR DRIVES BEHIND TREE



(b) CAR CRASHES INTO TREE



(c) CAR DRIVES IN FRONT OF TREE

Figure 1: ASL classifier example with accompanying depth map images recorded with a Kinect camera.

recognition/verification, classifiers are still relatively underexplored.

Thesis statement I hypothesize that we can extend our ability to recognize sign language by leveraging depth maps to develop a method using improved hand detection and handshape classification to recognize selected classifier-based grammatical structures of American Sign Language.

Table 1 summarizes the research contributions of this work. The table provides concise information about the research questions, the initial hypotheses to answer those questions, the methods applied and finally the experimental analysis. The rest of the dissertation is organized as follows. Chapter 2 is related work. In this chapter I introduce CopyCat, an educational ASL game we developed [49] that was designed to help deaf children improve their language abilities. CopyCat requires children to engage in a progressively more difficult expressive signing task, to describe a graphic as they advance through the game. In Chapter 3 the difference between *verification* and *recognition* is explained. We show that by leveraging context we can use verification, in place of recognition, to boost machine performance for determining if the signed responses in the CopyCat game are correct or incorrect. Chapter 4 describes a two-pass technique that combines signed input in both forward and reverse directions to improve the quality of the machine verifier's ability to identify the boundary of the signs. In Chapter 5 we demonstrate how the verification method described in Chapter 3 can be used for ASL sentence verification by replacing the existing CopyCat sensing hardware with a single Kinect camera. Chapter 6 gives details about the method for recognizing selected classifier-based grammatical structures of American Sign Language. Details of recognition pipeline for ASL classifier recognition and the new hand detection approach are also given in this chapter. Chapter 7 describes a method for hand detection using domain driven random forest regression. Chapter 8 is the conclusion and future work.

Table 1: Research contributions towards development of an American Sign Language verification infrastructure and a method to recognize selected classifier-based grammatical structures of American Sign Language.

Contributions	Research Questions	Hypothesis	Method	Data	Analysis	Publications
Collection of a unique dataset of deaf children signing ASL sentences and comparison between automated ASL sentence verification, recognition and human scoring.	1. How does automated ASL verification compare to recognition? 2. How does the automated ASL verifier compare to a human scorer? 3. What signing variations are seen in deaf children?	1. Human scorer has considerably better performance. 2. Using strict grammars will boost verification rates.	Recognition results obtained using part of speech (POS) grammars and verification using strict grammars.	Gwinett dataset and GSD dataset, see Table 3 in Chapter 2.	1. True positive, false positive rate trade off for POS and strict grammars. 2. Compare ground truth with live responses from human scorer.	ICPR 2010 [113], LREC 2010 [15], ICLS 2010 [101]

Continued on next page

Table 1 : continued from previous page

Contributions	**Research Questions**	**Hypothesis**	**Method**	**Data**	**Analysis**	**Publications**
Improved sign segmentation using a two-pass technique that combines signed input in forward and reverse directions.	1. Does the two-pass technique improve ASL sentence verification accuracy? 2. Does the two-pass technique improve sign segmentation?	Both the verification accuracy and sign segmentation will improve.	1. Generate confidence measures in forward and reverse pass. 2. Combine the confidence measures for verification; select the best for segmentation.	Gwinett dataset and GSD dataset, see Table 3 in Chapter 2.	1. Verification results using leave-one-signer-out validation. 2. Using ground truth compute segmentation error. Compare standard approach to the two-pass technique.	CVPRHB 2010 [114]
						Continued on next page

Table 1 : continued from previous page

Contributions	**Research Questions**	**Hypothesis**	**Method**	**Data**	**Analysis**	**Publications**
Investigation of Microsoft Kinect to leverage depth maps and skeleton tracking for ASL sentence verification.	1. How does Kinect compare to the CopyCat Sensor Platform for ASL sentence verification? 2. How does the verification performance with Kinect compare when the signer is seated vs standing?	1. Kinect will perform better than CopyCat Sensor Platform. 2. Skeleton tracking will be poor when signer is seated resulting in poor verification performance.	Use the verification framework from the ICPR 2010 [113] paper that uses strict grammars and likelihood thresholding.	CC-Kinect and CC-Adult datasets, see Table 3 in Chapter 2.	Verification results using leave-one-signer-out validation.	ICMI 2011 [115]

Continued on next page

Table 1 : continued from previous page

Contributions	Research Questions	Hypothesis	Method	Data	Analysis	Publications
Design and collection of a dataset that contains examples of selected ASL classifiers signed by fluent signers.	1. Do the signers agree on the structure of the ASL classifier sentences? 2. Are there variations in the signing of classifier predicates? 3. Are there any fixed patterns in the variations (handedness etc.)?	1. Variations can be expected due to signer's background (fluent vs. native signer). 2. Linguistically acceptable variations exist that do not change the interpretation.	1. Collaborate with sign linguists and fluent/native signers to design the dataset and select interesting classifier interactions that are challenging to automatically recognize but are within the bounds of hardware limitations. 2. Record signing using the Kinect camera. 3. Store RGB images, depth maps and output from the skeleton tracker.	Classifier-Kinect dataset, see Table 3 in Chapter 2.	Prepare the ASL gloss for each sentence in the dataset and manually label the dataset to identify segments based on the gloss. Report statistics on signing variation.	

Continued on next page

Table 1 : continued from previous page

Contributions	Research Questions	Hypothesis	Method	Data	Analysis	Publications
A method for recognition of selected ASL classifiers.	1. Can we identify the ASL classifier? 2. How do we assign nouns to classifiers? 3. Can we identify the ASL classifier predicate?	1. Identify landmarks using motion characteristics of the left and right hand. (Long pause for classifier 1 and rapid acceleration for classifier 2). 2. Classifiers can be identified using hand shape. 3. Assign nouns and identify the classifier predicate using an HMM-based technique.	1. Find where the classifiers occur. 2. Based on the found landmarks, segment the ASL sentence into 3 parts. 3. Recognize handshapes at the landmark points to identify the classifier. 4. Based on the classifiers, recognize, using HMMs, the nouns and the classifier predicate in a reduced search space.	Classifier-Kinect dataset, see Table 3 in Chapter 2. The hand locations reported by the Microsoft Kinect SDK Skeleton Tracker are used.	Recognition accuracy of ASL classifiers, detection of classifier interactions, noun assignment accuracy and overall sentence recognition accuracy.	

Continued on next page

Table 1 : continued from previous page

Contributions	Research Questions	Hypothesis	Method	Data	Analysis	Publications
A dataset of manually labelled hand locations in 75000 video frames of ASL classifier signing data.	1. How does the Microsoft Kinect Skeleton Tracker compare to the ground truth with respect to hand locations? 2. Under what circumstances does the MS Kinect Skeleton Tracker fail more often?	The MS Kinect Skeleton Tracker will likely fail in the following situations: 1. When the hands come close to other parts of the body. 2. When hands come close to or touch each other. 3. When the arms cross.	Develop a labelling tool to label the hand locations for the left and right hands.	Classifier-Kinect dataset, see Table 3 in Chapter 2.	Accuracy of the MS Kinect Skeleton Tracker within an error margin of 5 cm compared to the ground truth. A heatmap to show errors across different locations in the signing space.	

Continued on next page

Table 1 : continued from previous page

Contributions	Research Questions	Hypothesis	Method	Data	Analysis	Publications
A method for hand location prediction in depth images of ASL data.	1. Can we improve accuracy of reported hand locations compared to the Microsoft Kinect SDK Skeleton Tracker? 2. Does having better hand location information improve the recognition accuracy?	1. Using a data driven approach we can train a regressor to predict the location of the left and right hand. 2. Better location information will result in better HMMs, which will boost recognition.	1. Use random forest regression to predict the location of the left and right hand. 2. Conduct supervised training using hand labelled location information. 3. Generate features using depth data in signing space. 4. Train a separate regressor to predict the x and y coordinate of each hand and use a reverse lookup to find the z coordinate.	Classifier-Kinect dataset, see Table 3 in Chapter 2. Ground truth hand location information is obtained by labelling the location of the left and right hand manually by hand in every frame.	Accuracy of the predicted hand locations with an allowed margin of error of 5 cm compared to hand labelled data. Leave-one-out and k-fold cross-validation. Compare with Microsoft Kinect SDK Skeleton Tracker.	

Continued on next page

Table 1 : continued from previous page

Contributions	Research Questions	Hypothesis	Method	Data	Analysis	Publications
A method for robust hand shape recognition in depth images of ASL data.	1. How can we combine color image data with depth data for robust hand shape recognition? 2. What kind of problems occur when finding correspondences between color image pixels and depth image pixels?	1. Segmented hand in the color image can be used as a mask to obtain corresponding depth information. 2. Depth data may not align properly over the color image data. 3. Depth data may be missing for some color pixels that belong to the hand, resulting in "holes".	1. Use skin color and connected components to segment the hand. 2. Use local linear extrapolation to fill "holes." 3. Generate features based on local depth histograms of hand pixels. 4. Use a database lookup approach to classify the ASL classifier handshapes.	Classifier-Kinect dataset, see Table 3 in Chapter 2. Three sets of hand location information are used for testing. The Kinect Skeleton Tracker locations, random forest predictions and ground truth hand locations. For training only the ground truth locations are used.	Accuracy of hand shape recognition with and without extrapolation for each of the three sets of hand locations. Leave-one-out and k-fold cross-validation.	

CHAPTER II

RELATED WORK

Hidden Markov models (HMMs) are popularly used for speech recognition [42, 90] as well as for automatic sign language recognition (ASLR) [7, 13, 96] since they provide a powerful architecture to build statistical models of temporally varying, limited and noisy data. Sign language recognition is a growing research area in the field of gesture recognition. Research on sign language recognition has been performed around the world using many sign languages, including American Sign Language [14, 99], Korean Sign Language [47], Taiwanese Sign Language [52], Chinese Sign Language [29, 32], Japanese Sign Language [80], and German Sign Language [8]. Previous sign language recognition systems have used various kinds of sensors. Starner and Pentland [83] used a single camera compared to Vogler and Metaxas [95] who used a motion capture system. Gao et al.'s Chinese Sign Language recognition system [30] uses data gloves and position trackers whereas Hernandez et al. use hand crafted sensor networks [37]. Brashear et al. [14] have shown in the past that combining sensor data from cameras and accelerometers can result in significantly improved ASL recognition.

A variety of computer vision features are used in ASLR. Ong lists the following categories in his survey paper [68]: two-dimensional segmentation, two-dimensional moment-based, motion vectors, three-dimensional hand positions and three dimensional hand orientations. In addition to these common features we would add: two-dimensional head tracking [26, 56, 113], three-dimensional head orientation [98] and motion templates [20, 107, 108].

2.1 Language modeling in ASL recognition

Researchers differ greatly in their approach to modeling basic units of signed languages. The simultaneous nature of meaningful left hand, right hand, and head gesture in sign languages poses a challenge to many of the sequential techniques used in speech recognition [93]. Many researchers choose to use the sign as a base unit of modeling [13, 83], while others attempt to use a structure similar to phonemes to create models [26, 80]. Vogler and Metaxus have proposed several techniques for handling simultaneous phonemes using the Movement-Hold linguistics model and parallel HMMs [96, 97]. Bowden et al. use a two-tiered approach that is designed to learn from small amounts of data and classifies the *tab-dez-sig* features from Stokoe's phonology [87] and passes the results to a Markov chain [26]. *Tab* refers to "tabula" or sign location, *dez* means "designator" or handshape and orientation, and *sig* means "signification" or motion and action.

2.2 ASL verification and sign spotting

Sign language verification remains relatively underexplored compared to recognition, particularly at the sentence level. The SignTutor system developed by Aran et al. [1] verifies isolated signs in a two stage process. In the first stage a general HMM is used to perform recognition and select one candidate class and a cluster of signs that were confused with this class during a previous cross-validation process. In the second stage the earlier likelihood is combined with the likelihood obtained from a more dedicated model to make the final decision. There has also been some work performed on sign spotting [65, 105]. Yang, Sclaroff, and Lee [105] have proposed a method to spot signs from a continuous stream of data using a conditional random field (CRF) based threshold model. Their method first detects and then recognizes signing patterns and is able to spot signs from continuous data with 87.0% accuracy and to recognize isolated signs with 93.5% accuracy.

2.3 Speech utterance verification

Confidence measures have been used in speech recognition for nearly two decades [55, 91, 111]. Rose et al. [78] were the first to formulate the utterance verification problem as a statistical hypothesis test and proposed the use of the likelihood ratio test. Sukkar and Lee [90] expanded on this work and presented a framework for discriminative utterance verification. Jiang [42] has provided an extensive survey of several works in speech recognition that employ confidence measures.

2.4 Hand tracking and detection

Methods used for locating and tracking hands can be categorized according to the capture system. In the context of sign or gesture recognition, these can be sensor-based, appearance-based or depth-based.

2.4.1 Sensor-based methods

Sensor-based systems, like data gloves, utilize different types of wearable sensors to capture hand motion and position. Such systems can comprise of optical sensors, flexion sensors, accelerometers, magnetic sensors, inertia measurement units or a combination of different transducers [17, 37, 69].

Despite their high accuracy, sensor-based systems are usually expensive and require regular calibration to guarantee optimum functionality. In addition, most of these systems require sensors to be attached on the users arm or hand, which might restrict the user's movement and cause inconvenience.

2.4.2 Appearance-based methods

Appearance-based systems use RGB or color cameras to perform hand detection and tracking. Common methods for hand localization using appearance are skin color detection [34, 41, 62, 85], colored glove detection [2, 12, 48] and template matching [10, 51].

Wang and Popović [100] used a glove that has ten full saturated colors distributed randomly in 20 patches. Using a single RGB camera, hand position was defined after filtering the image in the HSV domain by setting a high threshold on the saturation level. The pattern of multiple colored markers helped to classify hand shapes by matching a low resolution version of the captured image with samples of possible hand shapes.

Generally, appearance-based methods are affected by illumination, and their accuracy can drop due to changes in lighting conditions. Hence, they may require regular calibration to operate robustly [35, 64].

2.4.3 Depth-based methods

Depth maps can be generated by a number of algorithms and varying special camera configurations. Stereo camera rigs, RADAR, LIDAR, structured light techniques and sonar have all been used to generate depth maps [82]. Time-of-flight cameras are popularly used in computer vision and robotics to generate depth maps at a high frame rate [88]. Until now commercially available depth camera systems were expensive and out of reach for most researchers and developers in the fields of computer vision and gesture recognition. The availability of inexpensive off-the-shelf depth cameras such as Microsoft Kinect, Intel SENZ3D or ASUS Xtion have allowed researchers to extensively investigate and implement techniques for hand tracking and localization using depth information. Suarez and Murphy, in their survey about gesture recognition using depth images listed 13 methods developed by researchers for hand localization [89].

Common methods used for hand detection in depth images are depth thresholding [31, 60, 63, 77], and region growing [19, 24]. Despite its simplicity, depth thresholding may not be suitable for real world applications because it assumes that hand position is always within a given proximity from the camera or a reference point. Region growing is able to extract hand location assuming it is not in contact with other body parts or objects in the scene. The latter case occurs often in ASL signs, which makes region growing an undesirable hand locating technique for ASLR.

Several researchers have developed other techniques to perform hand tracking from depth information. Stenger et al. [86], Frati et al. [31] and Park et al. [72] used Kalman filters, which performs a recursive least squares estimate to define hand trajectory in sets of subsequent frames. Mean shift and continuous adaptive mean shift algorithms have been used by Yoo et al. [109], Yang et al. [104], Chen et al. [19], and Keskin et al. [45]. These algorithms approximate the hand velocity and direction using gradient descent in an iterative fashion.

Shotton et al. [81] have implemented a real-time human pose recognizer. The recognizer is able to provide an estimate for the location of each body appendage. The authors used depth comparison features generated for each pixel in the image. These features were selected to train a random forest with 300,000 depth images. The recognizer generates decisions for each pixel, defining if it belongs to a human body or not, and to which body part it belongs to. Then it applies mean shift with a weighted Gaussian kernel on the generated results to form a local mode-nding approach to estimate joint positions. This method for human body pose estimation has been used by the Microsoft Kinect skeleton tracker.

2.5 3D hand pose estimation using depth data

Before depth cameras like the Kinect became popular previous works on 3D hand pose estimation have used model-based methods [22, 76, 86], nearest neighbor matching in a database of poses [4, 5, 3, 100] and even multiple cameras to determine the hand pose [21].

Database matching and model-based methods continue to be popular with researchers working on 3D hand pose estimation using depth cameras. Doliotis et al. employ a technique whereby they match the segmented hand, obtained using depth data from the Kinect sensor, against a database consisting of synthetically generated hand images to create a ranked list of shape class, 3D pose orientation and full hand configuration parameters [23]. Xu and Cheng propose a three-step pipeline that first finds the in-plane orientation and the 3D location of the hand followed by generation of candidate 3D poses from a dataset of synthetic depth images of the hand and then finally perform an optimization to find the best fit [103]. Keskin et al. use pixel-based random forest classification to assign each pixel to a hand part in a 21-part hand model and use the classification results to estimate the joint positions [44]. They report results of 99% accuracy on an ASL digits dataset.

Although the above mentioned works that use depth cameras for hand pose estimation have reported promising results, a closer look at the test datasets reveals that data was captured at close distances from the camera where the depth resolution is at the maximum. The Microsoft Kinect's depth estimation apparatus consists of an infra-red (IR) light projector and a standard CMOS camera to capture the image of the IR pattern. The distortion of the IR pattern is used to calculate depth maps, which have a per-pixel depth resolution of 2 mm within 1 m from the camera, 1 cm at 2 m distance and 2.5 cm at a 3 m distance

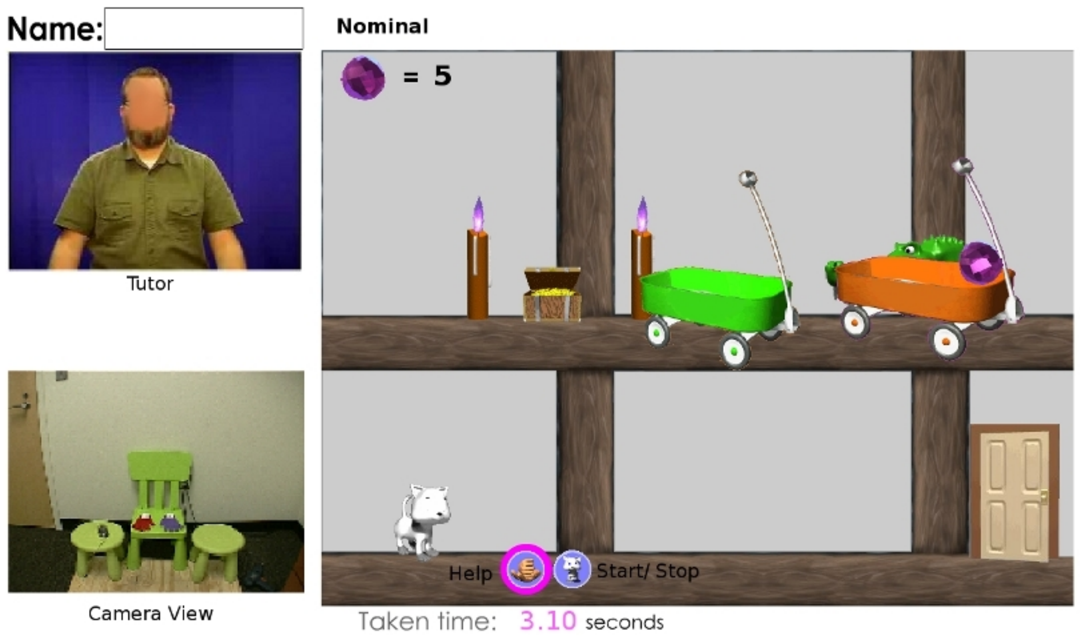
[46]. In real-world ASLR applications the signer will have to stand further away, at least 5ft, from the camera in order to capture the full signing space. At these distances (1.5m) the depth resolution begins to degrade, and we will likely see noisier data making it challenging to do accurate hand pose estimation.

2.6 CopyCat Project

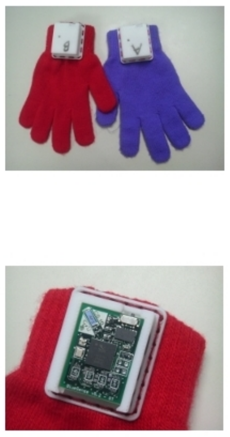
Ninety-five percent of deaf children are born to hearing parents, many of whom do not know American Sign Language [61]. Studies have shown that starting sign language learning at an early age is critical to have a long-lasting advantages [57]. The CopyCat project was initiated to improve ASL exposure for such children, which is often limited to school [36, 49]. The goal of the game was to improve the word span and short term memory skills of the deaf children . The CopyCat project consists of a suite of educational adventure games for deaf children that facilitate interaction with the computer using gesture recognition technology and serve as a practice tool for children to improve their language skills [13, 113]. A core component of the game is determining whether the required ASL sentence has been signed correctly.

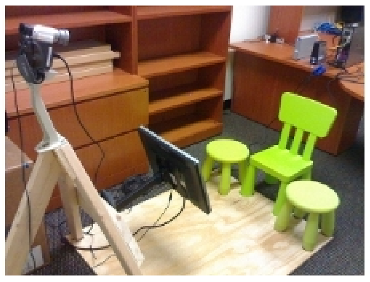
2.6.1 The CopyCat game

The software for the games is written using Flash and runs within a web browser. The children are required to wear gloves of a different color (red and purple) on each hand. One 3-axis accelerometer is attached to each glove (Figure 2b). The camera, computer monitor, and the chair to seat the child are positioned as shown in Figure 2c. A screen shot of the Flash game is shown in Figure 2a. The child has to express the scenario presented in the graphic using ASL to control the character in the game. A tutorial video, which demonstrates the correct ASL sentence, can be accessed using the help button if the child has difficulty in understanding the scene. To start signing the child clicks the Start/Stop button; he clicks the button again to stop signing. The *Start* and *Stop* signals from the mouse clicks provide information for the temporal segmentation of the vision and accelerometer data streams. Video frames are captured from a single IEEE1394 camera at a resolution of 720x480 at 20 fps. The camera view captured by the IEEE1394 camera is



(a)





(b) (c) (d)

Figure 2: CopyCat apparatus: (a) Flash game (b) gloves with accelerometers (c) kiosk (d) camera view

Table 2: CopyCat game vocabulary

nouns		prepositions	adjectives
subject	object		
alligator	bed	behind	black
cat	box	in	blue
snake	chair	on	green
spider	flowers	under	orange
	wagon		white
	wall		

shown in Figure 2d. Two wireless bluetooth 3-axis accelerometers, with a range of -2g to +2g, stream data at 40 Hz.

Currently, CopyCat supports a 19 sign vocabulary (Table 2) with 60 different sentences. Many of the more sophisticated ASL linguistic constructions, such as facial gestures, are not included in game interactions in order to reduce the complexity of the game engine. Three-sign, four-sign and five-sign sentences appear at different levels of difficulty in the game. Each sentence follows the sequential sign-based grammar:

[adjective1] subject preposition [adjective2] object

Both adjectives are optional in a three-sign sentence whereas only the first adjective is optional in a four-sign sentence. Both adjectives are required for the five-sign sentence. The four-sign sentence corresponding to the graphic in Figure 2a is *ALLIGATOR BEHIND ORANGE WAGON.*

2.6.2 Hardware Changes

From this point forward, for clarity, we will refer to the hardware setup that includes wrist mounted wireless bluetooth accelerometers and a single Firewire camera as the *CopyCat Sensor Platform.* Although, as we will show later, we have been able to extract good recognition performance with this platform, it has disadvantages and presents some practical challenges to the end user. Since there is more than one sensor involved, it increases the number of failure points. Moreover, the computer vision algorithms used for eye tracking and hand tracking are impacted by environmental factors such as lighting conditions.

The CopyCat Sensor Platform requires a kiosk-like setup that adds additional costs and maintenance efforts. Accelerometer sampling errors can also occur that become further exaggerated when the batteries have not been charged adequately. Circumstances may arise when controlled lighting is required so that the tracking component performs accurately. All of these factors make it necessary for a researcher to be present to run the CopyCat system during deployment. One of the major drawbacks of the CopyCat Sensor Platform is that it requires the signer to wear gloves with accelerometers strapped to the wrist to enable the capture of gestural information, which impacts comfort level.

In the past, a stereo camera was considered for replacing the CopyCat Sensor Platform; however, the high cost of commercially available stereo cameras was a significant deterrent for adoption. Moreover, a stereo camera is again plagued by environmental factors like lighting conditions. The Microsoft Kinect [116], which estimates depth by projecting an infrared light pattern, is less affected by environmental factors, resulting in a very reliable depth map output. More importantly, it provides a simple plug-n-play interface that even individuals without a technical background will be able to operate with ease. The commercial version of the Kinect costs a mere $250. All of the above factors combined make the Kinect a promising alternative to the CopyCat Sensor Platform. We will refer to the new hardware platform, which consists of a computer connected to the Kinect via a USB cable, as the *CopyCat Kinect Platform.*

2.6.3 CopyCat Datasets

Over the past few years several data collection efforts have taken place to collect sign language data using both the CopyCat Sensor Platform and the CopyCat Kinect Platform. Data, in the form of ASL sentences, was collected from a wide range of signers that includes deaf children, fluent adult signers and hearing adults with varying levels of signing skills. Table 3 gives an overview of all the datasets that have been collected. The *Gwinnett* dataset (collected at the Minor Elementary School, part of the Gwinnett County Public Schools), *GSD* dataset (collected at the Georgia School for the Deaf), *CC-Adult*, *CC-Kinect-seated* and *CC-Kinect-standing* datasets (collected at Georgia Tech) all contain ASL sentences from the CopyCat game. The Classifier-Kinect dataset contains signed examples of ASL classifier use. For more details on the design and collection of this dataset please refer to Chapter 6. Prior to these datasets there have been other datasets that were collected for the CopyCat project during pilot deployments [13].

In the Gwinnett dataset the method of collection was Wizard of Oz. In other words, since there was no recognition engine that was developed yet, the CopyCat game responses were mimicked by a human. The human scorer, who is an ASL linguist, made the decision whether or not the child was signing the required ASL sentence correctly. This input from the human scorer was relayed back to the game to generate the appropriate response in the user interface. The children, unaware of the human intervention, believed that the computer was understanding their signing. The Wizard of Oz method facilitated the collection of a large dataset that was utilized to build user-independent models of the signs in the CopyCat game vocabulary. In the CC-Adult dataset, the Wizard of Oz method was again used. However, the signers were all hearing adults with varying levels of signing skills ranging from beginner to expert. For the GSD dataset, a fully-functional game engine was deployed eliminating the need for a human scorer. The CC-Kinect-seated and CC-Kinect-standing datasets were collected using the new CopyCat Kinect Platform. The sentences were the same as the ones from the CopyCat game, but, instead of playing the CopyCat game, the signers viewed a video of the sentence and repeated the signing back. The CC-Kinect-seated had signers seated similar to the earlier datasets, while in CC-Kinect-standing the signers were standing. The responses were recorded with the Kinect camera. For the Classifier-Kinect dataset the signer looked at English text that appeared on the screen and then translated the text to ASL making use of ASL classifiers in the sentence.

2.6.4 ASL sentence recognition with Gwinnett dataset

ASL sentence recognition for the CopyCat project was addressed in Brashear's Ph.D. thesis [11]. The main focus was the creation of an ontology that described the contents of the Gwinnett dataset. The ontology was used to improve automatic sign language recognition and to add customized language processing capabilities. Experiments were conducted using

Table 3: CopyCat datasets collected for this dissertation

Dataset	**#Signers**	**Ages**	**Skill**	**Method of Collection**	**Total sentences**
Gwinnett	18	6 - 11	deaf children with basic signing skills.	Wizard of Oz.	1830
GSD	11	6 - 11	deaf children with basic signing skills.	Live game responses.	1432
CC-Adult	8	22 - 58	Hearing adults with beginner to expert level signing.	Intentional correct responses (Wizard of Oz).	820
CC-Kinect-seated	8	22 - 58	Hearing adults with beginner to expert level signing.	Sentence repetition.	555
CC-Kinect-standing	3	22 - 58	Hearing adults with beginner to expert level signing.	Sentence repetition.	155
Classifier-Kinect	5	22 - 35	Fluent deaf signers with expert level signing.	English text to sign translation.	492

1192 sentences from the Gwinnett dataset. User-independent models of the signs from the game, the vocabulary from out-of-game communication from the children, and the disfluencies discovered in the signing samples were created using hidden Markov models (HMMs) for recognition of hand-based American Sign Language gestures. Using N-best recognition, word accuracy of 87.10% and a sentence correctness of 48.53% was achieved. Finally, a multiple hypothesis language parser classified samples from the game as correct or incorrect. The combination of the N-best recognition and the classifier judged the sentence correctness at a rate of 75.63%.

CHAPTER III

VERIFICATION OF ASL SENTENCES

In this chapter, we will demonstrate that to boost the performance of the CopyCat game engine we can use *verification* directly in place of *recognition* to determine if the signer has signed the ASL sentence correctly. We explain the difference between verification and recognition using an example from the CopyCat game language. Figure 3 shows a word lattice for the game language. We can construct several sentences by stringing together signs from the left to the right. If we make the adjectives optional there are a total of 3440 three-, four- or five-sign sentences that can be constructed. However, the CopyCat game only uses 60 of those sentences, one of which is highlighted in Figure 3. If we employ recognition then the search space is large, equal to 3440 sentences, but we could reduce the search space by limiting it to the 60 sentences in the game. We can further limit the search space by exploiting the game context. In each step of the game, the game engine has knowledge of the current sentence that the signer has to sign. So, instead of searching through the entire word lattice or matching against 60 game sentences, we can "verify" if the sentence signed by the signer is the same as the one required at the current step. This process is somewhat complicated as four or five sign phrases are accepted when only three signs are sufficient. As long as the children signed a phrase consistent with the graphical world provided the sentence is verified as correct. Formally, the process of verification proceeds as follows. First, the *null hypothesis* is established, which is the game sentence that is being tested in the current step. Second, the signing data is collected by a push-to-sign method whereby the signer clicks on a button to begin signing and clicks the button again to stop signing. Finally, the input is aligned with the hypothesis sentence, and a confidence measure is applied to the individual signs to determine if a sign is rejected or retained. If all signs in the input sentence are retained then the ASL sentence is *verified*.

The verification experiments in this chapter were conducted on the Gwinnett dataset.

Adjectives	Subjects	Prepositions	Adjectives	Objects
black	alligator	behind	black	bed
blue	cat	in	blue	box
green	snake	on	green	chair
orange	spider	under	orange	flowers
white			white	wagon
				wall

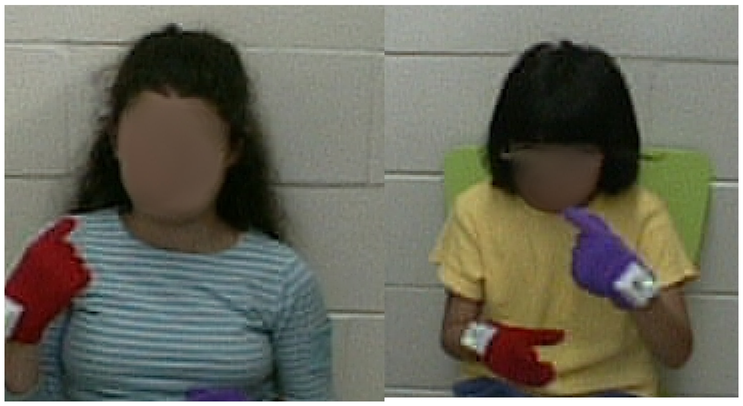
Figure 3: Word lattice for the CopyCat game language that follows a sequential sign-based grammatical structure.

This dataset contains signing data obtained from 18 deaf children from the ages of 6 - 11 years. In this age group we are likely to observe variation in signing among children based on hand dominance as well as linguistically accepted variations of signs. The younger children usually have yet to establish their dominant hand and so mixed (both right and left hand) signing is observed (see Figure 4). Based on the analysis of the Gwinnett dataset the vocabulary size was expanded from 19 signs to 48 signs to include all variations (left and right handed variants, one and two handed versions and other sign specific variations). The variations slightly increase the scope of the verification task for the CopyCat game. Now, verification involves matching the signed input against one path through the word lattice while accounting for all the possible variations for the signs in that path alone.

3.1 Experimental Setup

The training data was collected using a Wizard of Oz version of the game in which an external observer played the role of the ASL verifier. We chose 1204 sentence examples of the Gwinnett dataset consisting of signed ASL sentences from 11 deaf children playing the game during the deployment of CopyCat. This set included 894 correctly signed sentences and 310 incorrectly signed sentences. All of the examples contain three-, four-, or five-sign sentences.

Both vision and accelerometer features are combined to form feature vectors that are


(a) GREEN

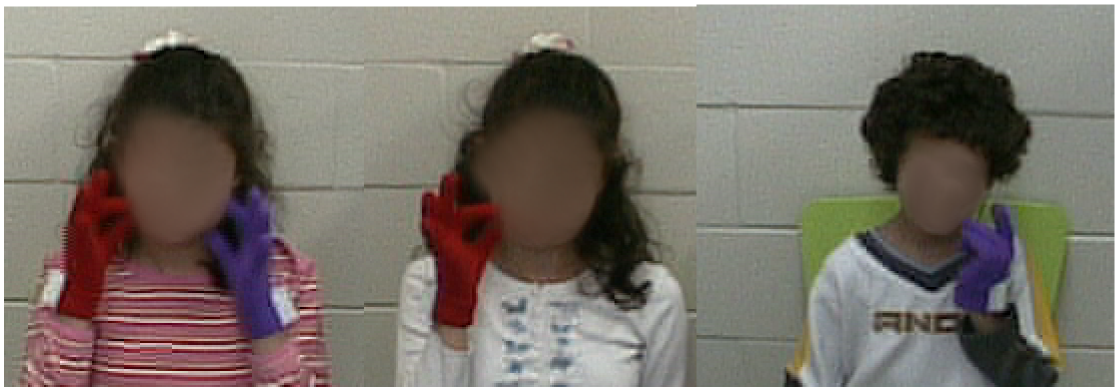

(b) CAT

Figure 4: Signing variations for the ASL signs GREEN and CAT observed in the Gwinnett dataset.

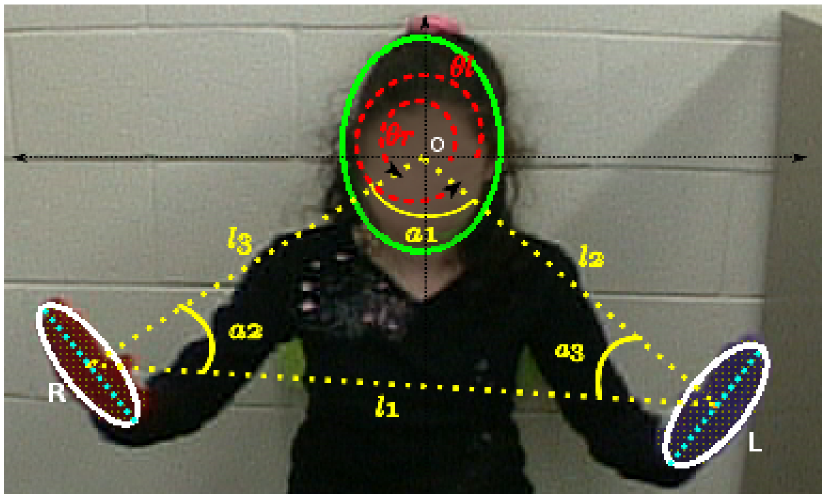


Figure 5: Diagrammatic representation of features extracted using computer vision techniques superimposed on the image of a signer.

used to train left-to-right four-state HMMs with one skip state for each sign. HSV histograms are used to track colored gloves on each hand. Blob features include change in X and Y directions (d_x and d_y), blob size, length of major and minor axes, eccentricity, orientation of the major axis, and change in the major axis orientation in the direction of rotation [13].

In addition to blob features, hand shape features are computed by performing principal component analysis (PCA) on concatenated histograms of the dominant hue (blue and red for purple and red gloves respectively), obtained from a 4x4 grid of the extracted hand region. Assuming a fixed position of the light source, which was the case during the game deployment, this feature allows us to distinguish hand shapes based on shading information. We use head tracking to compute pose features that include grid quantized (x,y) positions of the hands relative to the head position; the angles a_1, a_2, a_3 formed by the triangle defined by the points O, R and L ($\triangle$ORL); the normalized lengths l_1, l_2, l_3 of the sides of $\triangle$ORL; and the angles θ_l and θ_r formed by OL and OR respectively in Figure 5. Finally, we add acceleration values and the frequency power spectrum from each axis of the two accelerometers to the feature vector. The summary of all the types of features is listed in Table 4.

3.2 Verification

3.2.1 Human Performance

To collect data for training and testing, an ASL linguist played the role of a human scorer. The children played the CopyCat game thinking that the computer was validating their signing. Initially, scorer labels were used as ground truth, but upon manual inspection of the signed ASL sentences it was found that the human scorer had made mistakes. The scorer had a true positive rate of 98% and, surprisingly, had a high false positive rate of 31%, achieving an overall accuracy of 90%.

3.2.2 Machine Performance

Within the dataset, children exhibited several acceptable variations for each sign. In most cases the variations depended on whether the child was right- or left-hand dominant as in

Table 4: Computer vision and accelerometer features extracted from the signing data.

Type	Description
Blob	second moment shape descriptors (length of major and minor axes, eccentricity, orientation of major axis)
Hand Shape	shading based fatures obtained by performing PCA on concatenated histograms of V (from HSV) from a 4x4 grid of the extracted hand region.
2D Image Motion	d_x and d_y of the blob center
Pose (2D geometry)	angle formed between the blob center and the horizontal passing through the midpoint between the eyes
Acceleration	x, y & z acceleration values and frequency domain representation of each axis.

the case of GREEN in Figure 4a, but in some cases additional variations occurred. One example is a two-handed version of CAT shown in Figure 4b. Each ASL sentence in the dataset was manually labeled by specifying the segmentation for the signs, and each variant of the sign was given a unique class name. The size of the vocabulary grew from 19 signs to 48 signs.

Hidden Markov models (HMMs) were trained for the 48 classes signs labeled using correctly signed sentences. In addition to the sign models, eight transition models representing transitions between grammar elements of the game language were trained. We used the Georgia Tech Gesture Recognition Toolkit (GT2K) [102] for training and testing. Cross validation was performed by using a randomly chosen 90% of the data for training and the remaining 10% for independent testing. The trained models were then evaluated on the incorrectly signed data to determine the false positive rate. The process was repeated 50 times, and the results were averaged. Additionally, up to two Gaussian mixture components were used for the training process. For verification a strict grammar is used, and a common rejection threshold is applied on the normalized log likelihood score of each sign.

Table 5: Average sentence verification accuracy on the Gwinnett dataset from 50 cross validation trials using a 90-10 split.

	Verification (%)			Recognition (%)		
	TP	FP	Acc.	TP	FP	Acc.
M=1	86.0	38.0	79.8	54.0	3.0	64.9
M=2	88.6	37.2	82.0	56.8	3.3	67.0

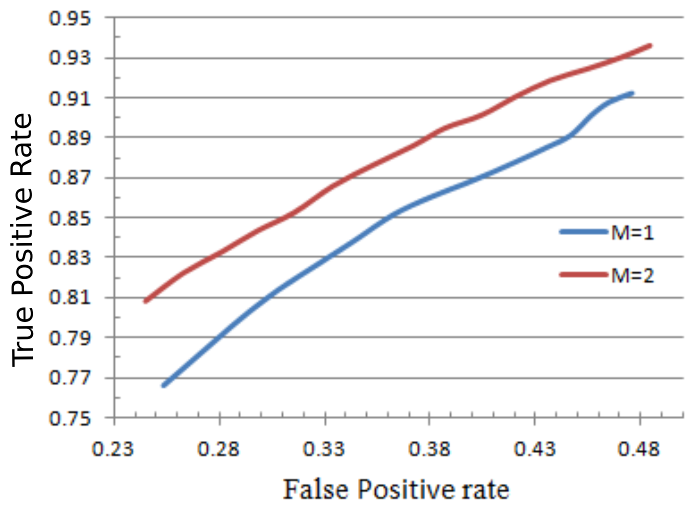


Figure 6: The ROC curve for sentence verification on the Gwinnett dataset using up to two gaussian mixture components in the HMMs.

If the likelihood score for each sign passed the thresold, then the sentence is verified as correct, but if the likelihood score of even one sign falls below the threshold the sentence is incorrect. The value of the threshold was varied in order to pick a suitable value for the online verification system.

Table 5 shows cross validation results for sentence verification using 48 signs. M denotes the number of Gaussian mixture components. In the table the value of the log likelihood threshold (T) for verification is the median value (T=-456) of the range [-470, -442] that was used in the experimentation. An initial experiment using the original 19 sign tokens yielded a sentence verification accuracy of only 60%, highlighting the importance of identifying all signing variations and treating them as separate tokens.

For reference we have included the results of recognition using a context-free grammar.

We can clearly see that for verification there is a significant gain in the true positive (TP) rate. Although false positive (FP) rates are high, verification has higher overall accuracy. We can further boost our accuracy rates by using two mixture components, however significant over-fitting was observed with three or more mixture components. Figure 6 shows the ROC curve for verification obtained by varying the value of the log likelihood threshold (T) from -470 to -442 in steps of 2. The highest overall accuracy of 82.8% (TP=93.6%) is achieved at T=-470 with M=2. However the high false positive rate (48.4%) is not favorable from the viewpoint of using the CopyCat game for educational purposes. We can tune the values of T and M to find a better optimal point. By choosing an alternative value for T (=-454) and with M=2 a significantly lower false positive rate of 35.3% can be achieved while keeping the overall accuracy high at 81.7% (TP=87.7%), which is within 1% of the highest overall accuracy (82.8%).

3.3 Live deployment of CopyCat

The CopyCat system was deployed at the Georgia School for the Deaf with the live version of the sentence verifier, which achieved 82% sentence recognition accuracy on the Gwinnett dataset (see Table 5). The protocol for the study was similar to the Wizard of Oz study in [15]. A control group of children went through regular classroom activity whereas for the experimental group, 45 minutes of classroom activity was substitued with CopyCat game play. Tests were conducted before and after the two week period when the game was installed in the classroom. For the receptive language test the children are shown a video of signing and asked to manipulate figurines to show what they understood from the video. In the expressive language test, the children are shown a video of an actor manipulating figurines and the children are required to express the content in ASL. In the sentence repetition test the children repeat signing that they watched on video. Results show that on average the children who played the game almost doubled their test scores. In comparison, a control group either showed minor improvement or showed no improvement in the test scores. The results are shown in Figure 7.

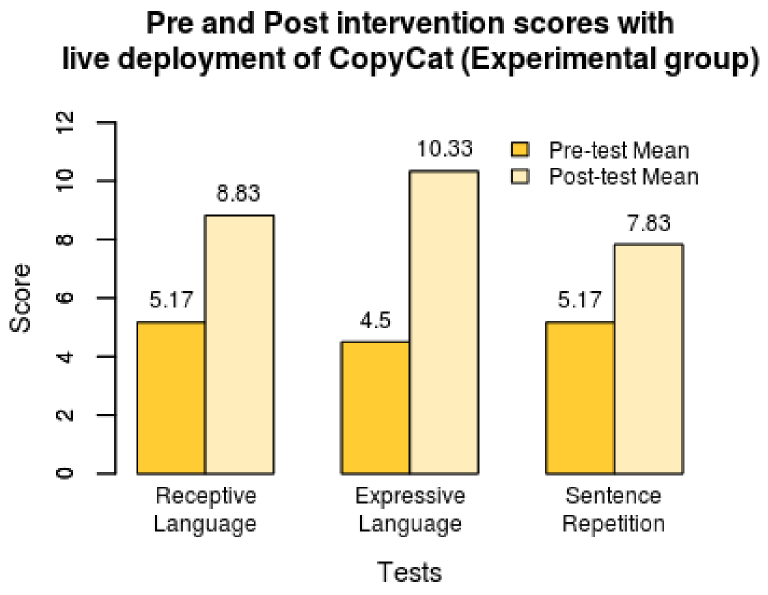


Figure 7: Pre and Post test scores showing educational effect of the CopyCat game in a deployment using a live sentence verifier.

3.4 Summary

In this chapter we have introduced the task of ASL sentence verification. We have shown that by expanding the vocabulary of the CopyCat game to include the signing variations observed in the Gwinnett dataset, the ASL sentence verification accuracy can be improved significantly. Experimental results comparing the automated ASL verifier to the ASL recognizer and a human scorer were published in the proceeding of the 20th International Conference on Pattern Recognition (ICPR 2010) [113]. Details about the corpus of ASL sentences from the CopyCat game signed by deaf children were presented at the 4th Workshop on the Representation and Processing of Sign Languages: Corpora and Sign Language Technologies and appeared in the proceedings of the seventh international conference on Language Resources and Evaluation (LREC 2010) [15].

Findings from a study conducted with deaf children playing the machine verifier version of the CopyCat game show that 1) the current system can verify ASL sentences in real-time but is not as accurate as a human scorer 2) the deaf children who played the CopyCat game got statistically significant educational benefits compared to a control group. Details

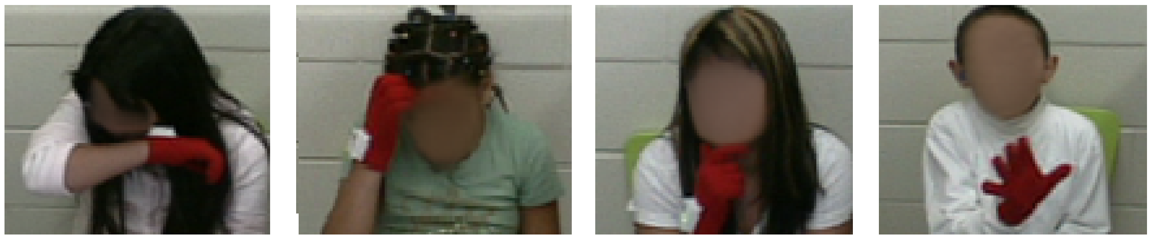


Figure 8: Signing disfluencies that were observed in the Gwinnett dataset: sneezing, scratching, thinking, and interjecting.

about the study and the educational effects of the CopyCat game were published in the proceedings of the 9th International Conference of the Learning Sciences (ICLS 2010) [101]. Additionally, these results are validated in a deployment done at the Georgia School for the Deaf, where the the live sentence verifier was used and the CopyCat game still provided educational benefits to the children playing the game (Figure 7).

A drawback of the technique for sentence verification described in this chapter is that the rejection threshold for the online verifier must be selected manually. Chapter 5 discusses a method for automatically computing a unique threshold for each sign class. One issue that has not been addressed in this chapter, but has been addressed in Brashear's Ph.D. thesis [11], is disfluencies in signing, examples of which are shown in Figure 8.

CHAPTER IV

A TWO-PASS TECHNIQUE FOR IMPROVED VERIFICATION PERFORMANCE

For the task of verification of ASL sentences, besides determining if the signed response is correct or incorrect, it is important that we correctly segment the signs as well. Having accurate boundary information is crucial from the future perspective of providing feedback to signer about exact location of the errors that occur in signing. Precise sign segmentation information will enable us to improve the current version of the CoyCat game by incorporating appropriate fine-grained feedback information, which will allow the signer to examine the parts of the ASL sentence that were signed incorrectly in addition to the coarse-grained response indicating whether the entire ASL sentence is signed correctly or not.

In this chapter we employ an approach to generate an additional confidence measure that helps us make a more informed decision about 1) accepting or rejecting a sign and 2) choosing between two alternative segmentations of the same sign. We introduce the notion of using *reversed* signing (i.e., reversing the temporal order of the signed input) in addition to the regular input. For the rest of the paper we will refer to the HMM training/testing using the regular input (the standard approach) as the *forward pass* and the training/testing process using the temporally reversed input as the *reverse pass*.

My initial motivation for employing the reverse pass was to use it solely as an alternative to forward pass in order to avoid false starts made by deaf children playing the CopyCat game. The Gwinett dataset contains many examples where children make false starts committing several mistakes at the beginning of the sentence but ultimately went on to sign the correct sentence. For example, in response to the graphic in Figure 2a of Chapter 2, a child might sign *SNAKE BEHIND SPIDER ALLIGATOR BEHIND ORANGE WAGON*. The correct response would be *ALLIGATOR BEHIND ORANGE WAGON* without the initial gibberish. The first three signs are incorrect, but these could be rejected easily if we use a

WAGON ORANGE BEHIND ALLIGATOR

(a) Unmodified reverse grammar

WAGON ORANGE BEHIND ALLIGATOR [<ANY>]

(b) Expanded reverse grammar to accommodate false starts

Figure 9: Strict sign-based reverse grammars for the graphic in Figure 2a. a) unmodified b) expanded to accommodate false starts.

wild card in the restricted grammar of the *reverse pass*. Figure 9 shows the standard and the expanded restricted grammars for the reverse pass. The expanded restricted grammar contains the wild card [<ANY>], where "ANY" denotes any sign in the vocabulary including a garbage model that models out-of-vocabulary gestures. The angle brackets denote one or more repetitions and the square brackets denote optional items [110].

Besides the advantage gained by using reverse pass in handling false starts, we hypothesize that we can improve verification performance by combining confidence measures computed in the reverse and forward passes. We have experimented with two types of confidence measures, namely the *normalized likelihood score* and the *log likelihood ratio* (LLR). The LLR test is employed as a typical solution to speech *utterance verification (UV)* formulated as a statistical hypothesis test [42, 78, 90]. If an utterance U has been recognized as a word W, then the LLR is given as follows [42, 112]:

$$LLR = \frac{p(U|H_0)}{p(U|H_1)} \tag{1}$$

where:

H_0 (*null hypothesis*): U is truly a representation of W

H_1 (*alternative hypothesis*): U is a representation of something other than W

The null hypothesis H_0 is accepted if LLR $> \beta$, where β is the critical threshold.

Verification performance in the standard *forward pass* method is compared to the new method that combines the confidence measures from the forward pass and the reverse pass. To summarize the contributions:

1. We introduce the notion of using temporally reversed signing input to generate an additional reverse pass confidence measure that, when combined with the forward pass confidence measure, aids in making a more informed decision about (i) accepting

or rejecting a sign and (ii) choosing between two segmentations of the same sign.

2. We will show that combining forward pass and reverse pass confidence measures can in some cases lead to a gain in verification performance; more importantly we can improve the accuracy of sign segmentation.

4.1 Verification

For a given test ASL sentence $\boldsymbol{P}$, verification is performed independently on each sign s_1, s_2, ... s_n of the sentence. If all the signs have passed the verification criteria then we say that $\boldsymbol{P}$ has been verified. We have used two methods to compute confidence measures to define our verification criteria. First, we use the *normalized likelihood score.* We will refer to this as CM^1 and is defined as follows:

$$CM^1 = \log[L(O_i|s_i)] - (\mu_i - \gamma\sigma_i) \tag{2}$$

where:

$\log[L(O_i|s_i)]$	is the normalized log-likelihood score of observation O_i given sign s_i
μ_i	is the mean of $\log[L(O_i|s_i)]$ obtained from training
σ_i	is the standard deviation of $\log[L(O_i|s_i)]$ obtained from training
γ	is a parameter to scale σ_i

The first verification criteria is:

$$V^{(1)} : CM^1 > 0 \tag{3}$$

Second, we use the *log-likelihood ratio* as another confidence measure; we will refer to this as CM^2. For CM^2 the notation used in Equation 7 of Sukkar and Lee [90] is followed. If we compare Equation 1 and 4 we can see that $L(O_i|s_i)$ models $p(U|H_0)$ and the geometric mean of the second term models $p(U|H_1)$. The geometric mean can viewed as a measure of the likelihood of an ***antisign*** model of s_i [90].

$$CM^2 = \log[L(O_i|s_i)] - \log\left[\frac{1}{N-1}\sum_{\substack{n=1\\n\neq i}}^{N} exp(\alpha\log[L(O_i|s_n)])\right]^{1/\alpha} \tag{4}$$

where:

$L(O_i\|s_n)$	is the likelihood score of observation O_i given sign s_n
N	is the size of the vocabulary
α	is a constant

The second verification criteria is:

$$V^{(2)} : CM^2 > \beta \tag{5}$$

where β is the threshold.

Next we define CM^3 as the sum of the *forward* and *reverse* pass values of CM^1. Similarly, we define CM^4 as the sum of the *forward* and *reverse* pass values of CM^2.
We have

$$CM^3 = CM^1_f + CM^1_r \tag{6}$$

$$CM^4 = CM^2_f + CM^2_r \tag{7}$$

The third and fourth verification criteria are:

$$V^{(3)} : CM^3 > 0 \tag{8}$$

$$V^{(4)} : CM^4 > 0 \tag{9}$$

Figure 10 outlines the general procedure for applying the verification criteria $V^{(3)}$ and $V^{(4)}$. The signs in the test sentence $\boldsymbol{P}$ are verified if the sum of the confidence measures in forward pass and reverse pass is > 0 and a sign segmentation is selected based on the maximum of the two.

4.2 Experiments

A subset of the Gwinett dataset that contained signing examples from deaf children who were right-hand dominant was selected for the experimentation. It included 420 ASL sentences from five different signers, eliminating variants of signs from left-hand dominant signers and mixed signers from the vocabulary. However, three of the 19 signs in the vocabulary still consistently showed linguistically acceptable variations. To accommodate these variations the vocabulary was increased to 22 signs. The types of features extracted from the input can be found in Section 3.1.

```
$(CM_f, S_f) \leftarrow \{$*forward pass* confidence measures
                    and segmentation for signs in $\boldsymbol{P}\}$
$(CM_r, S_r) \leftarrow \{$*reverse pass* confidence measures
                    and segmentation for signs in $\boldsymbol{P}\}$
**for all** signs (**s**) **do**
  $CM^s \leftarrow CM_f^s + CM_r^s$
  **if** $CM^s > 0$ **then**
    *accept* **s**
    **if** $CM_f^s \geq CM_r^s$ **then**
      $choose\ S_f^s\ as\ the\ segmentation\ for$ **s**
    **else**
      $choose\ S_r^s\ as\ the\ segmentation\ for$ **s**
    **end if**
  **else**
    $reject$ **s**
  **end if**
**end for**
```

Figure 10: Procedure for applying the verification criteria $V^{(3)}$ or $V^{(4)}$ given in equations 8 and 9 respectively, and choosing segmentations for the signs in ASL sentence $\boldsymbol{P}$.

4.2.1 Forward and reverse pass training and testing

We trained left-to-right four-state hidden Markov models (HMMs) with one skip transition for each of the 22 signs. The skip occurs from state one to state three. The Georgia Tech Gesture Toolkit (GT2K) [102] was used for training and testing. In addition to the regular *forward pass* training process we trained sign models by reversing the input features. The N^{th} data frame becomes the 1^{st} frame, the $(N-1)^{th}$ the 2^{nd}, and so on. The hand labeled sign segmentation is then translated to correspond to the reversed features. The HMMs are also changed accordingly, the skip transition now occuring from state two to four. The intuition for using the *reverse pass* is as follows.

Model parameters for both the *forward* and *reverse* pass are estimated iteratively using the Baum-Welch method [75]. Baum-Welch does not guarantee the same results given different orders of input, and it is unlikely for it to generate the same values for reverse versus forward input. The children's sign is highly variable, and sometimes the end of a sign has much less variance than the beginning, or vice versa. Thus, depending on whether the models are trained on the forward or reverse data, alignment of the models' states can vary wildly. Thus, the models can be of better or worse quality depending on the variance observed in the beginning or end of the sign. A similar artifact might be observed during Viterbi decoding. In theory, the forward algorithm should pick an optimal path (assuming that the problem is first-order Markovian, which may not hold here). However, in practice, pruning is needed due to memory and processor limits. If a sign has high variance at the beginning, all paths through the Viterbi lattice could be of approximately the same low probability. Thus, promising paths may be pruned inopportunely. However, with reversed input, the "correct" paths should have significantly higher probability values than the incorrect paths and therefore avoid pruning. Thus, the correct paths may be better preserved in one direction of Viterbi decoding than in the other. We hypothesize that this situation could directly impact the confidence measure for each sign in the *forward* and *reverse* directions. We can take advantage of this fact and combine the measures in a manner already described in Figure 10.

Signer-independent cross-validation was performed by training on data from four students and testing on data from the fifth student. The focus is on sentence-level verification; first, the test sentence $\boldsymbol{P}$ is aligned with the *null hypothesis*; then we check to see if each sign in the sentence has passed the verification criteria outlined in Section 4.1. In the CopyCat game the current graphic (see Figure 2a) directly determines the *null hypothesis* to be used. During data collection the identifier for the game scene was stored along with the sentence sample, which made it possible to obtain the *null hypothesis* for the test sentence during offline testing.

Table 6 lists the the number of training and test sentences used in signer-independent cross-validation for each student. The "x 2" indicates that a second simulated test set of same size, but having incorrect signing, was used to determine the *false alarm* rate. To simulate incorrect signing, sentences from the test student's own signed sentence examples that had one or more signs different from the sentence currently being verified were chosen. The goal was to select as many sentences as possible that were different by only one sign in order to maximize the difficulty of the *false alarm* testing. Table 7 lists two examples of sentences for the sentence depicted in Figure 2a. The first example is straightforward, with ALLIGATOR being replaced by SNAKE; however, in the second example GREEN

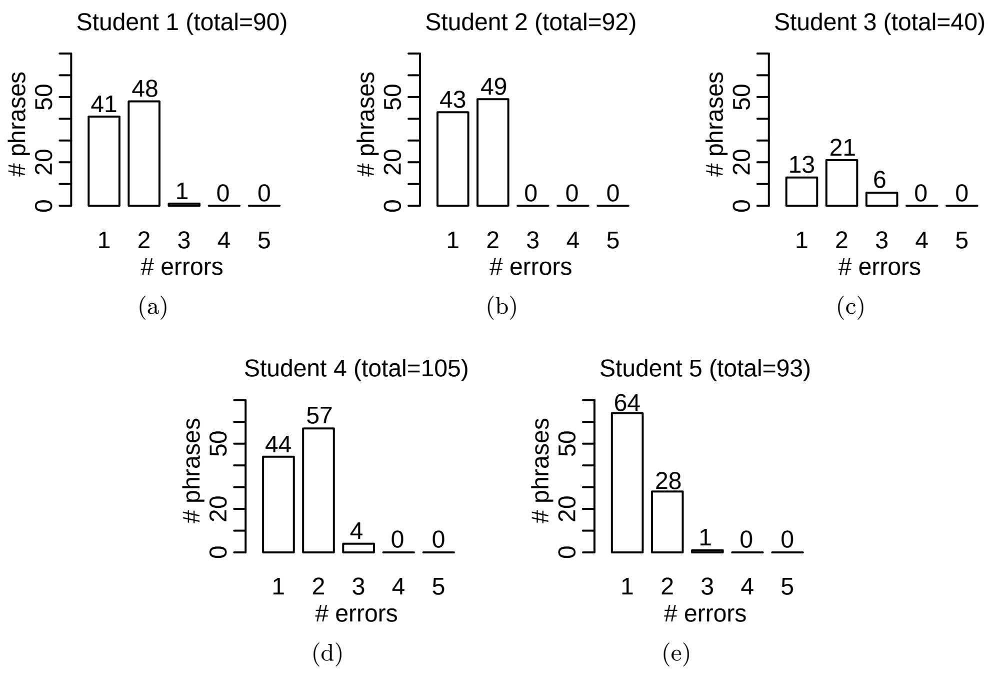


Figure 11: Distribution of simulated sign errors for each student.

Table 6: Training and testing split for signer-independent cross-validation.

Student	#Training	#Testing
1	330	90 x 2
2	328	92 x 2
3	380	40 x 2
4	315	105 x 2
5	327	93 x 2

is not an error since the grammar used by CopyCat allows for an optional first adjective for 4-sign sentences. In Figure 2a GREEN is, in fact, the color of the ALLIGATOR. For the complete distribution of comparison sentences with differences chosen for each student, see Figure 11. With the exception of student five, the majority of sentences had two sign differences.

4.3 Results and Discussion

We obtained signer-independent cross-validation results by applying the four verification criteria outlined in Section 4.1. For CM^1 and CM^3 the value of γ was varied between -5 to

Table 7: Examples of one-sign-error sentences. Errors are shown in bold.

Example 1.	
test sentence	ALLIGATOR BEHIND ORANGE WAGON
error sentence	**SNAKE** BEHIND ORANGE WAGON
Example 2.	
test sentence	ALLIGATOR BEHIND ORANGE WAGON
error sentence	**GREEN** ALLIGATOR BEHIND ORANGE WAGON

Table 8: Head-to-Head wins comparing verification criteria, $V^{(1)}$ vs $V^{(3)}$ and $V^{(2)}$ vs $V^{(4)}$.

Student	$V^{(1)}$ vs $V^{(3)}$		$V^{(2)}$ vs $V^{(4)}$	
	Acc_{max}	Winner	Acc_{max}	Winner
1	76%	$V^{(3)}$	86%	$V^{(4)}$
2	65%	$V^{(1)}$	71%	$V^{(4)}$
3	86%	$V^{(3)}$	88%	$V^{(4)}$
4	82%	tie	94%	$V^{(2)}$
5	69%	$V^{(3)}$	81%	$V^{(2)}$

15. For CM^2 and CM^4 we set the value of α to 0.5 and varied the value of β between -70 to 100. Figure 12 shows the ROC plots of False Alarms vs False Rejects for each student. We see that $V^{(2)}$ and $V^{(4)}$ in general perform better than $V^{(1)}$ and $V^{(3)}$, with curves more closely approaching zero. This result is not surprising given that, according to Neyman-Pearson theory, if the exact densities $L(U|H_0)$ and $L(U|H_1)$ are known then the likelihood ratio test (LRT) is the best available test that gives the lowest false alarm rate for a given false rejection rate. In practice the probability density functions are not easily obtained, but they can be approximated as in the case of Equation 4 [90]. Table 8 lists the winning verification criteria in terms of the maximum achieved accuracy (Acc_{max}) for each student. $V^{(3)}$ scores three wins and ties once with $V^{(1)}$, whereas $V^{(4)}$ and $V^{(2)}$ seem equally matched, taking three and two wins respectively. However, since the number of students is small we will reserve judgement about $V^{(4)}$'s performance until further analysis is conducted.

From this point forward, we will restrict our discussion to comparing $V^{(2)}$ and $V^{(4)}$, the verification criteria that use the LLR as the confidence measure. Figure 13 shows accuracy plots that provide a better perspective of how $V^{(4)}$ performs against $V^{(2)}$. The blue and

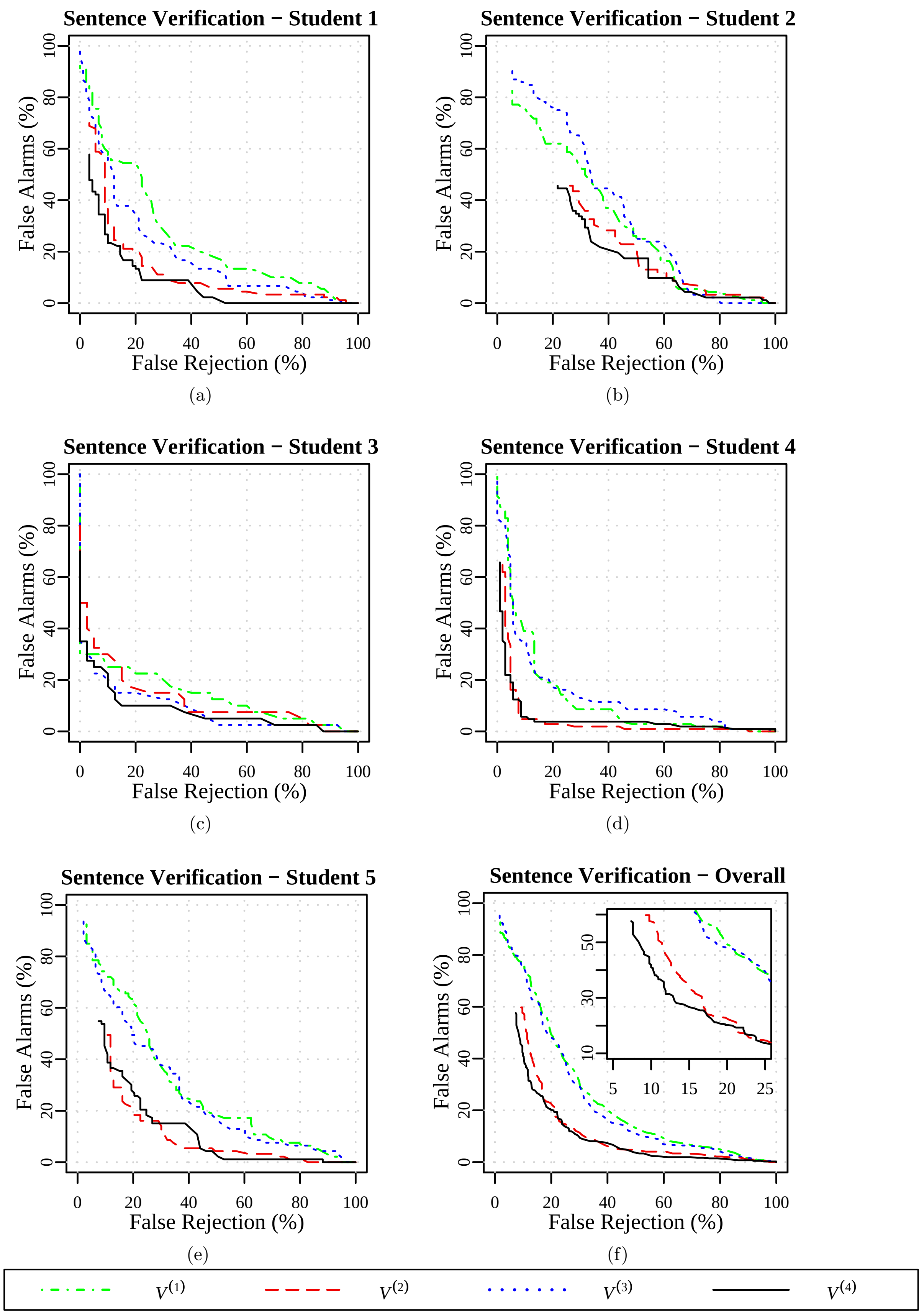


Figure 12: ROC plots for signer-independent cross-validation.

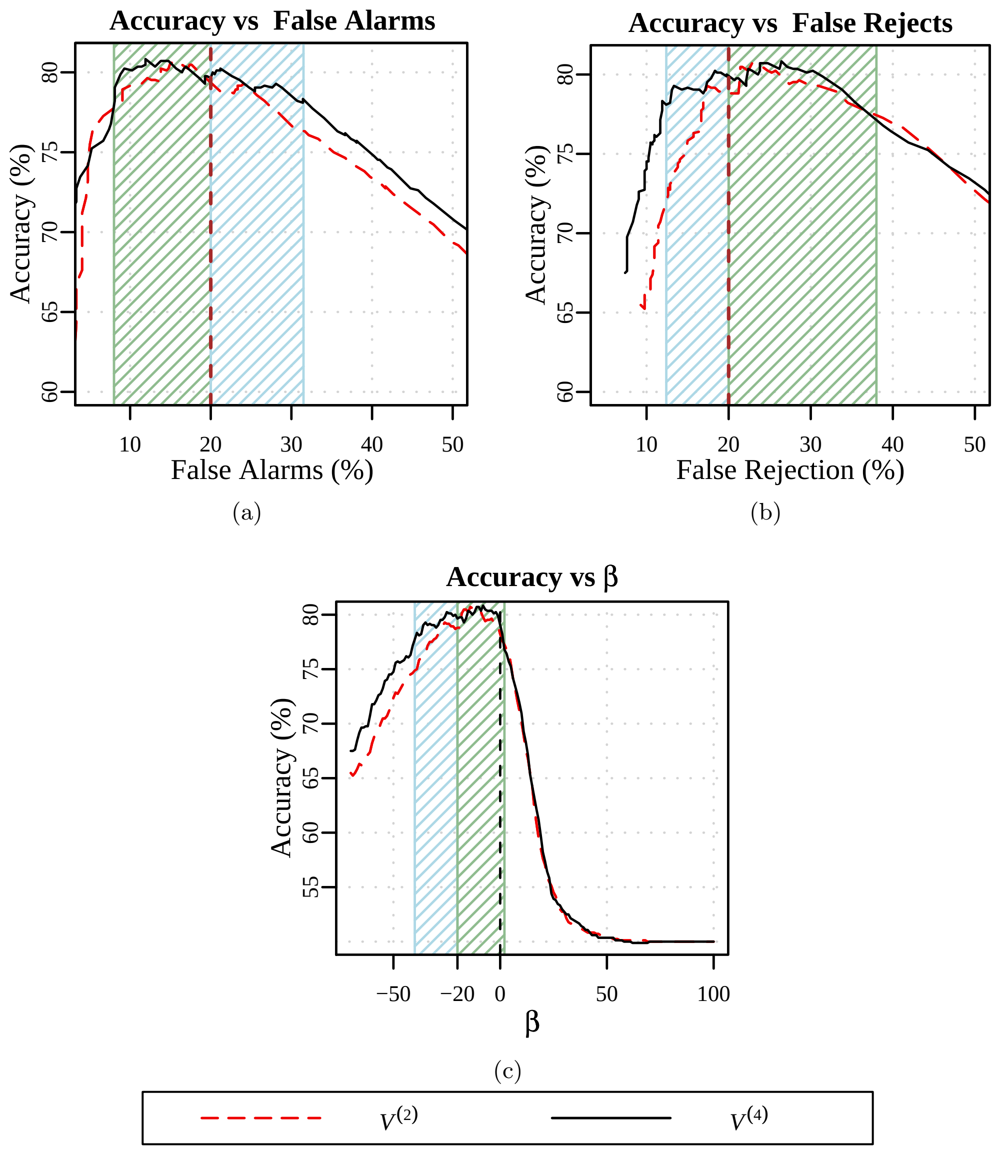


Figure 13: Comparison of verification accuracy for verification criteria $V^{(2)}$ and $V^{(4)}$. (a) Accuracy vs False Alarms (b) Accuracy vs False Rejects (c) Accuracy vs β.

green shaded regions should allow the reader to make better connections between the three plots. Looking at Figure 13b, if the false reject rate is to be maintained below 20%, we can clearly see that there is a significant advantage by choosing $V^{(4)}$ over $V^{(2)}$. However, the corresponding false alarm rates are above 20%. If one were to limit the false alarm rates to below 20% (Figure 13a), the advantage is minor, and at the same time the false reject rates get prohibitively higher. From the perspective of the CopyCat game we would like to keep the false alarm rate as low as possible to avoid any negative impact on the student's language development. Keeping β at -20 as seen from Figure 13c gives an *equal error rate* of 20%, which is a good compromise between false alarms and false rejects. However, with $\beta \geq$ -20 there appears to be no significant advantage in terms of accuracy with respect to $V^{(4)}$ over $V^{(2)}$. Strictly speaking with the likelihood ratio test we should not be considering $\beta < 0$, since in general the likelihood ratio will be ≥ 0 when the observation sequence matches the *null hypothesis* and will be negative otherwise.

We will show with further analysis that indeed there are advantages to using $V^{(4)}$, less in terms of accuracy but more in terms of how well the chosen boundaries match with the hand labeled ground truth boundaries for the signs. Interestingly, on analyzing the *true positive* cases we found that the sentences verified by $V^{(4)}$ and $V^{(2)}$ were not all the same. At $\beta = 0$ Table 9 gives the exact numbers of "All" sentences verified, the "Common" ones, and the sentences "Exclusively" chosen by $V^{(4)}$ and $V^{(2)}$. Overall there are 420 test cases across the five students. $V^{(4)}$ accepts 278 sentences, and $V^{(2)}$ accepts 275 sentences. We see that 259 of these are common to both, 16 are exclusively chosen by $V^{(2)}$, and 19 exclusively by $V^{(4)}$. To measure the distance between the ground truth and the test case boundaries we define a new cost function on the basis of the parameters shown in Figure 14.

$$d = \frac{d_s + d_e}{l_1 + l_2} \tag{10}$$

Remember that according to the decision rule in Figure 10 either the *forward pass* or *reverse pass* boundaries may be chosen depending on which one of the confidence measures is the higher positive value. This rule means that the boundaries for the signs in sentences chosen by $V^{(4)}$ could be significantly different from those of $V^{(2)}$, even for those sentences that are

Table 9: True positive cases for verification criteria $V^{(2)}$ and $V^{(4)}$.

	$V^{(2)}$	$V^{(4)}$
All	$\|P^{V^{(2)}}\| = 275$	$\|P^{V^{(2)}}\| = 278$
Common	$\|P^{V^{(2)}} \cap P^{V^{(4)}}\| = 259$	
Exclusive	$\|P^{V^{(2)}} - P^{V^{(4)}}\| = 16$	$\|P^{V^{(4)}} - P^{V^{(2)}}\| = 19$

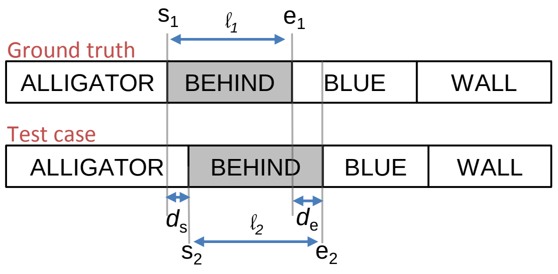


Figure 14: Parameters for computation of boundary distance.

common to both. Figure 15 shows the mean boundary error computed using Equation 10. We can clearly see that in all three cases the sentences chosen by $V^{(4)}$ have sign boundaries that have a better match with the ground truth. Particularly in the "Exclusive" case, the boundary identification performance of $V^{(4)}$ is far superior than that of $V^{(2)}$.

4.4 Summary

In addition to having high sentence verification accuracy the efficacy of the sentence verifier depends on the accuracy of the reported sign boundaries. Having accurate boundary information is crucial from the future perspective of providing feedback to signer about exact location of the errors that occur in signing. In this chapter, we introduced a two-pass technique for ASL sentence verification that utilizes temporally reversed signing input to identify sign boundaries that have a better match to the ground truth as compared to the standard approach. Precise sign segmentation information will enable us to improve the current version of the CoyCat game by incorporating appropriate fine-grained feedback information, which will allow the signer to examine the parts of the ASL sentence that were signed incorrectly in addition to the coarse-grained response indicating whether the entire ASL sentence is signed correctly or not. Additionally, we have shown that if a low

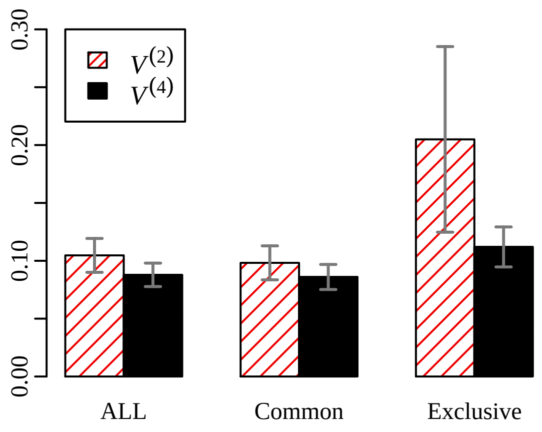


Figure 15: Comparison of mean boundary error, after alignment, between verification criteria $V^{(2)}$ and $V^{(4)}$.

false rejection rate is desired the new method can provide better verification accuracy. In the case of an educational game like CopyCat, this priority is of particular interest. The results from this chapter were published in the proceedings of the 3rd IEEE Workshop on Computer Vision and Pattern Recognition for Human Communicative Behavior Analysis (CVPRHB 2010) [114].

CHAPTER V

ASL VERIFICATION WITH KINECT

The CopyCat Sensor Platform consists of two types of sensors, a camera for eye and hand tracking and a pair of accelerometers that are strapped on the wrists over colored gloves. Although the gloves and accelerometers enhance the performance of the system, there are significant drawbacks to their use. They increase the complexity of the system making it harder for the teachers to set up the system for the children to play the game in the classroom. They require constant maintenance; the batteries need to charged frequently, and the gloves need to be washed regularly. Additionally, the wearable sensors add a possible point of failure.

In this chapter, I present experimental results that show that the Microsoft Kinect sensor can yield performance comparable to the CopyCat Sensor Platform but may require a change in user interaction. The Kinect is desirable in that it provides a more natural interaction and is inexpensive and easy to maintain.

5.1 Data Collection

Data was collected using both the CopyCat Sensor Platform and the CopyCat Kinect Platform in order to compare the performance of the two sensor platforms. The data contains ASL sentences from the expressive signing tasks of the CopyCat game but was collected from adult participants for preliminary testing. The three datasets used for comparison are:

- **CC-Adult** contains data collected from adult signers playing the CopyCat game using colored gloves, wrist-mounted accelerometers and a camcorder.
- **CC-Kinect-seated** contains data that mimics the CopyCat game interactions using the sentence repetition task (discussed in the next section) in a seated position. The seated position matches the configuration used in previous CopyCat studies.

- **CC-Kinect-standing** contains data that mimics the CopyCat game interactions using the sentence repetition task (discussed in the next section) in a standing position. The standing position differs from the original CopyCat task but is more compatible with Kinect skeleton tracking algorithms, which were optimized for standing (and not sitting).

5.1.1 Kinect Data Collection

We recruited seven participants, all of whom were hearing adults with varying levels of ASL expertise, ranging from beginner to expert. All seven signers participated in data collection for the seated data, and three signers participated in data collection for the standing dataset. During each session, signers had 60 minutes to sign as many phrases from a set of 60 sentences as was comfortably possible. One signer was unable to complete a standing session due to an interruption, so that signer had fewer standing examples. A total of error-free 555 phrase samples were collected in the seated pose and 155 error-free phrase samples were collected in the standing pose.

5.1.1.1 Sentence Repetition Task

The sentence repetition task required participants to view videos of ASL sentences from the CopyCat game and sign them back to the camera. A front end application was developed in Java that allowed the signers to view the ASL videos and view real time information from the Kinect camera consisting of the depth map image with skeleton tracking superimposed and the two-dimensional RGB image. The Java interface is shown in Figure 16.

Once the participant viewed a video, he or she would click the “Start Signing” button to begin signing and then the “Stop Signing” button to indicate the termination of the ASL phrase. Since the purpose of this task was to collect data to examine the feasibility of the Kinect camera as a sensor for doing sign language recognition, we encouraged the participants to repeat signing the phrase until they signed it correctly and discarded the phrases with signing errors. Each participant was asked to complete signing all the 60 phrases of the CopyCat game at least once if possible within the 60 minute session. All signers in the seated dataset were able to sign the 60 phrases at least once with some signers

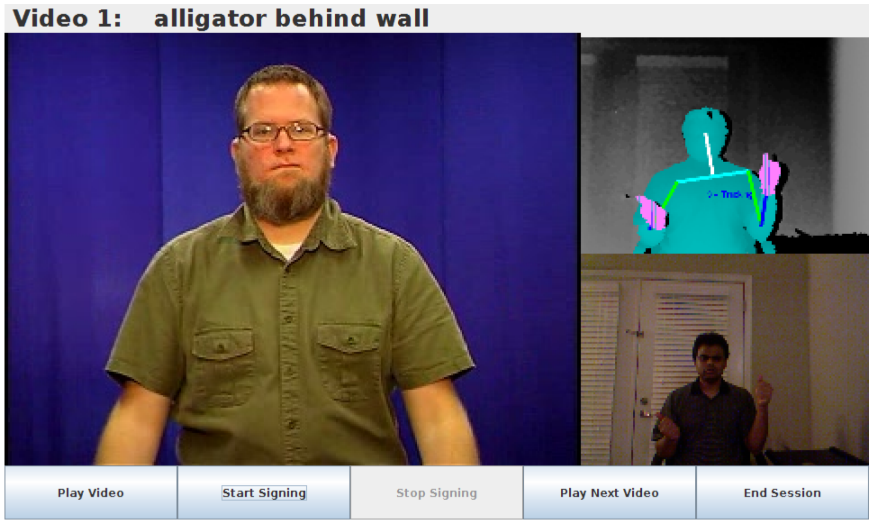


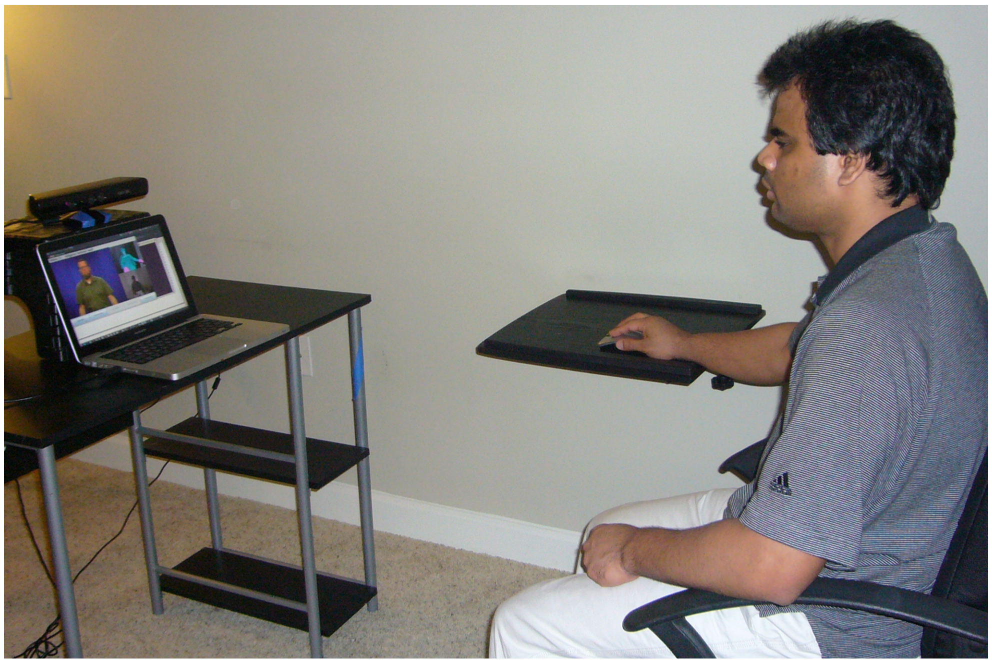
Figure 16: User interface for the sentence repetition task. Video on the left shows a tutorial for the phrase. The right side shows the raw video (bottom) and the video annotated with the skeleton tracking (top). The bottom buttons allow the signer to navigate the system.

managing to sign the first few phrases for a second time. Two signers in the standing dataset completed the entire set of 60 phrases exactly once while one signer did not complete all of them due to an interruption. Figure 17a shows the physical setup where the participant is seated facing the Kinect camera and watching the ASL videos on a laptop computer. The signer controls the interface via a mouse placed on a table next to the chair.

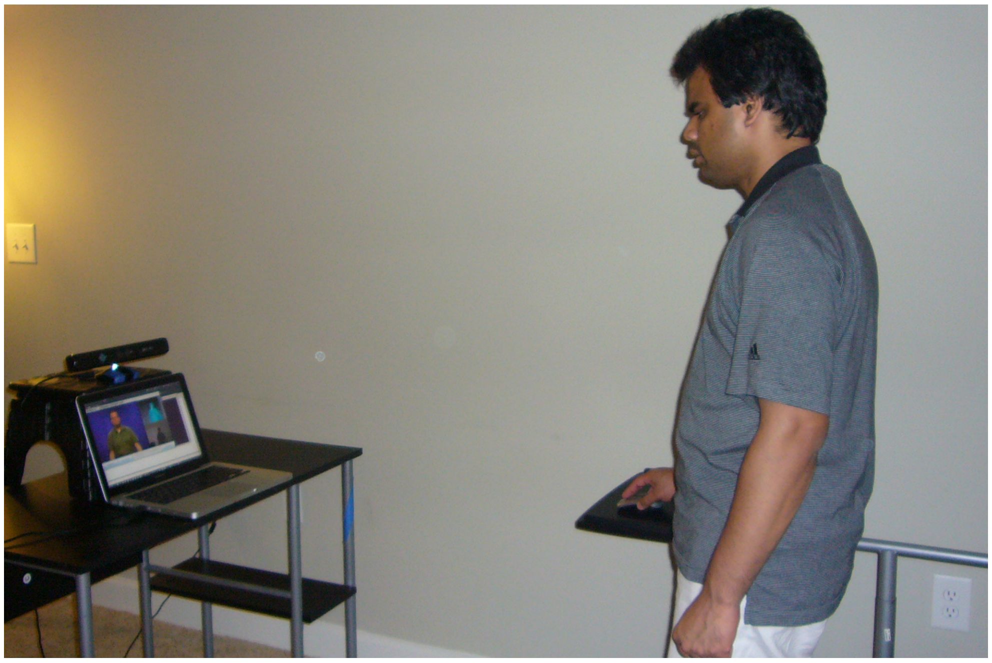
5.1.1.2 Signer Pose: Seated vs. Standing (Kinect data).

The skeleton tracker suffered from performance issues when the signers were seated, losing track of the participant's body pose several times and accompanied by jitter in the limb positions. Since we did not anticipate ahead of time that this problem would be a major issue, statistics on the frame-level accuracy of the tracker were not gathered, especially since obtaining ground truth data by labelling hand locations in each frame is a time-intensive activity. In many cases the participants were allowed to ignore tracking errors and move forward once the phrase was signed correctly. Two signs that caused most of the failures were BED and ALLIGATOR, with BED being the most unfavorable of the two (see Figure 18).

The experiment in the seated position represents the ideal case for a straightforward



(a)



(b)

Figure 17: Participant pose for data collection. (a) Seated (b) Standing.

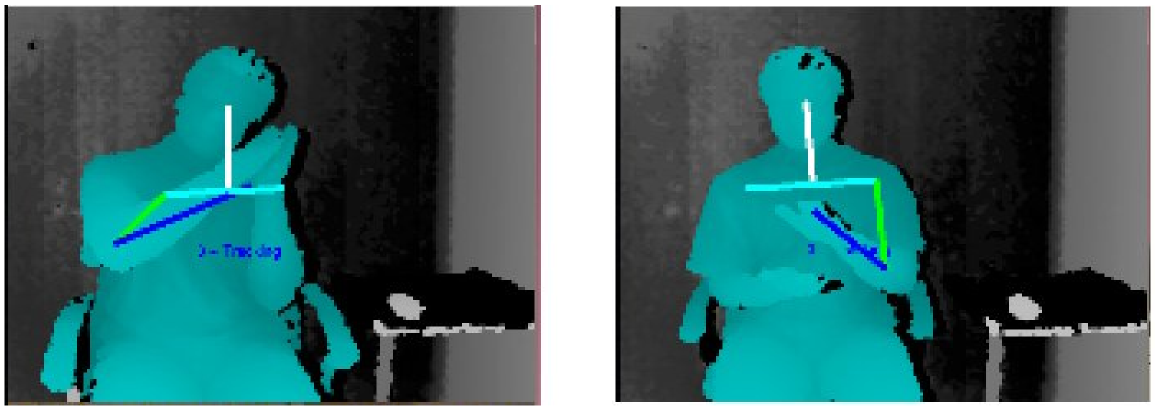


Figure 18: Tracking failure with BED and ALLIGATOR, respectively (Kinect data). Note that the lines should correspond with the signer's arm positions but do not.

comparison since it correctly mimics user interaction used in previous work for the CopyCat project. The high occurrence of tracking failures in the seated condition prompted the collection of a new dataset, for the sentence repetition task, with three signers in the standing position. The standing position setup is shown in Figure 17b. The configuration difference for the standing position is that the camera is angled to capture the body from the knee up, and the mouse is placed on a platform that can be easily reached in the standing position. It was hypothesized that the skeleton tracking library in the OpenNI framework would yield better tracking performance in the standing position.

5.1.1.3 Data Pruning

The standing position suffered significantly less tracking errors compared to the seated position. However, there is no confirmation that all the available skeleton tracking implementations for Kinect depth data suffer from the same problem. The dataset was pruned to remove tracking failures, yielding 146 signed phrases (from 155 possible) in the standing pose and 348 signed phrases in the seated pose (from 555 possible).

5.1.2 CopyCat Adult Data Collection using the CopyCat Sensor Platform

For the CC-Adult dataset we recruited eight participants, all of whom were hearing adults with varying levels of ASL expertise, ranging from beginner to expert. Of the eight participants recruited for this dataset, three also participated in the Kinect data collection.

The dataset contains up to six sessions per participant, each session containing 20 of the possible 60 CopyCat phrases with repetitions occurring between sessions. Participants sat at the CopyCat kiosk (Figure 2c) and played the CopyCat game.

5.1.2.1 Data Pruning

First, we randomly selected two sessions per participant. Each session contains 20 samples, thus giving 320 phrase samples in total. Of these, 43 samples contained errors in data collection and were excluded. In order to choose samples compatible with the sentence repetition task, only those samples that contained correct signing were selected. Five samples were excluded due to incorrect signing, and 26 samples were excluded because they contained variations in the signing such as sign repetition or other disfluencies. One of the signs excluded for language variation also contained machine errors. The resulting dataset contained 247 usable sentence samples.

5.1.3 Data Collection Summary

Table 10 summarizes the number of error-free and variation-free phrase samples collected in each dataset, the number of samples that remain after pruning out data that was unusable due to data collection errors (machine errors) and the percentage of the unusable data. The least amount of data collection errors were observed in the Kinect standing dataset.

Table 10: Error-free samples collected versus sensor failures for all data collected.

Dataset	**#Error-free Samples**	**#Corruption-free Samples**	**%Corrupt Samples**
CC-Kinect-seated	555	348	37.3%
CC-Kinect-standing	155	146	5.8%
CC-Adult	290	247	14.8%

5.2 Experimental Design

We follow a three-step process. The first step is feature extraction whereby salient information that will enable us to perform automatic recognition is obtained. The second step is to train hidden Markov models (HMMs) for each sign in the game vocabulary. In the final

Table 11: Description of body pose features generated using OpenNI framework (Kinect data).

Description	**# Features**
Unit 3D vector from $RS \rightarrow RE$	3
Unit 3D vector from $RE \rightarrow RH$	3
Unit 3D vector from $LS \rightarrow LE$	3
Unit 3D vector from $LE \rightarrow LH$	3
Unit 3D vector from $RH \rightarrow LH$	3
$\angle N - RS - RE$	1
$\angle RS - RE - RH$	1
$\angle N - LS - LE$	1
$\angle LS - LE - LH$	1
dist $(RH \rightarrow LH)$	1
Total	20

step, the generalization performance of theses models is measured by testing on independent data.

5.2.1 Features

The feature extraction process for the CopyCat Kinect Platform differs significantly from that of the earlier CopyCat Sensor Platform. Refer to section 3.1 for details on feature extraction with the CopyCat Sensor Platform. Feature extraction with the CopyCat Kinect Platform is described below.

5.2.1.1 Body pose features

To extract body pose features, the skeletonization capability provided by the the OpenNI framework was used. OpenNI stands for "Open Natural Interaction" and provides a set of APIs that allows developers to write applications that utilize natural interaction. The OpenNI framework interfaces with NITE middleware provided by PrimeSense, the company that developed the Kinect hardware [73].

The skeleton tracker can be configured to return only the joints in the upper body. The shoulder, elbow and hand positions for the right and the left side of the body are obtained. The body pose feature is built using the joint angles of the shoulders and the elbows, the unit vectors of the elbows with respect to the shoulder, the hands with respect to the elbows

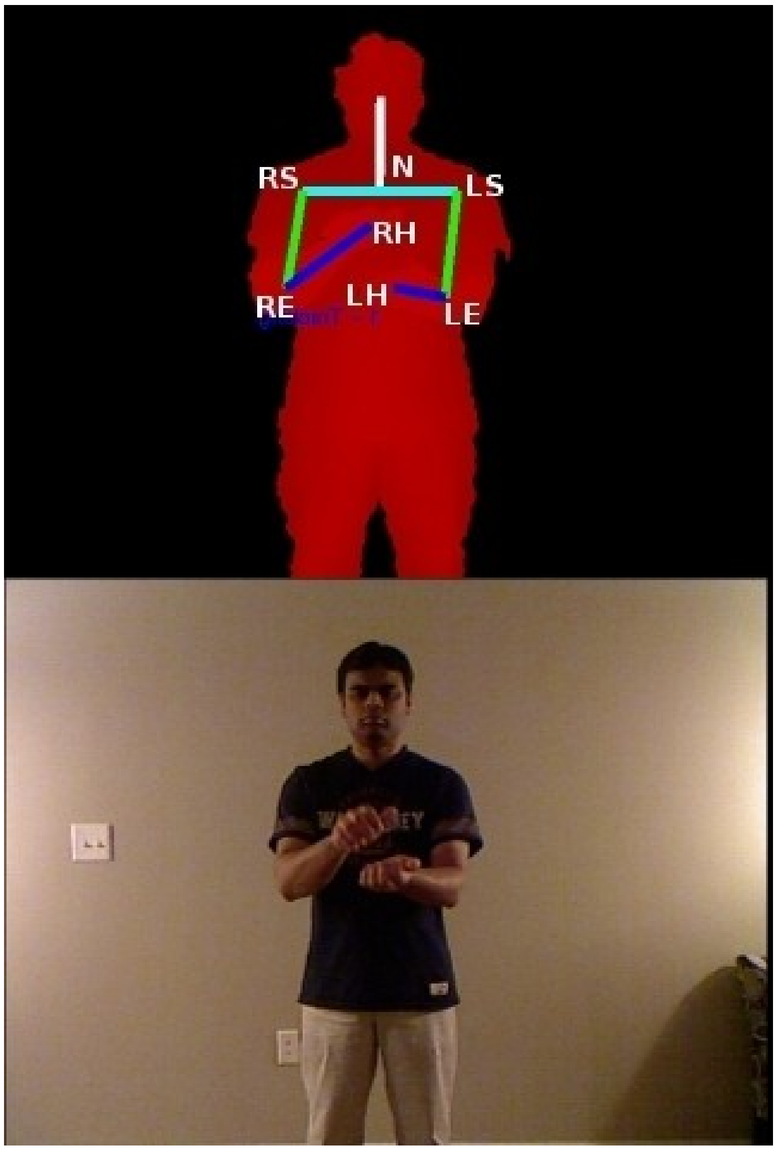


Figure 19: Visualization of upper body skeletal joints generated using OpenNI framework (Kinect data).

and finally the left hand with respect to the right hand. See Figure 19 and Table 11 for details of the body pose feature. The body pose feature has 20 dimensions.

5.2.1.2 Hand shape features

Three-dimensional points in the neighborhood of each of the hand positions (RH and LH in Figure 19) returned by the skeleton tracker are collected. The points corresponding to each hand are clustered separately to obtain a mixture of six Gaussians (MoG). For simplicity we use diagonal covariance matrices while learning the mixture using expectation maximization (EM), resulting in a combined feature of 72 dimensions. The parameters of the Gaussians serve as the initial features. We then perform principal component analysis (PCA) on the entire dataset of MoG hand features to reduce the dimensionality of the hand shape feature from 72 to 20.

5.2.2 Training and Testing

Once the features are collected, a left-to-right four-state hidden Markov model with one skip transition is trained for each of the 19 signs in the vocabulary shown in Table 2. Training and testing was done using the Georgia Tech Gesture Toolkit GT^2K [102]. To evaluate the performance of the system, two types of tasks, *recognition* and *verification* are performed. The difference between recognition and verification is explained in the introduction of Chapter 3.

For both recognition and verification the generalization performance of trained models is assessed using signer-independent cross validation. However, the procedure is different in each case. For recognition, the approach is straightforward. We set aside one participant's data for testing and train on data from the remaining participants. In verification, since a likelihood threshold is applied, we first need to determine the appropriate threshold to use. The procedure to learn a threshold for each sign is outlined in Figure 20. It is a two-level signer-independent procedure whereby the inner signer-independent step is used to determine the log-likelihood threshold to be used for each sign, and the thresholds thus computed are then used in the outer signer-independent step. The results from each signer-independent run, the outer runs in the case of verification, are combined to obtain the

```
for all participants (p) do
  $V_p \leftarrow$ set aside participant $\mathbf{p}'s$ data for validation
  for all remaining participants (r) do
    1. $V_r \leftarrow$ set aside participant $\mathbf{r}'s$ data for validation
    2. Train on remaining $N-2$ participants and test
       on $V_r$
    3. For each instance of a sign in $V_r$ note the log
       likelihood value obtained via Viterbi alignment
  end for
  For each sign calculate the mean($\mu$) and standard
  deviation($\sigma$) of the log − likelihood values collected
  Set a log − likelihood threshold for each sign
  ($\mu - \kappa * \sigma$, where $\kappa > 0$ is a parameter to scale
  the standard deviation)
  Train on remaining $N-1$ participants and test on $V_p$
  using the computed thresholds
end for
```

Figure 20: Procedure for computing log-likelihood thresholds to perform verification.

overall recognition and verification accuracy.

5.3 Results

We evaluate performance based on both sentence recognition and sentence verification accuracies.

5.3.1 Baseline results from experiments on the Gwinnett dataset

Table 12: Experiments using Gaussian mixture models for the Gwinnett dataset show a maximum sentence recognition accuracy of 67.0% and a maximum sentence verification rate of 82.0% [114].

	Verification (%)			**Recognition (%)**		
	TP	**FP**	**Acc.**	**TP**	**FP**	**Acc.**
M=1	86.0	38.0	79.8	54.0	3.0	64.9
M=2	88.6	37.2	**82.0**	56.8	3.3	**67.0**

A comparison of sentence recognition and verification rates for the Gwinnett dataset resulted in a maximum sentence recognition accuracy of 67.0% and a maximum sentence verification accuracy of 82.0% (see Table 12) [114]. Performance was measured using cross validation by setting aside 90% of the data for training and using the remaining 10% for testing. For more details refer to section 3.2.2.

5.3.2 Kinect Data

Table 13 and Table 14 show the recognition and verification results of leave-one-out cross validation in the seated and standing poses respectively. For the seated dataset the threshold selection for verification was done using the procedure shown in Figure 20.

Table 13: Sentence recognition and verification results for the pruned CC-Kinect-seated dataset (CopyCat Kinect Platform).

	Recognition		**Verification**	
Subj	**Word Acc**	**SENT Acc**	**Word Acc**	**SENT Acc**
1	51.79	26.67	89.29	60.00
2	77.19	43.9	96.56	87.80
3	73.1	35.00	95.17	82.50
4	69.11	25.00	92.68	71.88
5	75.53	32.2	96.38	85.59
6	80	43.48	92.78	76.09
7	70	40	96.00	86.67
Overall	74.48	36.2	95.16	82.18

Table 14: Sentence recognition and verification results for the pruned CC-Kinect-standing dataset (CopyCat Kinect Platform).

	Recognition		**Verification**	
Subj	**Word Acc**	**SENT Acc**	**Word Acc**	**SENT Acc**
1	65.04	26.79	87.61	58.93
2	75	38.33	98.33	93.33
3	85.71	50	99.25	96.67
Overall	73.62	36.3	94.49	80.82

The value of κ, described in the procedure in Figure 20, was varied from 1 to 10. At $\kappa = 6$ the increase in verification accuracy begins to flatten out. The results for $\kappa = 6$ are shown in Table 13. See the discussion section for the average verifcation accuracies for all values of κ. Since there were only three participants in the standing dataset, the threshold selection procedure could not be applied. Instead a common threshold for all signs was chosen, which was obtained by averaging the threshold values from the seated dataset, with $\kappa = 6$, across all signs (s) and all the participants (p) (see Equation 11).

$$\frac{1}{|p| * |s|} \sum_p \sum_s \mu - \kappa * \sigma \tag{11}$$

Based on the overall recognition and verification accuracies, there appears to be no significant difference in performance between seated and standing positions, which shows that the recognition/verification framework is robust to the skeleton tracking jitter that occurs in the seated position. When tracking errors are ignored the seated pose does slightly better in terms of verification acheiving 82.18% accuracy compared to 80.82% in the case of the standing pose.

5.3.3 CopyCat Adult Data

Table 15 shows the recognition and verification results of signer-independent cross-validation on the CC-Adult dataset. The threshold selection to perform verification was done using the procedure outlined in Figure 20, which is the same approach used for the CC-Kinect-seated dataset. The value of κ was varied from 1 to 10 but only results for $\kappa = 6$, the point at which the verification accuracy begins to flatten out, are shown in Table 15.

Table 15: Sentence recognition and verification results for the pruned CC-Adult dataset (CopyCat Sensor Platform).

	Recognition		**Verification**	
Subj	**Word Acc**	**SENT Acc**	**Word Acc**	**SENT Acc**
1	80.43	42.42	99.34	97.22
2	63.64	21.05	92.05	63.16
3	79.01	43.24	97.10	87.88
4	91.23	61.54	94.44	78.33
5	86.18	52.78	91.30	70.00
6	93.21	72.97	100.00	100.00
7	80.43	40.00	95.80	84.62
8	84.03	53.85	100.00	100.00
Overall	85.42	51.01	96.86	87.85

5.3.4 Discussion

Table 16 provides a comparison between the three datasets based on the following factors: Recognition/Verification Accuracy, Robustness, Aesthetics and Comfort, and User Interaction.

Table 16: Comparison between three configurations of the CopyCat game apparatus based on ratings for Recognition/Verification Accuracy, Robustness, Aesthetics and Comfort, and User Interaction (a rating of X means further investigation required).

	CC-Adult	**CC-Kinect-seated**	**CC-Kinect-standing**
Recognition / Verification Accuracy	good	fair	fair
Robustness	fair	X	good
Aesthetics & Comfort	fair	X	X
User Interaction	good	X	X

5.3.4.1 Recognition/Verification accuracy

The experiments show that the existing CopyCat Sensor Platform performs significantly better than the CopyCat Kinect Platform in terms of recognition rates and also yields higher verification rates. Table 17 gives a summary of the overall recognition and verification rates for the three datasets. The baseline results on the Gwinnett dataset are also included for reference. However, for these experiments the datasets were pruned to include only those samples with no data corruption. The numbers are likely to change, once erroneous data samples are included.

5.3.4.2 Robustness

The CopyCat Sensor Platform involves more than one sensor resulting in more failure points. Since conventional computer vision algorithms are used for eye tracking and hand tracking, environmental factors such as lighting conditions can have a negative impact on the robustness of tracking. Although the accelerometers are sampled at twice the video frame rate, wireless transmission delays and buffer overflow problems in the accelerometer hardware

Table 17: Summary of sentence recognition and verification results on the CopyCat Kinect Platform datasets and the CopyCat Sensor Platform datasets.

Dataset	**Recognition**		**Verification**	
	Word Acc	**SENT Acc**	**Word Acc**	**SENT Acc**
CC-Kinect-Seated	74.48	36.2	95.16	82.18
CC-Kinect-Standing	73.62	36.3	94.49	80.82
CC-Adult	85.42	51.01	96.86	87.85
Gwinnett	-	67.0	-	82.0

can cause loss of accelerometer data. When there is significant loss of accelerometer data, the inability to match a video frame with an accelerometer frame results in a corrupt data frame.

The Kinect estimates depth by projecting an infrared light pattern, making it almost immune to environmental factors like lighting conditions, resulting in a very reliable depth map output. However, the skeleton tracking facility provided by the OpenNI framework is robust only in the standing pose and fails more than one third of the time when the signer is seated (see Table 10). Once we factor in the tracking errors, considering them as recognition and verification failures, the standing pose emerges as the more favorable of the two configurations. A detailed comparison between the CopyCat Sensor Platform (CC-Adult dataset) and the CopyCat Kinect Platform (CC-Kinect-seated dataset) is shown in Figure 21. This graph highlights a significant drop in the overall verification accuracy for the CC-Kinect-seated dataset due to the large amount of corrupt data.

Further investigation is clearly warranted regarding the standing vs seated problem, and other frameworks for skeleton tracking with the Kinect need to be thoroughly explored in order to determine if they may be able to overcome the tracking problem in the seated position.

Table 18 lists the updated recognition and verification rates once we factor in the errors that occur in data collection for all three datasets. The CopyCat Sensor Platform still performs better at recognition, but given that in the CC-Kinect-standing dataset, fewer

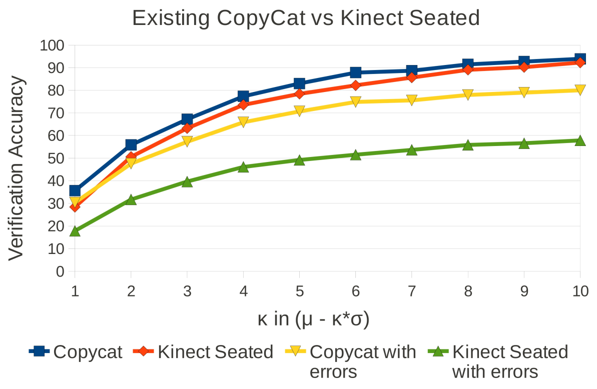


Figure 21: Comparison of verification accuracy between CopyCat Sensor Platform (CC-Adult dataset) and the CopyCat Kinect Platform (CC-Kinect-seated dataset), with and without data collection errors, for different values of the threshold.

data collection errors were observed, the effective verification accuracy for the CC-Kinect-standing dataset is better than the CC-Adult dataset.

Table 18: Comparison of sentence recognition and verification results on the unpruned CC-Kinect-seated, CC-Kinect-standing and CC-Adult datasets.

	Recognition		**Verification**	
	Word Acc	**SENT Acc**	**Word Acc**	**SENT Acc**
Kinect Seated	47.74	22.7	58.86	51.5
Kinect Standing	70.45	34.19	88.02	76.12
CopyCat Adult	73.72	43.44	83.59	74.82

5.3.4.3 Aesthetics & Comfort

The Kinect is a clean and neat, out-of-the-box solution for obtaining depth information. With minimal hardware, it provides a clutter free environment for the CopyCat game. The CopyCat Sensor Platform requires a kiosk-like setup. Additionally, accelerometer batteries may need to be regularly recharged, and circumstances may arise when controlled lighting is required. A researcher has to be present to run the CopyCat system during deployment, whereas the Kinect provides a plug-n-play interface that even individuals without a technical background will be able to operate with ease.

One of the major drawbacks of the CopyCat Sensor Platform is that it requires the signer to wear gloves with accelerometers strapped to the wrist to enable the capture of gestural information. Although previous studies have shown that children did not have problems wearing the gloves, the comfort factor with Kinect and the facility to interact naturally without aids may have a greater appeal to the children for whom the game is designed.

5.3.4.4 User Interaction

The existing CopyCat setup, where the signer is seated in a chair and interacts with the game using a mouse placed at a convenient location next to the chair (see Figure 2c), has been deployed several times in schools with deaf children. This interaction model was designed iteratively over several usability studies conducted in partnership with local deaf schools with the involvement of deaf children and their teachers [49]. One of the concerns when deciding upon the seated pose for the child was fatigue that may occur if the children have to stand continuously for the 30-45 minutes required to finish one session of the CopyCat game. Currently we have yet to ascertain the effect on user experience that the standing position may have, and to what extent the fatigue factor might come into play. In the least we can say that in the standing position the children will have more freedom of movement and as fatigue sets in, the resulting restlessness will likely cause inconsistencies in the data collected.

One critical aspect of the user interaction involves the teacher in the classroom who will assemble the CopyCat game for the children to play. As compared to the CopyCat Sensor Platform wherein the teacher has to extensively manage the hardware (e.g., charge the batteries regularly and maintain the gloves in good, usable condition), the Kinect provides a convenient out-of-the-box solution that frees the teachers from the hassles of the CopyCat Sensor Platform. Moreover, the Kinect is easy and inexpensive to replace.

5.4 Summary

In this chapter, I presented the CopyCat Kinect Platform as a viable alternative to the CopyCat Sensor Platform for the task of automatic ASL sentence verification in the CopyCat game. This approach mitigates the need for signers to wear gloves and strap sensors on their wrists, which is the most significant drawback of the current CopyCat system. The experimental results show that verification results with the Kinect are comparable to the existing system. However, it was discovered that when the signers are seated, the skeleton tracking system tends to make a significant amount of tracking errors, which could have a negative impact on the overall user experience while playing the CopyCat game. One alternative is to change the way signers interact with system, having them stand rather than being seated. Experiments show that this results in fewer tracking errors and slightly better verification accuracy if we compare results when the datasets were not pruned to remove data collection errors. Changing the signer's pose from a seated to standing position involves a shift in the interaction model for the game. The effects of this change on user experience have not yet been explored. The results on verification performance with the Kinect were published in the proceeding of the 13th International Conference on MultiModal Interaction (ICMI 2011) [115].

CHAPTER VI

ASL CLASSIFIER RECOGNITION

Classifiers are an important part of learning American Sign Language (ASL). They are powerful tools that give enormous expressive power to signers allowing them to represent complex concepts with much brevity. Classifiers in ASL utilize surrogate hand shapes for people or "classes" of objects and provide information about their location, movement and appearance [94]. The spatial constructions of ASL classifiers take advantage of the fact that visual spatial information is one of the most reliable sensing modality in humans [38]. Studies have shown that deaf signers have enhanced visual spatial processing capabilities in tasks such as mental rotation, visual attention and face recognition [9, 25, 70, 92]. Deafness *per se* is not the cause for this enhanced capability, exposure to a sign language is [71]. Studies have also shown that deaf children who have not been exposed to ASL classifiers at an early age have lower comprehension and diminished use of ASL classifiers at a later age [54]. Continuing with our earlier motivation to provide ASL learning tools that enable ASL exposure to deaf children of hearing parents at an early age, it becomes imperative that we expand our research in ASL recogition to support the recognition of ASL classifiers.

In the field of ASL recognition, much attention has been given to recognition of linguistic structures such as finger spellings, basic signs, facial expressions and interrogative words like WH-questions (e.g. who, what, where, and when), while recognition of ASL sentences and classifier-based grammatical structures is still comparatively underexplored. The complex spatial constructions in ASL classifiers make recognition challenging with input from a limited source of information such as a regular RGB camera. However, the addition of a depth camera can potentially provide ample information that would be sufficient to successfully recognize classifier constructions in ASL sentences. *I hypothesize that we can extend our ability to recognize sign language by leveraging depth maps to develop a method using improved hand detection and handshape classification to recognize selected classifier-based* *grammatical structures of American Sign Language.*

To remind readers of the concept of ASL classifiers I will revisit an example from Chapter 1. This example demonstrates the use of the TREE classifier and the VEHICLE classifier. It shows three situations, the first being CAR DRIVES BEHIND TREE (Figure 1a), the second being CAR CRASHES INTO TREE (Figure 1b) and the last one shows the CAR DRIVES IN FRONT OF TREE (Figure 1c). In the above example the location of the CAR changes in the physical construct to convey three different meanings. The accompanying depth map provides us with enough information and makes it easier to discern the semantic difference between the three situations. The same task becomes extremely challenging if only the RGB image is taken into consideration.

6.1 Classifier-Kinect dataset

Before undertaking the ASL classifier recognition task, a team of two ASL linguists, five graduate students from the field of ASL linguistics who were also fluent signers and three researchers in the field of ASL recognition met to design and collect a dataset containing representative examples of ASL sentences that demonstrated the use of ASL classifiers.

6.1.1 Design

Discussions during the design phase took into consideration the following criteria:

1. *The classifier sentences should be short in length yet demonstrate the expressive power of ASL classifiers.*

   The team agreed upon a sentence structure that involved a mix of basic signs and ASL classifiers including two predicates, a *verb of location* and a *verb of movement*. The *English* and the *ASL gloss* sentence structure are given in Figure 22. "Glossing" is a means by which we can transcribe ASL into English. The shared premise for all sentences is about the depiction of interaction between two nouns, a direct-object noun and a subject noun. Take for example the sentence CAR BUMPS-INTO BOY, where CAR and BOY are the subject and the direct-object nouns respectively, and BUMPS-INTO is the verb-of-movement predicate. The ASL gloss reveals that the

English: <subject> <verb> <direct-object>

ASL Gloss: <direct-object$_{LH}$> <classifier$_{doLH}$><verb of location> <subject$_{RH}$> <classifier$_{subjRH}$>-<verb of movement>

Figure 22: The typical sentence structure for the Classifier-Kinect dataset. (LH - left hand, RH - right hand, do - direct object, subj - subject)

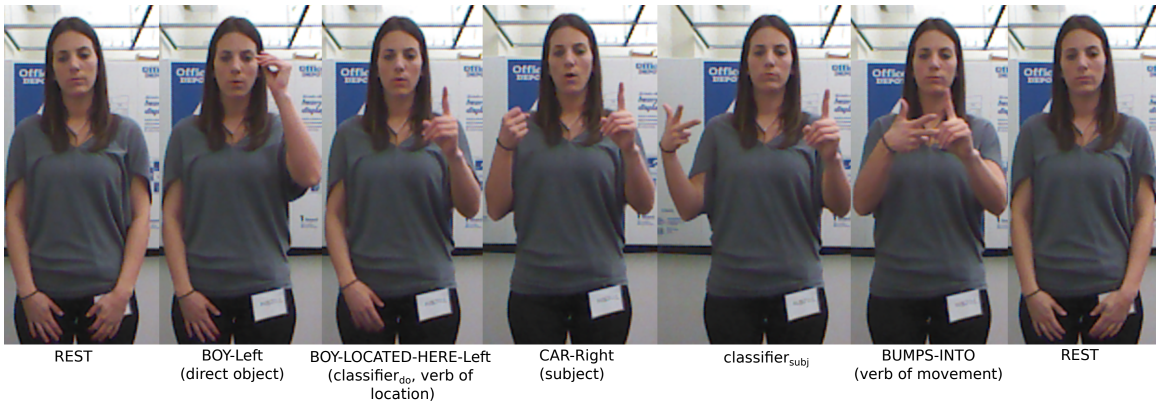

REST
BOY-Left (direct object)
BOY-LOCATED-HERE-Left (classifier$_{do}$, verb of location)
CAR-Right (subject)
classifier$_{subj}$
BUMPS-INTO (verb of movement)
REST

Figure 23: Gloss-based visual breakdown of the ASL classifier sentence CAR BUMPS-INTO BOY. (do - direct object, subj - subject).

direct-object noun is signed first followed by the classifier for the direct-object and verb of location to indicate the position of the object in space; then, the subject noun is signed followed by the classifier for the subject and the verb of movement. Figure 23 shows the visual breakdown, matching the ASL gloss against frames from the video of the ASL classifier sentence CAR BUMPS-INTO BOY. Besides being good candidates for demonstration of the brevity of ASL classifiers, two-noun interactions not only present a signing challenge to a novice ASL student having to switch from one hand to another but also provide a challenging case for automatic recognition.

2. *The classifier sentences should closely mimic real world use of ASL classifiers.*

Although the team members with experience in ASL linguistics argued that the sentence structure shown above doesn't frequently occur as is in ASL dialogue, it was deemed appropriate as a means to concisely incorporate context establishment with the use of ASL classifiers. It was agreed upon that this sentence structure adds pedagogical value and allows students to practice the use of ASL classifiers. From the standpoint of designing a future CopyCat-like ASL classifier game the team agreed that this sentence structure was useful.

3. *The dataset sentences will demonstrate the use of the most common ASL classifiers.*

   Through physical demonstration the linguists and the students iterated over several classifier examples in ASL, and chose five common classifiers that fit well into the chosen sentence structure. Then, one or more nouns of the same class were chosen and paired with each classifier. Table 19 lists the chosen classifiers, the respective handshapes and the noun signs that belong in each class. Furthermore, five verb-of-movement classifier predicates, listed in Table 20, were chosen to represent interactions between the nouns. The nouns were permuted with each other, independent of class, and then further permuted with classifier predicates to produce unique sentences to add to the dataset. Although a total of 500 permutations are possible, we chose 92 of those that the team thought had real world significance.

4. *Nuances that produce subtle changes in meaning of the classifier predicate will be omitted.*

   Lets consider the classifier predicate BUMPS-INTO from Table 20 with the VEHICLE (CL-3) classifier from Table 19. The termination state of this predicate as intended for the Classifier-Kinect dataset is shown in Figure 24a. A morphological change at the termination state as shown in Figure 24b would add additional meaning to the predicate, indicating that the vehicle was crushed after impact. Keeping in mind the current limitations of the hardware, specifically the low resolution of the depth camera, we thought it best to omit such subtle representation that would make recognition even more challenging.

6.1.2 Data collection

The Classifier-Kinect dataset was designed and collected in spring of 2012. The five fluent signing students of ASL linguistics participated in the data collection process by signing each of the 92 sentences once. Figure 25 shows the data capture configuration. A Microsoft

Table 19: Classifiers, classifier handshapes and nouns in the Classifier-Kinect dataset. (images © 2004, www.Lifeprint.com. Used by permission.)

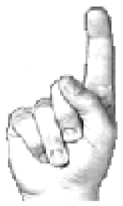
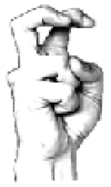
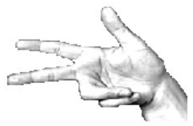
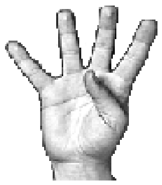
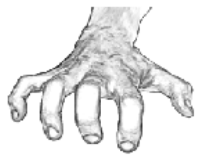
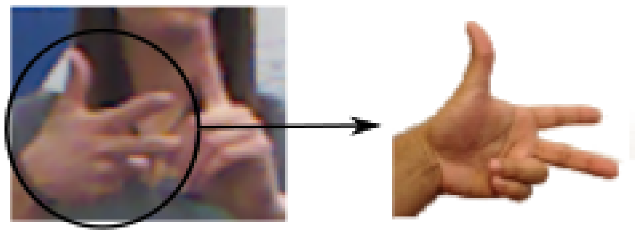
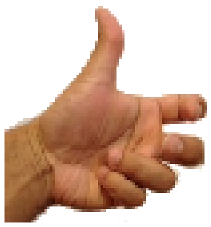
Classifiers	Handshape	Nouns
Person (CL-1)		MAN WOMAN BOY GIRL
Animal (CL-V-BENT)		LION CAT WOLF
Vehicle (CL-3)		CAR
Extended Object (CL-4)		FENCE
Firm or stable object (CL-5-CLAW)		HOUSE

Table 20: Verb-of-movement classifier predicates in the Classifier-Kinect dataset.

Classifier Predicates
MEETS BUMPS-INTO WEAVES-AND-BUMPS-INTO BUMP-INTO-EACH-OTHER JUMPS



(a) CL-3



(b) CL-3-bent

Figure 24: CL-3 and modified CL-3 convey different meanings. (a) BUMPS-INTO (b) BUMPS-INTO-CRUSHED.

Kinect camera, connected via USB to a laptop computer, is aimed at the signer's upper body while the researcher controls the data capture software and guides the signer to sign each of the classifier sentences in the dataset. To speed the process of data collection we used four identical hardware setups.

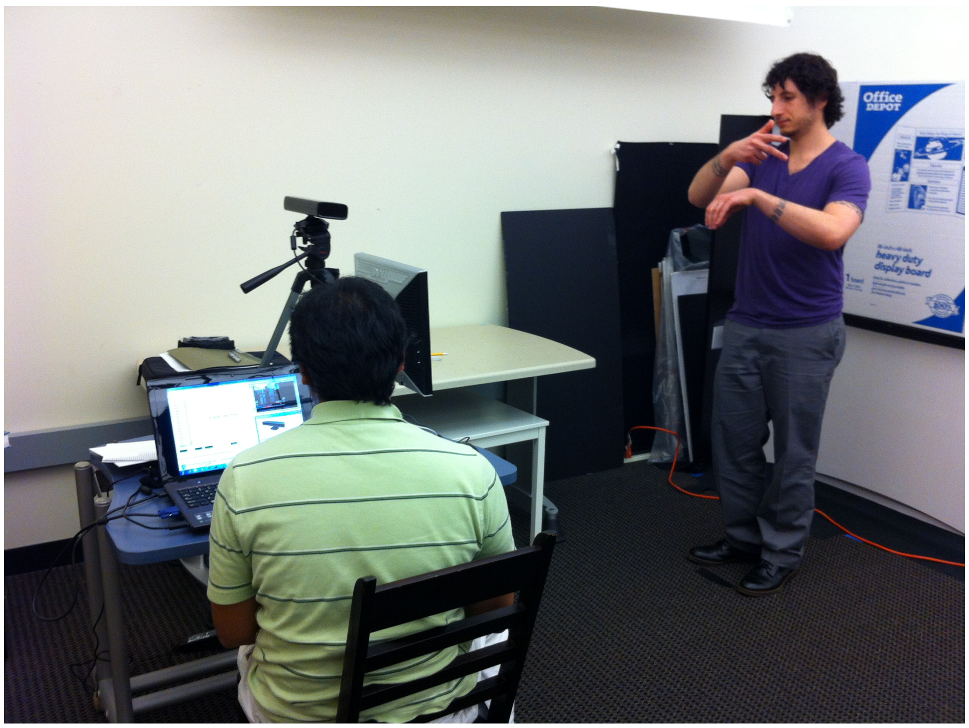
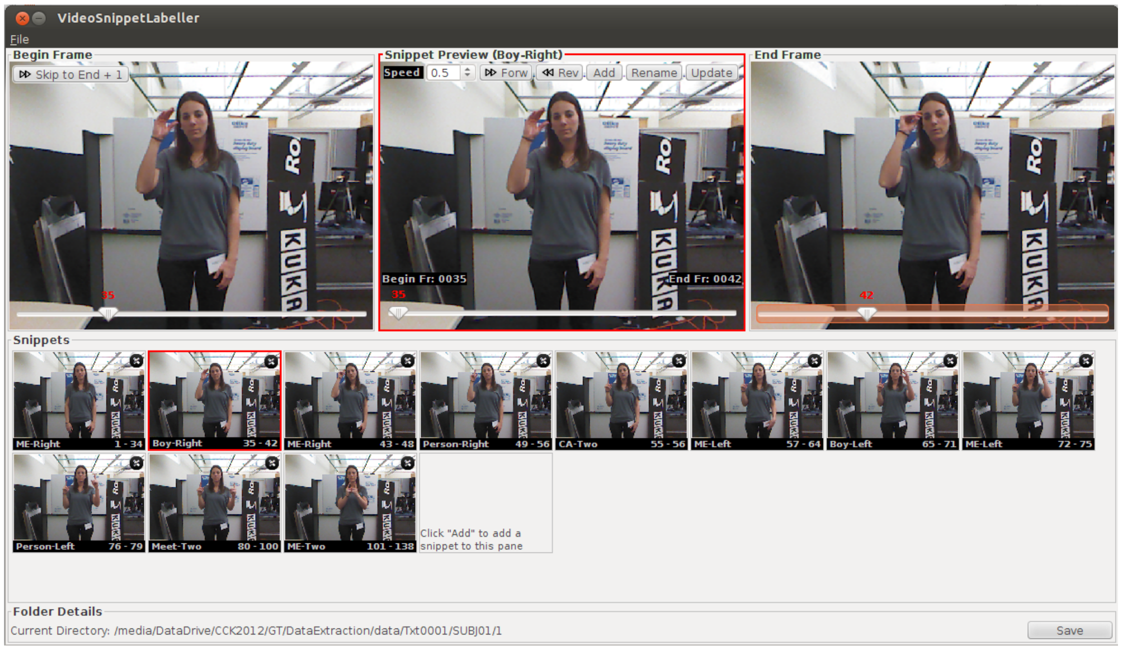


Figure 25: Data capture configuration to collect signing examples of ASL classifier sentences.

A total of 488 classifier sentences were collected with 92 unique sentences signed once by each one of the five participants. Twenty-eight of those sentences were repetitions to correct signing errors that were spotted during data collection. Seventeen other sentences were pruned out of the dataset after the data collection. These either contained signing errors overlooked at the time of data collection or other data collection errors. The final dataset contains 443 ASL classifier sentences. Using the Microsoft Kinect SDK 1.0, the data capture software was programmed to store 640x480 color images, 320x240 depth images captured by the depth camera, and 320x240 depth images that had been transformed to directly map over the color images, all at a frame rate of 30 fps. Additionally, the Microsoft Kinect Skeleton Tracker (MKST) was activated, and all joint information, including the hand locations was stored for each frame in 3D camera coordinates, 2D color image coordinates and 2D depth image coordinates. From this point forward we will abbreviate Microsoft Kinect Skeleton Tracker with the acronym MKST.

6.1.3 Data Labelling

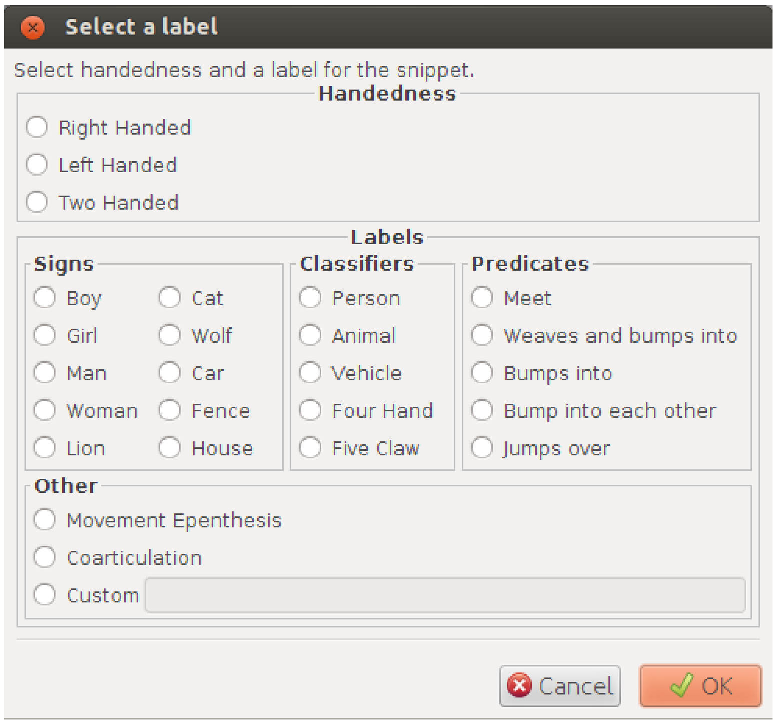


Current Directory: /media/DataDrive/CCK2012/GT/DataExtraction/data/Txt0001/SUBJ01/1

Figure 26: Screenshot of the segment labelling tool for the Classifier-Kinect dataset. The top-left and top-right panels allow for the selection of the start and the end frame of the segment. The panel in the middle is a custom video player that allows for the segment to be viewed from start frame to end frame at different speeds. The Snippets panel shows the currently labelled segments. Shown above: selection of snippet BOY-RIGHT.

A data labelling tool was developed in Java in order to label the frame boundaries of the various sentence segments for each sentence in the Classifier-Kinect dataset. Reliable label data is important for the training of hidden Markov models of noun signs and verb-of-movement classifier predicates. The segments to be labelled are identified in the ASL gloss in Figure 22. Additionally, the movement epenthesis, or the movement segment between the last part of one sign and the first part of the next sign [94] was also labelled carefully. The labelling task was very time consuming since each sentence has 10 segments, including the movement epenthesis, that needed to be labelled. On the positive side, the labelling tool was very helpful in producing reliable label data by providing a custom video player option with speed control, allowing each segment to be viewed separately from start frame to end frame. Additionally, warnings were generated when certain types of labelling errors were made, such as in the case when two segments were labelled with overlapping frame boundaries or when the same label was given to two different segments.

Figure 26 shows a screenshot of the the interface of the labelling tool. The top section has three viewing panels. The panels on the left and right allow for the selection of the start and the end frames of the segment, respectively. The panel in the middle is a custom video player that allows for the segment to be viewed from start frame to end frame at different speeds. The “Add Label” button pops up a form, shown in Figure 27, that provides options to select a label for the segment. First the handedness is selected indicating whether it was a left-hand or right-hand sign or a two-handed sign. Next, one label is selected from one of the four categories: signs, classifiers, predicates or other (for movement epenthesis). Once a segment has been labelled, it is added to the “Snippets” panel. Each snippet can then be individually selected to become the current active segment displayed, which then activates the “Update” option allowing for a segment to be renamed or frame boundaries to be altered and saved. The snippet can be permanently deleted from the Snippets panel by clicking the delete button on the top right corner of the snippet thumbnail. Once all segments have been labelled, segment information can be written to disk by clicking the “Save” button.


Select a label

Select handedness and a label for the snippet.

Handedness
- Right Handed
- Left Handed
- Two Handed

Labels

Signs		Classifiers	Predicates
Boy	Cat	Person	Meet
Girl	Wolf	Animal	Weaves and bumps into
Man	Car	Vehicle	Bumps into
Woman	Fence	Four Hand	Bump into each other
Lion	House	Five Claw	Jumps over

Other
- Movement Epenthesis
- Coarticulation
- Custom

Figure 27: Add label form to select an appropriate label for the segment. First select handedness and then select one label from one of four categories: Signs, Classifiers, Predicates or Other.

6.1.4 Findings

Although we did not impose any restrictions on the handedness of the signers, all five signers were right-hand dominant. For most sentences in the dataset all signers agreed upon the structure and followed the sequence as given in the ASL gloss in Figure 22 where the sentence begins with the direct-object signed with the left hand followed by the subject with the right hand and finally the classifier predicate. However, there were some exceptions particularly for sentences constructed using the classifier predicate MEETS. Examples of these sentences include those that have BOY, GIRL, MAN and WOMAN as the direct-object and the subject noun such as BOY MEETS GIRL, MAN MEETS WOMAN and so on with a total of 16 possible sentences. The five variations that were seen in the dataset are shown in Figure 28. The first variant listed as ASL Gloss-I is the original structure from Figure 22. The variations arise due to three variables namely the start sign (is the direct-object or the subject signed first?), the hand used for the start sign (left hand or

ASL Gloss-I: <direct-object$_{LH}$> <classifier$_{doLH}$><verb of location> <subject$_{RH}$> <classifier$_{subjRH}$>-<MEETS-v1>

ASL Gloss-II: <direct-object$_{LH}$> <classifier$_{doLH}$><verb of location> <subject$_{RH}$> <classifier$_{subjRH}$>-<MEETS-v2>

ASL Gloss-III: <direct-object$_{RH}$> <classifier$_{doRH}$><verb of location> <subject$_{LH}$> <classifier$_{subjLH}$>-<MEETS-v2>

ASL Gloss-IV: <subject$_{RH}$> <classifier$_{subjRH}$><verb of location> <direct-object$_{LH}$> <classifier$_{doLH}$>-<MEETS-v2>

ASL Gloss-V: <subject$_{LH}$> <classifier$_{subjLH}$><verb of location> <direct-object$_{RH}$> <classifier$_{doRH}$>-<MEETS-v2>

Figure 28: The five variant sentence structures found in the Classifier-Kinect dataset used in conjunction with the MEETS classifier predicate. (LH - left hand, RH - right hand, do - direct object, subj - subject, MEETS-v1 - originally intended MEETS classifier predicate, MEETS-v2 - variant of the MEETS classifier predicate).

right hand?), which form of MEETS is signed (MEETS-v1 or MEETS-v2). The difference between MEETS-v1 and MEETS-v2 is illustrated in Figure 29 and 30. In the first case the subject moves towards the direct-object while the direct-object remains stationary. In the second case the subject and the direct-object move towards each other.

The variations shown in Figure 28 stem from varying interpretations of the English sentence that was presented to the signer as a prompt to sign. The English sentences that were presented took the form A MEETS B. When the team chose MEETS as one of the classifier predicates the intended meaning was for the direct-object noun to stay stationary in space while the subject noun moves to where the direct-object is located (i.e. MEETS-v1), for example in the case of BOY MEETS GIRL the girl is standing at a location and does not move while the boy goes over to meet the girl. This sequence is shown in Figure 29. However, the most common interpretation by all signers was to bring the direct-object and the subject towards each other (MEETS-v2) by moving both hands as show in the sequence in Figure 30.

It appears that this alternate interpretation of MEETS does not impose the starting order show in Figure 22 in which the sentence always begins with the direct-object noun signed with the left hand. Signers chose to start either with the direct-object or the subject-noun and switched hands as well, which suggests that there was no distinction between subject and direct-object. Post discovery, in a discussion with the design team the signers revealed that the English sentences presented to them before signing were not specific enough to

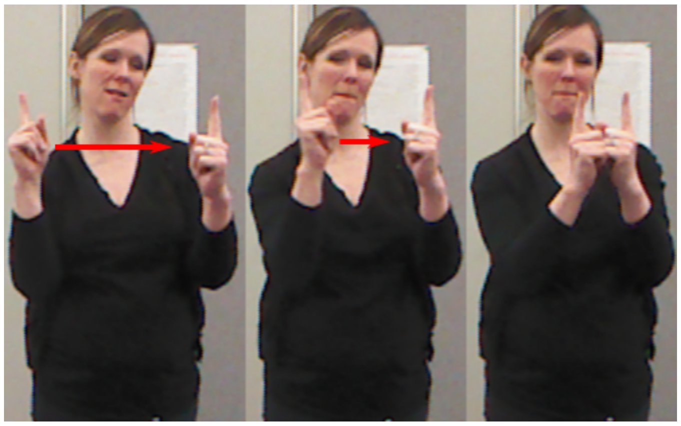


Figure 29: MEETS-v1: The predicate MEETS as intended in the dataset design where subject moves towards direct-object while direct-object remains stationary. Arrow indicates direction of movement.

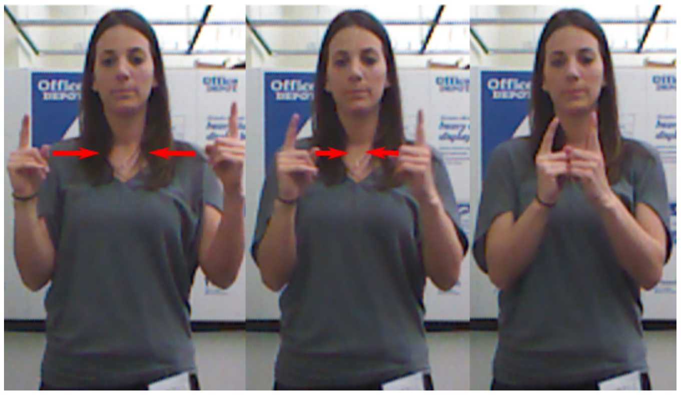


Figure 30: MEETS-v2: The alternate form of predicate MEETS where subject and direct object move towards each other. Arrows indicates direction of movement.

elicit the response the team had desired. Moreover, with the alternate interpretation the signers began the sentence with the sign/hand that felt more natural to them.

Table 21 shows the number of sentences, with the classifier predicate MEETS, per signer for each of the five sentence structures shown in Figure 28. We find that Signer 1, 2 and 4 pick one variant and are consistent in their signing. Signer 5 is mostly consistent by using the originally intended sentence structure from Figure 22 but changes the structure in one case. Signer 3 is the most inconsistent using three different variations only two of which match the original structure.

Besides the above-mentioned variations in the sentence structure for sentences with classifier predicate MEETS, we find another interesting variation in the case of the classifier

Table 21: Number of signed instances across signers for each variation in sentences that have the classifier predicate MEETS.

Variants †	Signers				
	1	2	3	4	5
I	0	0	2	0	15
II	0	0	11	0	0
III	0	0	3	0	0
IV	16 *	16 *	0	0	1
V	0	0	0	16 *	0

†see Figure 28 for each variant's sentence structure.

∗ There are a total of 16 sentences in Classifier-Kinect dataset that have the classifier predicate MEETS.

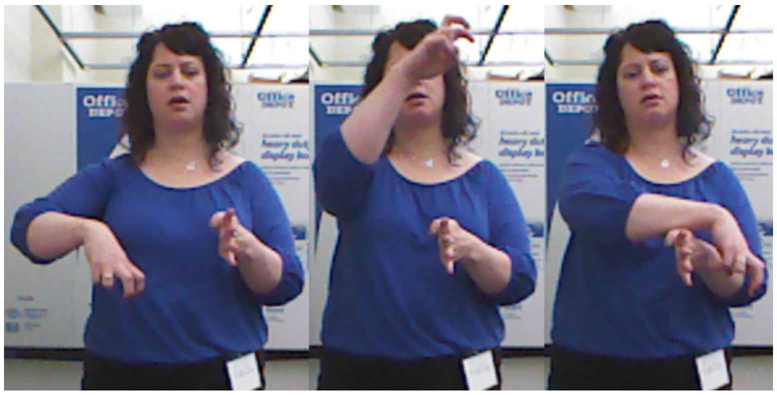


Figure 31: The classifier predicate JUMPS signed identically for both PERSON and ANIMAL subject nouns.

handshape used for the PERSON class that includes the signs BOY, GIRL, MAN and WOMAN. Typically the CL-1 handshape is use to represent members of the PERSON class, but with overwhelming agreement all signers used the CL-V-BENT handshape, typically used for the ANIMAL class, to sign the classifier predicate JUMPS. In a discussion with the signers it was determined that it is not mandatory to use the same classifier handshape for a given class of objects all the time. The signer chooses the appropriate handshape depending on the situation in order to convey the accurate meaning. In this case the act of jumping is more meaningfully represented by the CL-V-BENT handshape to subtly indicate that the person jumped, bent at the knees, much like an animal. Figure 31 shows the classifier predicate JUMPS which is identically signed irrespective of whether the subject noun is a PERSON or an ANIMAL.

6.2 Methodology for recognition of ASL classifier sentences in the Classifier-Kinect dataset

We illustrate the three high-level steps involved in the recognition of ASL classifier sentences in the Classifier-Kinect dataset using the graphic shown in Figure 32. Each individual step is described in detail below:

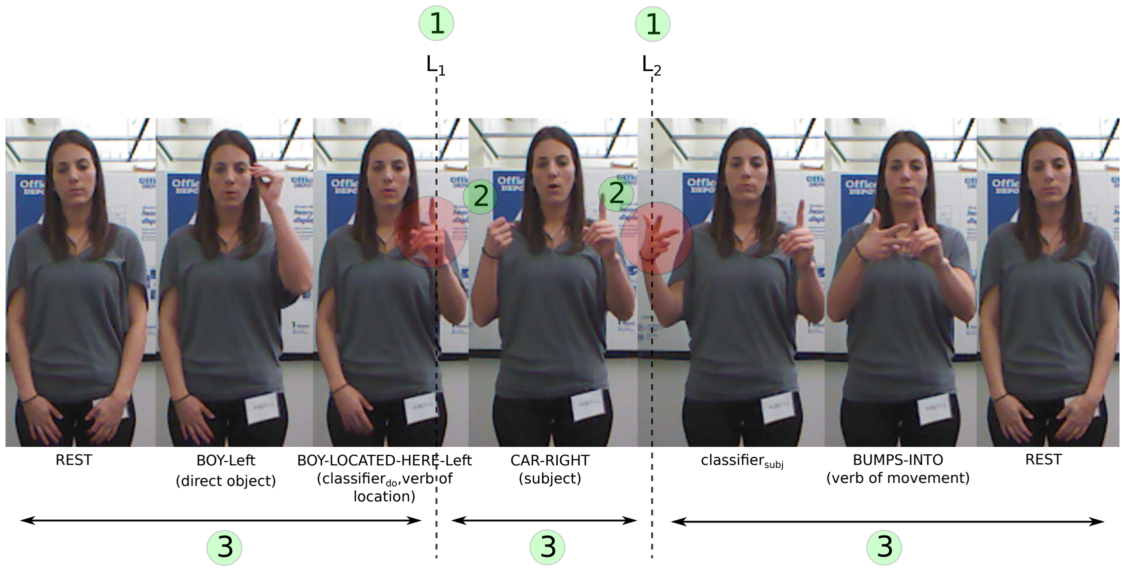

1
1
L_1
L_2
2
2
REST
BOY-Left (direct object)
BOY-LOCATED-HERE-Left (classifier$_{do}$,verb of location)
CAR-RIGHT (subject)
classifier$_{subj}$
BUMPS-INTO (verb of movement)
REST
3
3
3

Figure 32: Three steps in classifier sentence recognition: 1) segment sentence by identifying landmarks L_1 and L_2, 2) handshape classification and 3) sign and verb-of-movement classifier predicate recognition. (do - direct object, subj - subject).

1. *Identify landmarks to segment the classifier sentence.*

   We first identify the two landmarks denoted as L_1 and L_2 in Figure 32. L_1 is identified by spotting a long pause with no movement for one of the hands. L_1 is the point in the time line when this long pause begins, i.e. the point when the verb of location is signed. Then we look for a point that occurs after L_1 in the time line that indicates a rapid acceleration of the other hand towards the stationary hand, which is the start of the verb of movement. The point at the beginning of the acceleration is L_2. Determining L_1 and L_2 segments the timeline into three parts: the first corresponding to the direct object noun, the next to the subject noun and the last segment corresponding to the verb-of-movement classifier predicate.

2. *Identify handshape at the landmark points.*

   We expect that at landmarks L_1 and L_2 the observed handshape will be one of the five classifier handshapes. We classify the handshapes using a database matching technique that uses features derived from the localized depth data around the hand location. This technique is described in detail in Section 6.3.

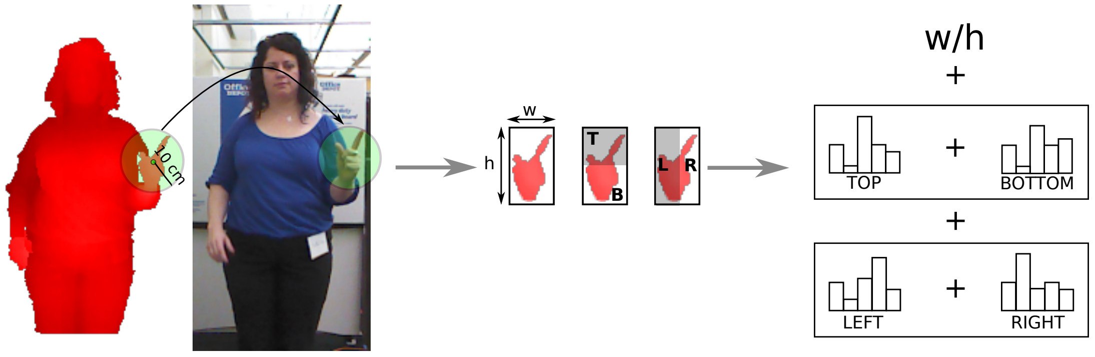
3. *Recognize nouns and the verb-of-movement classifier predicate using hidden Markov models* (HMMs)

   We train hidden Markov models using location- and movement- based features to recognize the subject and direct-object nouns and the verb-of-movement classifier predicate. The classifier handshapes identified in Step 2 help us narrow down the search for the noun sign in the case of PERSON (CL-1) and ANIMAL (CL-V-BENT) classifiers. For VEHICLE (CL-3), FENCE (CL-4) and HOUSE (CL-5-CLAW) the classifier handshape directly identifies the noun sign. In cases where there is uncertainty in the handshape classification we can perform a full search to recognize the noun signs. Details of sign and verb-of-movement classifier predicate recognition are given in Section 6.4.

6.3 Handshape classification to identify classifier handshapes

Once the classifier sentence has been segmented and the landmarks identified, we classify the handshapes seen at the two landmarks L_1 and L_2. Identifying the handshape can reduce the search space to recognize the noun signs using HMMs. If the handshape is CL-1 we know that the preceding direct object or subject noun is a PERSON either BOY, GIRL, MAN or WOMAN. If the handshape is CL-V-BENT then the noun is either ANIMAL or PERSON in cases where classifier predicate is JUMPS. In the Classifier-Kinect dataset CL-3, CL-4 and CL-5-claw directly identify the noun as CAR, FENCE and HOUSE respectively. To perform handshape classification we match each handshape against a database of handshapes and pick the majority class of the top ten matches. The matching is performed using features generated using depth information gathered within a radius of 10 cm from the hand location and eliminating pixels based on skin color.

6.3.1 Feature extraction



a) Extract depth pixels from a radius of 10 cm and eliminate unwanted pixels by matching against skin color model.

b) Compute bounding box of the extracted handshape. Divide into four regions: top and bottom, left and right.

c) Build feature vector by combining width to height ratio with a 5-bin histogram of depth values in each of the four regions.

Figure 33: Feature extraction for handshape classification.

To generate features for handshape classification we employ both the color image as well as the depth image that is registered with the color image. Beginning with the hand location in the depth image, all depth image pixels within a three-dimensional radius of 10 cm are collected. Those points whose corresponding color image pixel does not satisfy our skin color model are filtered out. This process eliminates background pixels and pixels on the body of the signer leaving only those that belong exclusively to the hand (see Figure 33a). We then construct a 21-dimensional feature vector where the first dimension is the ratio of the width to height of the bounding box of the hand pixels and remaining dimensions correspond to four five-dimensional histograms of depth values of the hand in the top, bottom, left and right regions respectively (see Figure 33b & c).

6.3.2 Experimental results

We use a database matching approach to classify the classifier handshapes. The database contains a list of features that were computed for the hand shapes using the method described in the previous section and the corresponding class labels. Cross-validation is done in a user-adaptive manner since the number of signers in the dataset is low (N=5). We create five separate reference databases for each signer whereby the reference database contains some examples from the test signer's data, specifically those taken from the first occurrence of a each classifier. Those examples that are added to the reference databases are left out from the test set. User-adaptation is a reasonable approach to employ, even during live recognition within an application or a game, when the training data has limited number of signers. The signer can be asked to sign a few representative examples of ASL classifiers including the handshapes, which are then subject to additional processing to recalibrate the system so as to boost the performance in future cases for the same signer. The amount of time spent in collecting and processing the new examples is a small price to pay in order to achieve better performance. In contrast, a system that has been trained with a large number of signers, thus capturing a wide variety of signing variations, may not require user-adaptation.

To match features we use the *L2* distance measure. Since we already know, from the spotting of L_1 and L_2 landmarks (see Figure 32), if the query hand is the left or the right hand, we only match left handshapes against a reference database of left handshapes and right handshapes against a reference database of right handshapes thereby eliminating confusion between the set of right handshapes and the set of left handshapes during classification.

In the first step, the features in the database are sorted in ascending order of distance to the query handshape. Then, we select the top ten closest matches and examine their corresponding labels. The most occurring label is selected as the final result. If there is a tie then the classifier handshape is "undefined", which means we can no longer use a reduced search space for recognizing direct-object and subject nouns.

Table 22 gives the results for handshape classification accuracy for each classifier handshape across all signers. FENCE and HOUSE only have left handed versions for the classifier handshape (CL-4 and CL-5-CLAW) since they only occur as the direct-object in the dataset. As per convention for right-hand dominant signers, while using classifiers the direct-object is always placed in space using the left hand, vice versa for left-hand dominant signers. Since all the five signers in our dataset were right-hand dominant we did not observe the right hand versions of the classifiers CL-4 and CL-5-CLAW. HOUSE-Left has the highest

Table 22: Percentage accuracy of handshape classification for each signer.

		Signers					
Classifier	Handshape	#1	#2	#3	#4	#5	Overall%
PERSON-Right	CL-1-R	84	81	79	86	82	82%
PERSON-Left	CL-1-R	86	84	82	88	84	85%
ANIMAL-Right	CL-V-BENT-R	79	78	74	76	75	76%
ANIMAL-Left	CL-V-BENT-L	77	79	76	81	80	79%
VEHICLE-Right	CL-3-R	84	84	82	86	81	84%
VEHICLE-Left	CL-3-L	79	76	69	64	74	72%
FENCE-Left	CL-4-L	77	74	68	66	76	72%
HOUSE-Left	CL-5-CLAW-L	87	86	86	90	84	87%

Table 23: Confusion matrix for classifier handshape classification. (The CL- prefix has been omitted from column names due to lack of space).

Classifier	1-R	1-L	V-BENT-R	V-BENT-L	3-R	3-L	4-L	5-CLAW-L	undefined
CL-1-R	82	0	12	0	4	0	0	0	2
CL-1-L	0	85	0	11	0	3	0	0	1
CL-V-BENT-R	15	0	76	0	5	0	0	0	4
CL-V-BENT-L	0	14	0	79	0	4	0	1	2
CL-3-R	6	0	7	0	84	0	0	0	3
CL-3-L	0	1	0	1	0	72	20	2	4
CL-4-L	0	2	0	3	0	19	72	1	3
CL-5-CLAW-L	0	2	0	4	0	5	0	87	2

overall accuracy whereas FENCE-Left and VEHICLE-Left have the least overall accuracy.

Table 23 gives the confusion matrix for the classifier handshapes. We include the label "undefined" to indicate when there is a tie between two or more output labels for the top spot. We notice right away that two hand shapes that had least accuracy, VEHICLE-Left (CL-3-L) and FENCE-Left (CL-4-L), have the most confusion amongst each other. Significant confusion also occurs in the case CL-1 and CL-V-BENT.

6.3.3 Findings

We find that for VEHICLE-Left (CL-3-L) and FENCE-Left (CL-4-L), the two handshapes with least accuracy, confusion mainly arises when the signer holds the CL-3 handshape with fingers pointing towards camera instead of the back of the hand facing the camera, which happens when the noun CAR is used as a direct-object in examples such as BOY JUMPS OVER CAR. In this case the depth resolution is insufficient to make the distinction between

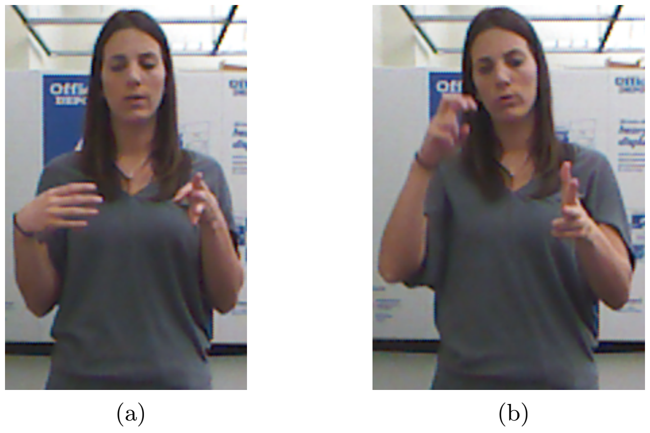


Figure 34: Comparison of (a) FENCE and (b) CAR classifier handshapes when used as direct-object (left hand).

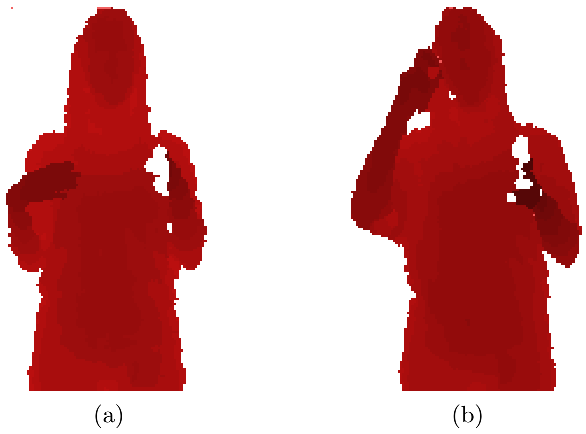


Figure 35: Comparison of depth images of (a) FENCE and (b) CAR classifier handshapes when used as direct-object (left hand).

whether two or four fingers, excluding the thumb, are outstretched in the case of CL-3-L and CL-4-L respectively. This case is highlighted in Figure 34.

If we closely examine Figure 35 we see that there are white patches around the hand that do not seem to have valid depth data. These are in fact “holes” that occur when the Kinect camera is unable to determine a depth value for that location. The problem occurs near the boundary of an object when there is significant depth disparity between the object and its background, which is generally the case with classifier handshapes. The origin of this problem can be attributed to the method used by the Kinect camera to compute depth. The Kinect uses an IR pattern to flood the scene as shown in Figure 36 and computes depth

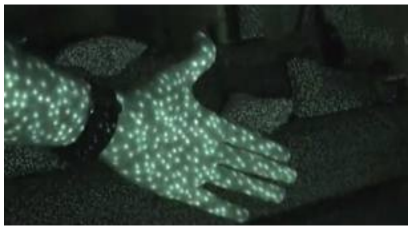


Figure 36: Kinect IR dot pattern used to estimate depth.

based on distortion of the scene pattern compared to a reference pattern. In boundary conditions two neighboring dots in the pattern fall on objects that have significant depth difference resulting in an unreliable depth value which is likely discarded resulting in a "hole." The problem is magnified for smaller objects at larger distances away from the Kinect camera. The size of the hole starts to become comparable to the size of the object, sometimes "swallowing" large parts of the object.

"Holes" in the depth data can pose a challenging problem for handshape classification, especially since the signers stand about 5 ft away from the camera to capture the entire signing space. Figure 35 show some examples where fingers and parts of the hand have been swallowed by holes, which clearly is a significant source for errors in handshape classification. In the next section we look at how to improve handshape classification by regenerating depth data in the holes.

6.3.4 Handshape classification with regenerated depth images

As mentioned earlier handshape classification using depth image data suffers from the problem of "holes" in the data around the boundary of the hand. Given the small hand size this can be a significant challenge for robust handshape classification. Since we have a pair of color and depth images to work with, we can use the color information to extract a mask to first determine the region of the hole that corresponds to the hand and then regrow that region with new depth data.

The process of determining what part of the hole belongs to the hand is illustrated in Figure 37. The process begins similar to Figure 33a except we do not discard the color information for which there is no depth data. We begin with the location of the hand and gather all color pixels within a three-dimensional radius of 10 cm. We then pick the largest

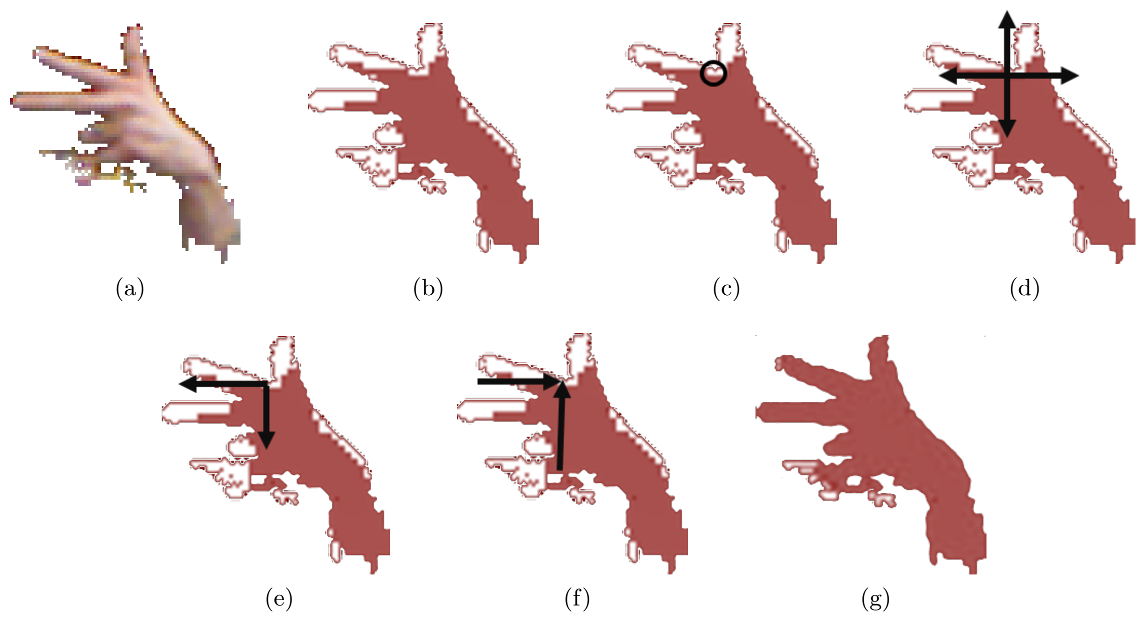


Figure 37: Process of regenerating depth values in "holes." a) color mask of the hand obtained by skin color segmentation b) depth pixels overlayed over color mask c) select an invalid pixel to begin regeneration d) examine local neighboring pixels in four principle directions e) perform linear regression to fit a line to neighboring pixels with valid data f) extrapolate values and compute average g) final result after regeneration.

blob of color pixels using connected components. Next we gather all data from the depth image corresponding to the color pixels including the invalid data in the "holes." Figure 37b shows the result of this process. Now that we know that the white regions (with invalid depth values) are actually part of the hand we can devise a process to regenerate the depth values in those regions.

We use local linear extrapolation to gradually regrow the hand region starting from the invalid pixels closest to pixels with valid depth data (Figure 37c). First, the neighboring pixels, within 2.5 cm distance in the four principle directions are collected (Figure 37d). If we are unable to find pixels in any particular direction that direction is discarded. Next we fit a line to the data in each remaining direction using linear regression (Figure 37e) and extrapolate a value for the new location. Finally, we average the values that were extrapolated from each direction (Figure 37f). This process is repeated for each pixel with an invalid depth value, gradually regenerating the depth values. The final result after regeneration is shown in Figure 37g.

Table 24: Percentage accuracy of handshape classification for each signer after regeneration of depth data.

		Signers					
Classifier	Handshape	#1	#2	#3	#4	#5	Overall%
PERSON-Right	CL-1-R	94	91	90	94	92	92%
PERSON-Left	CL-1-R	96	93	92	96	91	94%
ANIMAL-Right	CL-V-BENT-R	88	91	85	90	88	88%
ANIMAL-Left	CL-V-BENT-L	89	90	88	92	89	90%
VEHICLE-Right	CL-3-R	94	94	91	93	95	93%
VEHICLE-Left	CL-3-L	87	85	82	80	89	85%
FENCE-Left	CL-4-L	88	89	81	83	90	86%
HOUSE-Left	CL-5-CLAW-L	95	92	92	96	93	94%

Table 25: Confusion matrix for classifier handshape classification after regeneration of depth data. (The CL- prefix has been omitted from column names due to lack of space).

Classifier	1-R	1-L	V-BENT-R	V-BENT-L	3-R	3-L	4-L	5-CLAW-L	undefined
CL-1-R	92	0	6	0	2	0	0	0	0
CL-1-L	0	94	0	5	0	1	0	0	0
CL-V-BENT-R	8	0	88	0	3	0	0	0	1
CL-V-BENT-L	0	6	0	90	0	3	0	1	0
CL-3-R	3	0	4	0	93	0	0	0	0
CL-3-L	0	1	0	1	0	85	11	2	1
CL-4-L	0	1	0	2	0	10	86	0	1
CL-5-CLAW-L	0	1	0	2	0	3	0	94	0

6.3.4.1 Experimental results

Table 24 and 25 show the handshape classification accuracy and the confusion matrix respectively after performing handshape classification using the regenerated depth images. The process for feature extraction and database matching remain the same. We clearly see that the overall accuracy for all handshapes has improved compared to results in Table 22, and the number of "undefined" labels has significantly dropped. Moreover, the large confusion we saw earlier in Table 23 between VEHICEL-Left and FENCE-Left has significantly reduced.

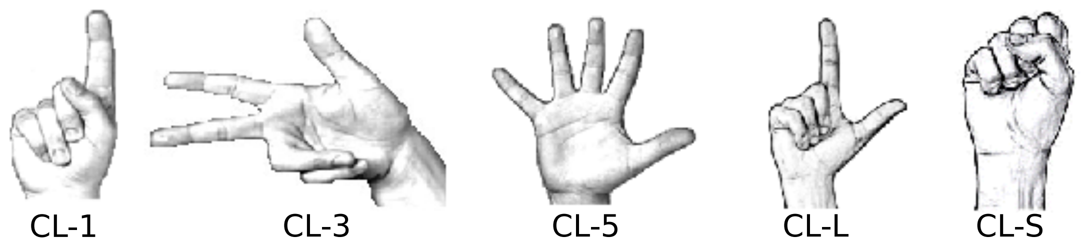


Figure 38: Proposed set of classifier handshapes with reduced minimal-pair confusion. (© 2004, www.Lifeprint.com. Used by permission.)

6.3.5 Discussion

Robust handshape classification is a critical step in our pipeline (see Figure 41) to recognize ASL classifier sentences. It can facilitate reduction of the output search space for recognition of direct-object and subject nouns thereby providing a boost in sentence recognition accuracy. However, we have limited depth map resolution at distances greater than 1m, which are required to capture the entire signing space, due to shortcomings in hardware technology. Moreover, the problem of "holes" in the depth data increases with distance away from the Kinect depth camera, which meant that although the output space for the classifer handshape was small (N=5) in the Classifier-Kinect dataset, by using our method for handshape classification we still encountered cases where there was considerable confusion among the minimal pairs like FENCE-LEFT and HOUSE-LEFT, PERSON-Right and ANIMAL-Right, and PERSON-LEFT and ANIMAL-LEFT. This problem was mitigated to some extent by employing a novel method to regenerate depth data in the "holes." We do not claim to have the best handshape classifier in the domain of ASLR, but it is safe to say that these problems will continue to exist until hardware technology improves to support high-resolution high-framerate depth cameras. Meanwhile, if our goal in building the ASL classifier recognition system is to support educational tools we may be able to mitigate these problems by better dataset design in which we reduce the occurrence of minimal pairs but still include a rich array of handshapes. Figure 38 shows an alternate set of five handshapes that we believe, based on our experience with the Microsoft Kinect depth camera, have fewer minimal pairs and will result in reduced confusion while classifying the handshapes.

6.4 Training hidden Markov models to recognize noun signs and verb-of-movement classifier predicates

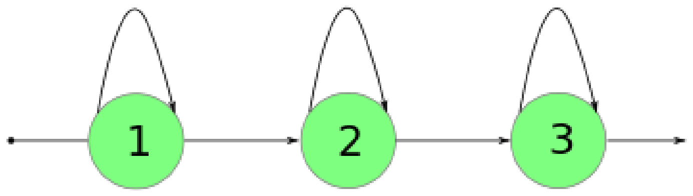


Figure 39: Three state left-to-right HMM.

To recognize the noun signs and classifier predicates we use hidden Markov models (HMMs). To train the HMMs we use the frame labels that were marked using the labelling tool (see Figure 26). For the purpose of recognition there is no difference in the process for recognizing the noun signs versus the verb-of-movement classifier predicates, although the features used for one-handed noun signs are different from the ones used for two-handed noun signs and classifier predicates. To model the 15 nouns signs and classifiers a 3-state left-to-right HMM is used (see Figure 39). Training and testing is done using the GT2K and HTK (HMM tool kit) [102, 113].

6.4.1 Feature extraction

Features for HMM training are based on hand location and movement. The hand locations are obtained using the Microsoft Kinect Skeleton Tracker (MKST). The hand locations are first transformed into signer coordinate space by subtracting the signer origin from the hand coordinates. The signer origin was marked in the first frame of the sentence and corresponds to the midpoint between the eyes of the signer. The x, y and z coordinates of the hand are then normalized based on the arm lengths of the signer. The process to compute the divisors for normalization is illustrated in Figure 40.

We use two types of features for training the hidden Markov models. The list of features is given in Table 26. For one-handed signs we use the normalized three-dimensional location of the hand (L_x L_y L_z for left-hand signs and R_x R_y R_z for right-hand signs) along with the first order derivatives in the X, Y and Z directions computed based on the current location and the location in the previous frame (ΔL_x ΔL_y ΔL_z and ΔR_x ΔR_y ΔR_z). For two handed

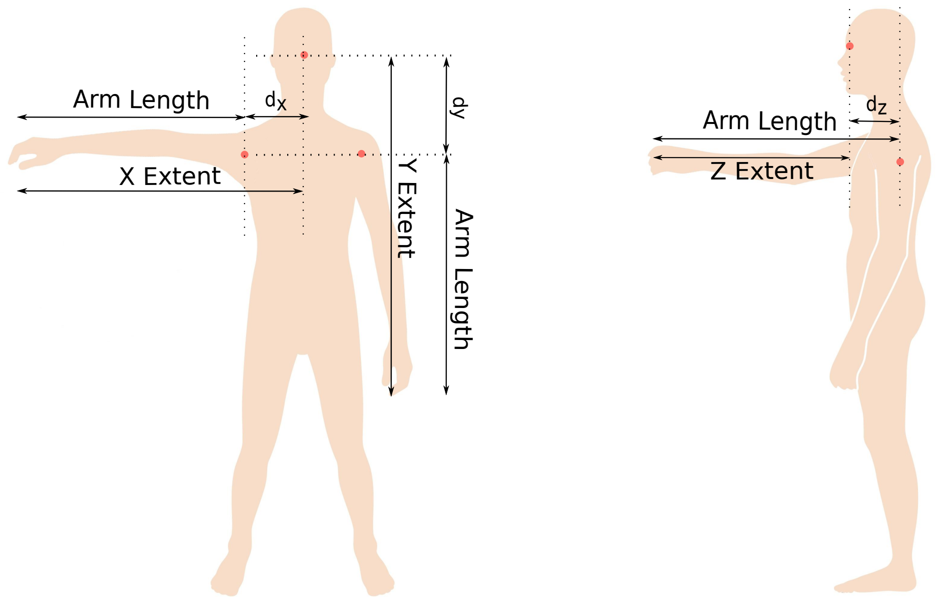


Figure 40: Process of normalizing three-dimensional hand coordinates using X-, Y- and Z-extents computed based on arm lengths.

Table 26: Features for training hidden Markov models to recognize noun signs and verb-of-movement classifier predicates.

Sign Type	Features
Right-hand nouns	R_x R_y R_z ΔR_x ΔR_y ΔR_z
Left-hand nouns	L_x L_y L_z ΔL_x ΔL_y ΔL_z
Two-handed nouns	R_x R_y R_z L_x-R_x L_y-R_y L_z-R_z
Verb-of-movement classifier predicates	R_x R_y R_z L_x-R_x L_y-R_y L_z-R_z

signs and verb-of-movement classifier predicates the features include the three-dimensional location of the right hand (R_x R_y R_z) and the location of the left hand relative to the right hand (L_x-R_x L_y-R_y L_z-R_z).

6.5 Sentence recognition results on the Classifier-Kinect dataset

For sentence recognition we follow the high-level three-step process outlined in Figure 32. The actual recognition pipeline that is executed is shown in Figure 41. The pipeline proceeds as follows: First we spot the landmark L_1 corresponding to the direct-object classifier. Once this is done we can in parallel identify the handshape for the direct-object classifier and spot the landmark L_2 which occurs later in the timeline compared to landmark L_1. At this point since we have identified both landmarks and segmented the sentence we can take

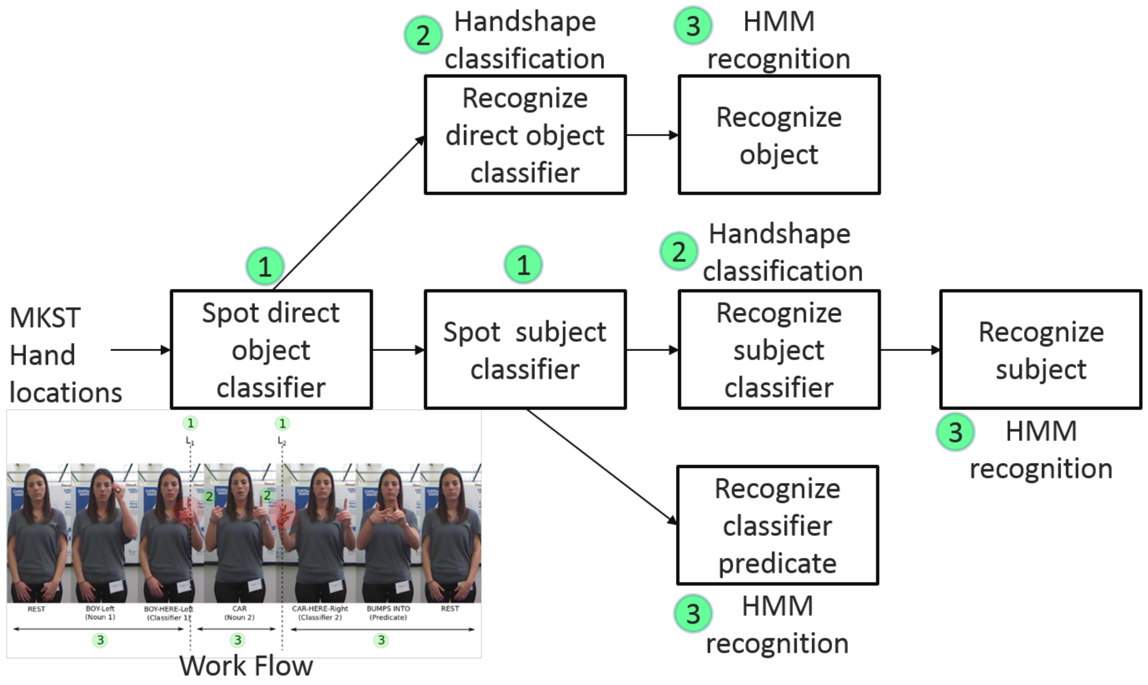

2 Handshape classification
3 HMM recognition
Recognize direct object classifier
Recognize object
2 Handshape classification
MKST Hand locations
1
Spot direct object classifier
1
Spot subject classifier
Recognize subject classifier
Recognize subject
3 HMM recognition
Recognize classifier predicate
3 HMM recognition
1 L1
1 L2
2
2
REST
BOY-Left (Noun 1)
BOY-HERE-Left (Classifier 1)
CAR (Noun 2)
CAR-HERE-Right (Classifier 2)
BUMPS INTO (Predicate)
REST
3
3
3
Work Flow

Figure 41: ASL classifier sentence recognition pipeline.

three actions in parallel; identifying the handshape for the subject noun classifier and HMM-based recognition of the direct-object noun and the verb-of-movement classifier predicate. The final step is the HMM-based recognition of the subject noun. At the two points in the pipeline where handshape classification is performed if there is uncertainty in the determination of the handshape, i.e. if the actual handshape is "undefined" because there were two or more class labels contending for the top spot during the database matching, we attempt to do full scale recognition on the sentence segment to determine which one of the 10 noun signs it is most likely to be. The experiments for handshape classification have been already been outlined in section 6.3.2 and were conducted using user-adaptive cross-validation given that there were many variations in the handshapes across the five signers making it harder to get good results using signer-independent cross-validation. However, for recognizing noun signs and verb-of-movement classifier predicates we perform signer-independent cross-validation since the features for recognition are based on hand location and movement and are scale-invariant due to normalization based on the signer's arm length. Signer-independent cross-validation in this case is more suitable to determine the robustness

Table 27: Sentence recognition accuracy. Experimental results on the Classifer-Kinect dataset using two different techniques each for hand detection and handshape classification.

		Signers					
Hand detection	Handshape classification	#1	#2	#3	#4	#5	Overall%
Ground truth	Oracle	96.5	93.0	91.2	95.1	94.2	93.4%
Ground truth	Section 6.3.4	90.9	87.2	84.8	90.1	87.5	88.1%
MKST	Oracle	84.9	82.6	78.4	83.7	81.9	82.3%
MKST	Section 6.3.4	80.3	77.5	70.9	77.4	76.2	76.4%

of the recognizer.

Table 27 shows the results of sentence recognition across all users obtained using the recognition pipeline shown in Figure 41. We select either the ground truth data or the MKST as the source for hand locations. For handshape classification we compare the technique from Section 6.3.4, whereby a novel approach was used to regenerate depth data in the holes, to an oracle handshape classifier that reports the correct answer 100% of the time. We notice that when ground truth hand locations are used the handshape classification errors cause the overall accuracy to drop by 5.3 percentage points compared to a drop of 5.9 percentage points when the MKST is used.

6.5.1 Discussion

Looking back at the sentence recognition pipeline we find that there are two factors that influence the sentence recognition accuracy. First is the handshape classification of the direct-object and the subject classifiers. Both the classification steps occur early on in the pipeline; an error in handshape classification is multiplicative as it propagates through the pipeline. We rely on the handshape classification to reduce the search space for performing recognition of the direct-object and subject nouns. This reliance can be counterproductive since an error in handshape classification always results in an error in the recognition steps; the one exception being cases in which the handshape is "undefined" resulting in full-scale recognition whereby there is still a possibility of correctly recognizing the noun signs. The second factor that can influence sentence recognition accuracy is accuracy of hand tracking or detection. For HMM-based recognition of the noun signs and the verb-of-movement classifier predicate the features are based on location and movement of the hands. Errors in hand tracking or detection will negatively affect recognition accuracy. On closer examination we find that there is a more complex interrelationship between the two factors we just mentioned. Figure 42 illustrates this relationship. An arrow indicates impact on results. We know that handshape classification directly affects the recognition accuracy of noun signs. The influence of hand tracking or detection is both direct and indirect. For HMM-based recognition of verb-of-movement classifier predicates better hand tracking will directly result in better recognition accuracy. However, the influence of hand tracking on recognition of noun signs is more complex. Better hand tracking can directly have a positive impact on noun recognition but can also have an indirect influence whereby it first improves handshape classification by providing a more reliable location of the hand. This better location enables better extraction of the hand region, which positively impacts handshape classification.

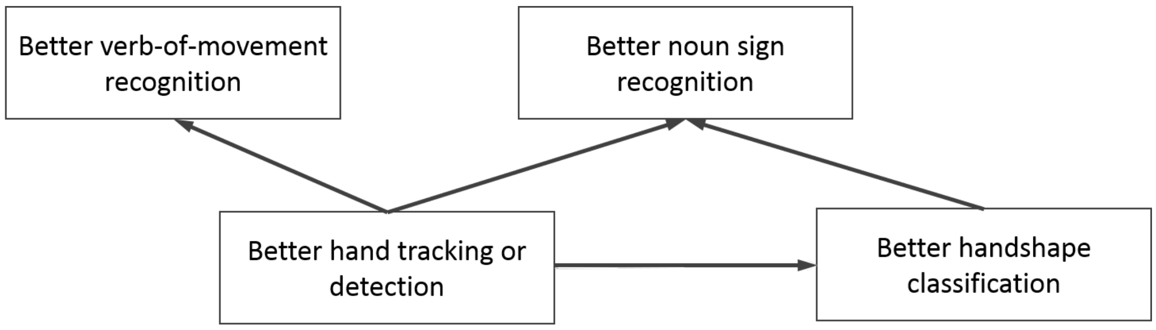

Better verb-of-movement recognition
Better noun sign recognition
Better hand tracking or detection
Better handshape classification

Figure 42: Interrelationship between factors that impact sentence recognition accuracy. An arrow indicates impact on results.

Of the two factors discussed here, in Section 6.3.4 we have already explored the idea of improving handshape classification accuracy by addressing the problem of “holes” in the depth data and performing handshape classification after regenerating missing depth data. In the next chapter we will address the issue of improving the overall sentence recognition accuracy by building a hand detector that performs better than the Microsoft Kinect Skeleton Tracker (MKST).

CHAPTER VII

IMPROVING CLASSIFIER SENTENCE RECOGNITION WITH BETTER HAND DETECTION USING DOMAIN-DRIVEN RANDOM FOREST REGRESSION

Robust hand tracking or detection is a crucial element in several real world applications such as gaming, gesture control input, and automatic sign language recognition (ASLR). In American Sign Language, signs are distinguished based on various parameters such as hand movement, location, shape, orientation and non-manual signals such as facial expressions [94]. In ASLR systems we find that hand location and movement is an important and descriptive feature [85, 106]. Not only is hand location important, the accuracy of reported locations directly impacts the determination of hand pose.

In the past there have been several approaches to hand tracking and detection, some using data gloves and sensors while others use computer vision approaches which use camera systems leveraging color information in a two dimensional space [12, 17, 37, 62, 69]. The advent of commercially available inexpensive depth cameras has given rise to a greater capability to utilize elements of body pose and provide rich information towards this end. One of the best available solutions in body pose recognition is the Microsoft Kinect Skeleton tracker (MKST) [81]. This tracker has been trained on generic datasets, with training data of real world human poses as well as synthesized data for greater variation. This training however, may not be robust enough for more specific problems, like ASLR.

The use of depth cameras in ASL applications has been widely demonstrated, such as in cases of isolated sign recognition, sentence recognition and hand shape classification [18, 77, 115]. However, our observations have shown that on real world continuous ASL data the MKST fails in cases where the hands are close to the body, close to or touching each other, or when the arms cross. The MKST, therefore, is not a robust or precise enough tracker for the ASL domain.

We present a detailed analysis of the errors of the MKST. We also propose a novel method to predict 3D positions of the left and right hand in depth images of ASL signing data, using domain-driven random forest regression. Our goal is not to create a perfect and generic hand detection scheme, but to develop a more robust and precise hand detector to benefit the work of ASLR research, specifically the recognition of ASL classifiers.

7.1 Ground truth data for hand locations

The ground truth data for the location of the hands was obtained by manually labelling the location of the left and right hand in 74474 color images across all 488 sentences in the Classifier-Kinect dataset. A labelling tool was developed that recorded the x and y coordinates of a mouse click on the color image. Additionally, the corresponding depth value at that location in the depth map was also recorded. As we have seen earlier in Section 6.3.3 the depth maps from the Kinect camera are not as reliable as the color images in providing a complete image without missing data, which can pose a problem during labelling. If labelling is done solely using the color images, and we blindly reference into the depth map to obtain the depth value, we may inadvertently mark the location of the hand in the color image at a location where the depth value is invalid although there maybe a close by point with a valid depth value that will serve equally well to indicate the x and y coordinate of the hand. In order that we don't introduce errors because of the above-mentioned problem the depth map is superimposed over the color image as a transparent layer making it easy to select points with valid depth values to mark the location of the hand. This process is illustrated in Figure 43. Figure 43a shows that the initial choice to mark the location of the left hand maps to an invalid depth value (Figure 43b). However, with superimposition of the depth map we are able to mark a better location with a valid depth value (Figure 43c). There were rare cases in the dataset in which the entire hand was missing due to a "hole" in the depth data, in which case the closest point with a valid depth value that lay on the extreme end of the arm was chosen.

Note that only the locations were recorded, and not the bounding box for the hand, since the method we developed for hand detection does not depend on the appearance of

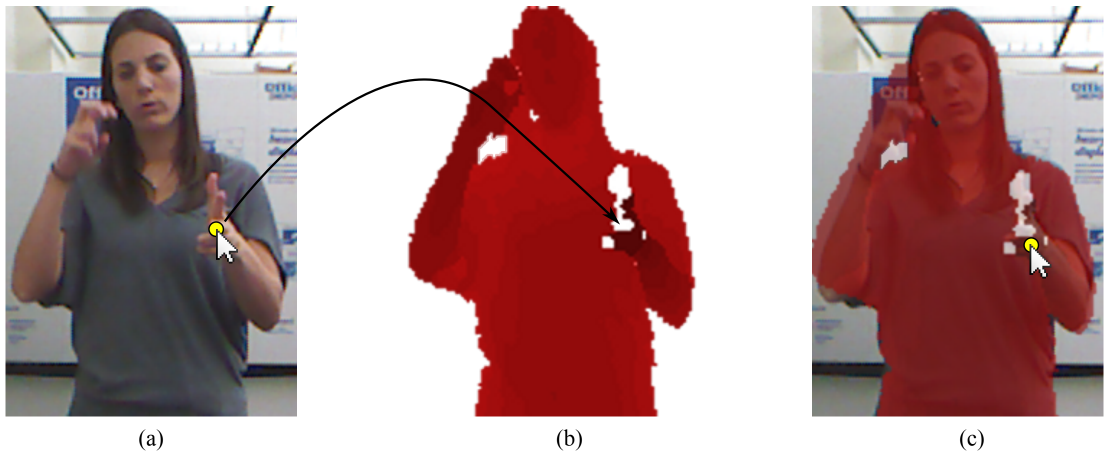
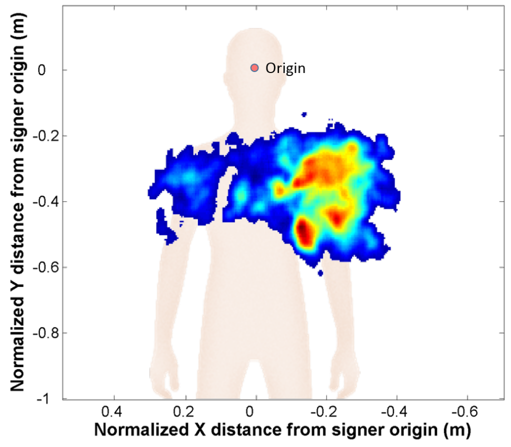


Figure 43: Labelling ground truth hand locations in the Classifier-Kinect dataset - (a) initial location of hand in color image (b) invalid depth value in the corresponding location in the depth map (c) a better choice is found, with the superimposed depth map, close to the initial selection but with a valid depth value.

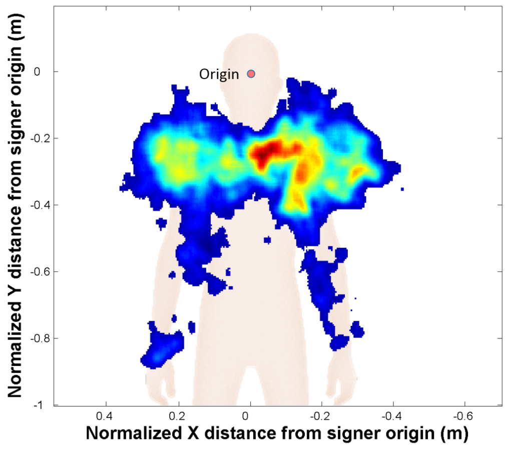
the hand. We found that the most efficient way to label the hand locations was to focus on one hand at a time. First, the right hand was labelled in all images followed by the left hand, thus involving a total of 2*74474 mouse clicks, quite an arduous task indeed. Finally, two rounds of visual inspection, at real time playback speed, were undertaken to correct any errors that occurred during labelling. Once ground truth image coordinates were collected they were mapped to camera coordinate space.

7.1.1 Skeleton Tracking Errors

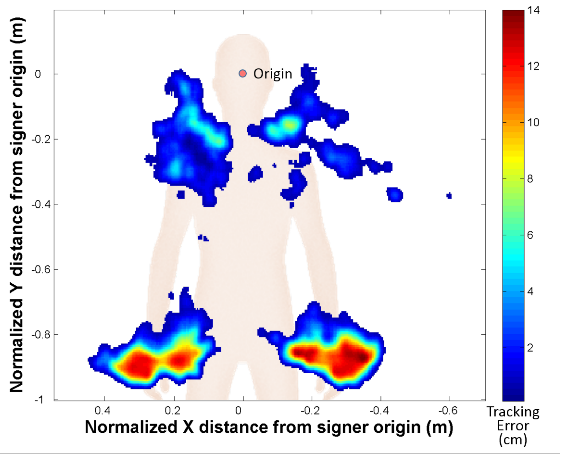
Figure 44 shows a heat map visualization of the skeleton tracking errors for the Classifier-Kinect dataset. Consider Figures 44a through 44c, which show the average tracking error at different locations in the signing space. The magnitude of average error varies from 0.1 cm (blue) to 14 cm or greater (red). Each figure represents a cross-section of the signing space in depth. This concept is better illustrated in Figure 45, which shows the perspective view of cross-sections. Figure 44a shows a cross-section of thickness 0.6 arm-lengths from fully outstretched down to 40% of the length. Figure 44b shows a cross-section of thickness 0.2 arm-lengths that extends from 20% to 40% of the arm length in front of the body. Figure 44c encompasses everything behind the body and up to 20% of the arm length in front of the body. The regions in white are free of data.



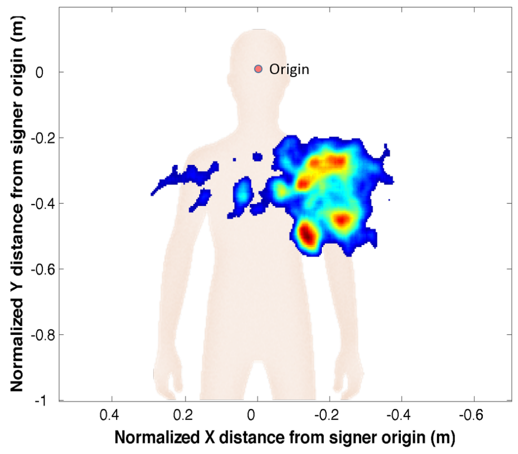
(a) depth = (-1,-0.4)



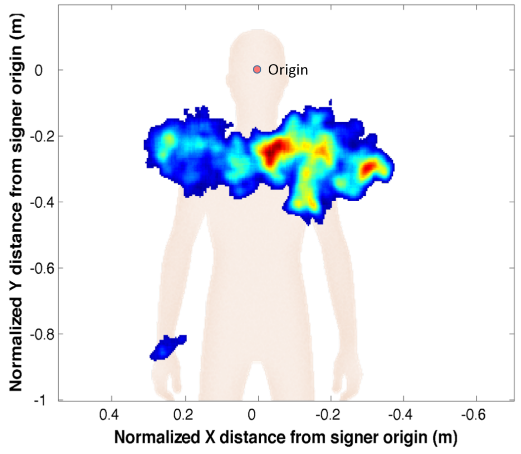
(b) depth = (-0.4,-0.2)



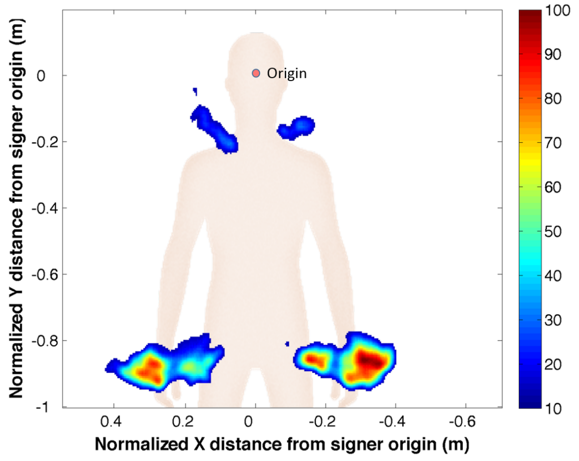
(c) depth = (-0.2, 1)



(d) depth = (-1,-0.4)



(e) depth = (-0.4,-0.2)



(f) depth = (-0.2, 1)

Figure 44: Heat maps of the errors by MKST. (a)-(c) depict average error with blue being almost no error to red, an error of 14 cm or greater. (d)-(f) depict likelihood of errors with blue being low chance of error to red being the locations of most frequent error.

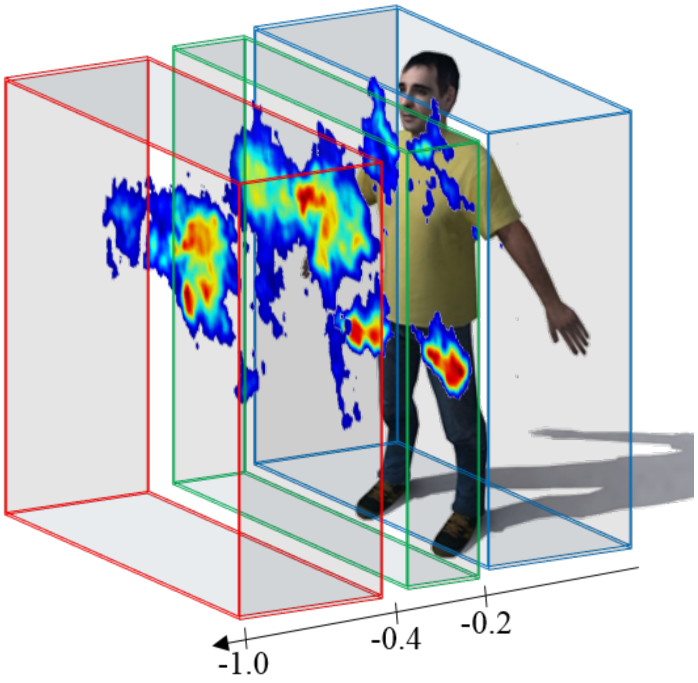
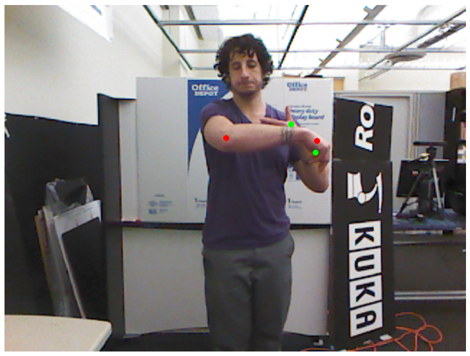


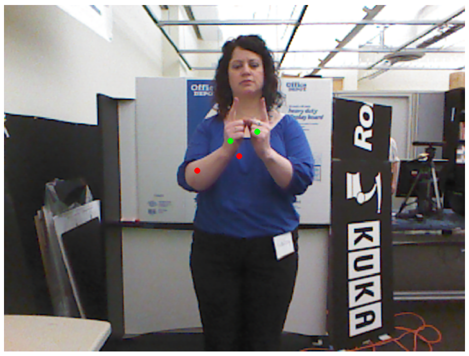
Figure 45: Perspective view of cross-sections in signing space showing hand tracking errors of the Microsoft Kinect Skeleton Tracker. Cross-sections are shown separated for clarity.

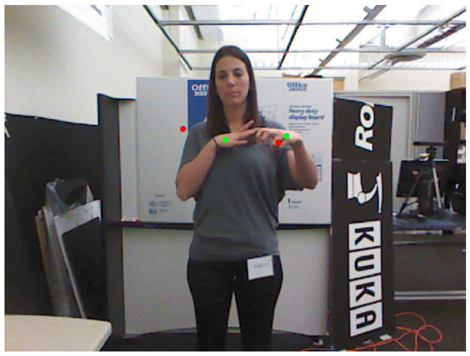
Figures 44d through 44f show the likelihood of tracking error at different locations in the signing space. Blue indicates low chance of error, whereas red indicates high change of error. Figure 46 illustrates three fundamental types of tracking errors for the Microsoft Kienct Skeleton Tracker (MKST) that we have encountered in the Classifier-Kinect dataset. It shows the ground truth labels and the tracked points by MKST in green and red respectively. The tracking errors typically occur in the following situations:

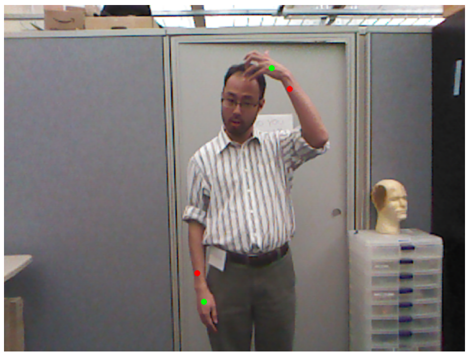
- Interaction between hands or crossed arms near or slightly away from the signer's body
- Interaction of signer's hands with the body or the face
- Hands at rest, beside the signer's body

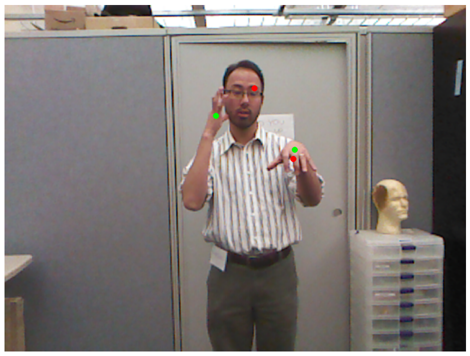
The first instance represents errors that occur when the hands interact with each other away from the body in the signing space. The Classifier-Kinect datset, and ASL in general, has numerous such signs. An example from our dataset is in Figure 46a where the verb-of-movement classifier predicate is JUMPS. In our dataset, the pool of signers all being


(a)


(b)


(c)


(d)


(e)

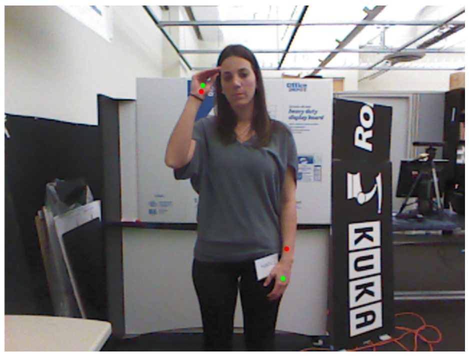

(f)

Figure 46: Examples of tracking errors by the Microsoft Kinect Skeleton Tracker in the Classifier-Knect dataset.

right-hand dominant, such verbs of movement terminate on the left hand side of the body, where the direct-object noun is located. Figures 44a and 44d show the heatmaps of errors in the region which is approximately occupied by such signs which have a close interaction between the hands. It is evident that the MKST mis-tracks most often and by the most amount in the areas where such interaction is taking place.

A closely related type of error is when the hands of the signer interact with each other close to the body. This error is visualized in Figures 44b and 44e. Figure 46b shows one example from the dataset wherein the classifier predicate is MEETS. Again, we can see that the MKST fails at crucial times and the magnitude of failure is also high.

Other errors occur when the hands interact with the head or face of the signer. The magnitude and frequency of such errors (top half of Figures 44c and 44f) are lower compared to the other kinds but not negligible. Examples of these errors from our dataset are in Figure 46c (right hand), Figure 46d (left hand) and Figure 46e (right hand).

Finally, the bottom halves of Figures 44c and 44f show the last type of error. The simplest way to explain this error is the "jittering" that occurs when the signer's hands are stationary and beside the body in the REST position, usually in the beginning or very end of the sentence. In these cases the MKST is unable to estimate the location of the hand since there is no separation of the arm from the body. In Figures 44c and 44f, the two blobs to the lower right and lower left are the locations of the two hands. Figures 46d and 46f show examples of such a situation for the right and left hand respectively.

The types of errors we encounter with the MKST can be a potential source of error that reduces the sentence recognition accuracy of our method. The reduction in sentence recognition accuracy is reflected in the sentence recognition results shown in Table 27, in the previous chapter, when we compare row 1 (ground truth hand locations + oracle handshape classifier with 100% accuracy) with row 3 (MKST + oracle handshape classifier with 100% accuracy). Hence, we propose a new robust hand detector that avoids these errors and potentially yields better sentence recognition accuracy with the same recognition method.

7.2 Hand detection using domain driven random forest regression

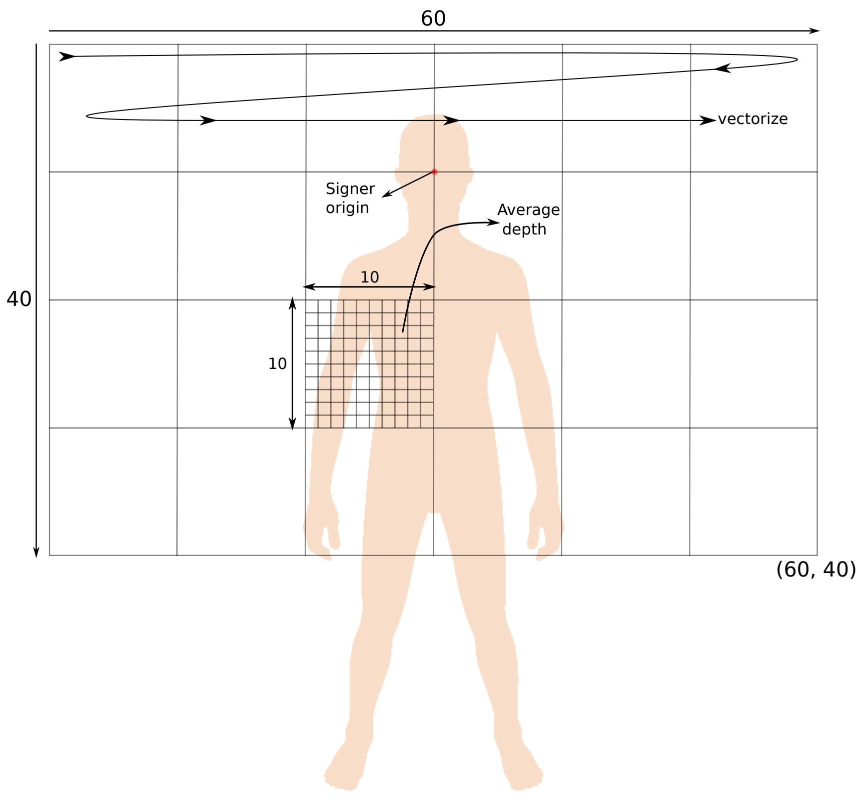
We propose a novel method to predict the location of the right and left hand using domain-driven random forest regression. First, we extract normalized features from the signing space and divide the signing space, based on our domain knowledge of ASL, into regions of varying density with respect to hand locations during signing. A random forest classifier allows us to first index into these divisions wherein we use dedicated random forest regression to make predictions for the hand locations.

7.2.1 Feature Extraction

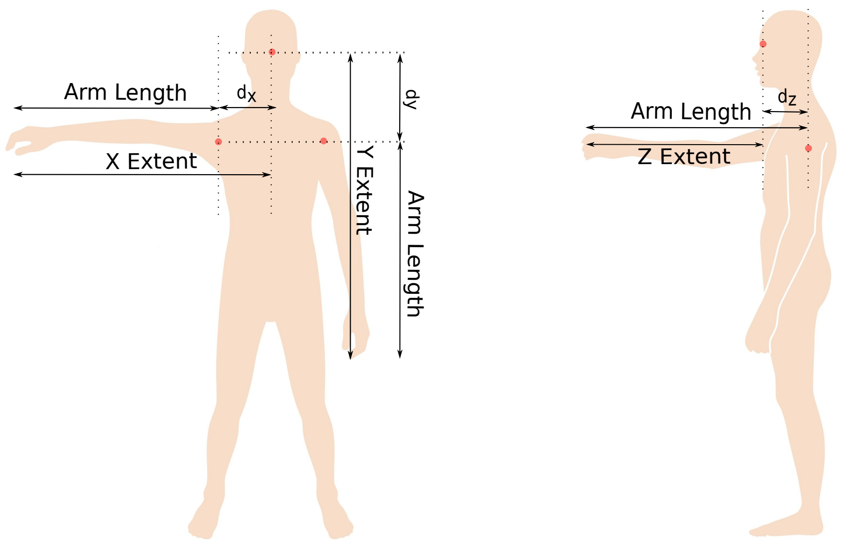
The feature extraction process has three steps:

1. *Convert depth image coordinates, a given x-coordinate, y-coordinate and its corresponding depth value (iX, iY, D), to signer coordinate space.*

   The signer coordinate space refers to the real world space centered at the *signer origin* shown in Figure 47a. To convert image coordinates into signer coordinate space we first convert the image triplet (iX, iY, D) to camera coordinates. Although this information is available at runtime through Microsoft Kinect SDK API's we did not store this data since it would slow down the data capture frame rate. Instead, we used known, correctly tracked hand points, from the skeleton tracking information, to obtain corresponding camera coordinates (cX, cY, D) for the depth image coordinates (iX, iY, D). Then, we estimated the transformation matrix to convert any given image triplet (iX, iY, D) into the camera coordinate space. Finally, the camera coordinates are translated with respect to the *signer origin.* Now, all locations are expressed in terms of meters, in the X, Y and Z directions, from the signer origin. The coordinates in signer space are further normalized to account for the variation in arm lengths of the signers, using maximum possible arm extents in the X, Y and Z directions. Figure 47b shows how the extents are calculated, which are used as divisors for normalization. From now on for feature extraction and random forest training we will use the normalized coordinates. Finally, we eliminate depth image



(a) Feature Extraction



(b) Computation of X, Y and Z extents for normalization

Figure 47: Process for generating features normalized based on arm lengths.

points that do not correspond to the signer's body by restricting the signing space to between -1.0 and +1.0 times the X-extent in the X direction, +0.5 to -1.0 times the Y-extent in the Y direction and -1.0 to + 0.5 times the Z-extent in the Z direction.

2. *Quantize the signing space and compute depth averages.*

   The signing space is first divided into a 60x40 grid in the X-Y plane as shown in Figure 47a. Then we compute the average of the depth values (normalized Z-coordinates) of all points that fall within each grid location.

3. *Vectorize and eliminate zero-columns.*

   Once the average depth value in each grid location is obtained, the values are vectorized in a left-to-right manner, starting from the top-left grid location and going down to the bottom-right grid location. This process yields a vector of length 2400 for each depth image. This initial feature vector is computed for all depth images in the dataset. Then, we construct a Nx2400 matrix, where N is the number of depth images in the dataset, and find columns of the matrix that contain a zero for all the N feature vectors. For example, for the location (60,40) marked in Figure 47a, the average depth value for all images in the dataset will be zero since it is likely that no depth data is ever recorded in that location. These zero-columns are eliminated from the feature vector as they do not give us any additional information that may be useful for predicting the hand locations. After eliminating the zero-columns, the feature vectors for our dataset were of length 1295, reduced from the original 2400.

7.3 Training

Our method to train a predictor to detect the hand locations uses random forest regression. Random forests or randomized decision trees were first proposed by Leo Breiman [16]. The term "forests" is used to highlight that they are a combination of decision trees that

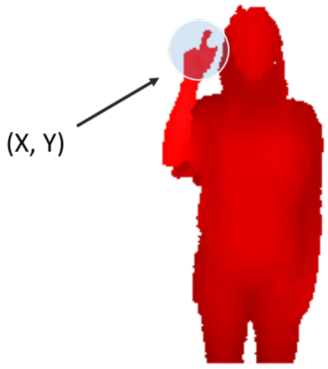


Figure 48: Reverse lookup mechanism to obtain the z coordinate of the hand. The predicted x and y location aid a local search to obtain the z coordinate by averaging values in the depth map.

operate on randomly sub-sampled features. For classification each decision tree outputs a class label and then the most occurring label is chosen as the final output, whereas for regression the output of all the trees is averaged. Random forests are robust and popular with the computer vision community due to their ability to handle highly noisy feature spaces well [28, 50, 79].

For hand detection four separate predictors are trained, using identical feature vectors, to predict the x and y locations of the right and the left hand. The z-coordinate is each case is predicted using a reverse lookup mechanism whereby we use the x and y location to perform a local search in the depth image and then obtain the z value by averaging the depth values. The reverse lookup method is illustrated in Figure 48. Due to time constraints we restricted the number of trees in the random forest to 240, which meant that the average prediction from all the trees was not as accurate compared to if there were thousands of trees. To mitigate this problem and still keep the time required for training to be manageable we first divided the training data into subsets, based on our domain knowledge of ASL, such that the range of the predicted variable for each subset was considerably smaller compared to the full range. This process is illustrated in Figure 49 wherein the ground truth y coordinate of the left hand is plotted against the normalized depth value of the point, in the depth map of the signer, closest to the camera. Similar plots can be created for the ground truth x coordinates of the left hand and the ground truth x and y coordinates of the right hand.

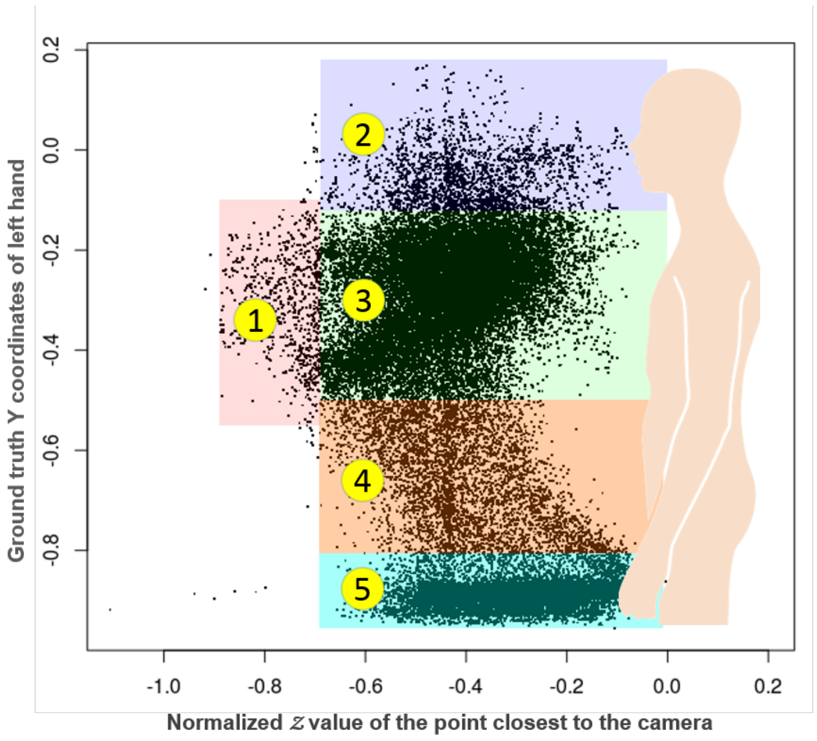


Figure 49: Plot of ground truth y coordinates of the left hand against the normalized depth value of the point in the depth map closest to the camera (lower X-axis values represent points closer to the camera in the -1.0 direction).

We chose the X-axis variable for the plot in order to visualize the relationship between the range of depth in the depth map (the closer a hand is to the camera the larger is the range of the depth map) and the range of the predicted variable. We can see from the plot that larger the depth range (as we get closer to -1.0 on the X-axis) the smaller is the range of the Y-axis variable (in this case the y coordinate of the left hand).

For a moment we request the reader to glance away from Figure 49 and reflect back on the MKST errors. We talked about errors happening away from the body, close to the body when hands touch each other, near the face and jitter errors in the REST position. Ignoring the errors, the first three regions at a high level represent locations in the signing space where signing mostly occurs. Now looking back at Figure 49; the shaded regions 1,2,3 and 5 map directly to the four aforementioned regions. Region 4 captures the movement epenthesis that occurs as the signer moves his hand away from the REST position as he/she gets ready to sign. These five divisions of the data stem directly from our domain knowledge of ASL and knowledge of the method of data collection (sentences starting and ending in the REST position). Notice that there is a high concentration of data points in front of the body above the waist and below the chin, and in the REST position. By reducing the range of the predicted variable in those regions and by virtue of having lots of data points for training we reduce the magnitude of error and thus are able to improve predictions for a large portion of the data.

The question arises as to how we can, at the time of predicting for a test data point, index into the right subset of the data and chose the appropriate random forest to apply. For region 1 it is straightforward; we apply a threshold on the depth range in the depth map and if the range is large >70% of the arm-length then we apply random forest #1. Regions 2-4 fall on the other side of the threshold (<=70%) but there is no straightforward way to index into the correct region. For that situation, we first train a random forest classifier to index into regions 2-4, then apply the corresponding random forest #2 - #4 that has been trained specially for the chosen region. By restricting the range of predictions of the random forest we are shrinking the error introduced by averaging ill-spread predictions that occur when a single random forest is applied to the full range of the data.

7.4 Experimental Results

We test our hand detection performance against that of the Microsoft Kinect Skeleton Tracker (MKST). We employ user-adaptive cross-validation due to the sparsity of signers in our dataset. Given that we only have five signers, running user-independent experiments would not allow us to observe the true power of the approach as we would not have enough variation of body types and signing styles represented in our dataset. For user-adaptive, we add all ground truth data points for hand locations from the first example of each sign or classifier predicate from the left-out fifth signer into the training set and then test on the remaining instances of the fifth signer. This approach demonstrates the results in our tracking performance with a small amount of training effort to adapt the system to a new signer. As discussed earlier in Section 6.3.2 the user-adaptive approach is reasonable in cases where the training data has fewer signers; a small effort in recalibrating the application or game for a new signer will significantly boost performance.

7.4.1 Comparing domain driven random forest regression (DDRFR) with Microsoft Kinect Skeleton Tracker (MKST).

Table 28: Comparison of hand detection %error - DDRFR vs MKST.

	Signer					
Method	1	2	3	4	5	Overall%
MKST	34.12	38.48	45.50	44.72	39.09	40.38%
DDRFR	23.64	34.2	61.8	36.1	26.9	36.53%

Table 28 shows the comparison of hand detection error with MKST and DDRFR. The error results are based on predicted three-dimensional points lying outside a margin of error of 5 cm. The justification for this margin is that 5 cm approximately represents half the width of a hand, and we can expect that if a predicted point lies outside this 5 cm margin it is likely to not contain depth information significantly relevant to the hand. Accurate hand detection enables us to obtain reliable data to do handshape classification. We see that in the user-adaptive case the DDRFR method produces fewer errors on average compared to the MKST. In fact, we observe that most of those errors are due to one signer, Signer 3, which is likely due to significant changes in the signing style of Signer 3 compared to other signers. If we omit Signer 3 from our results, the average error drops to 30.2% which is significantly better than the error rate of the MKST at 40.38%.

Table 29 shows the comparison of sentence recognition accuracy achieved by using the DDRFR method for hand detection versus using the MKST. The sentence recognition results are obtained by using the recognition pipeline shown in Figure 41. For a sentence

Table 29: Comparison of sentence recognition accuracy DDRFR vs MKST.

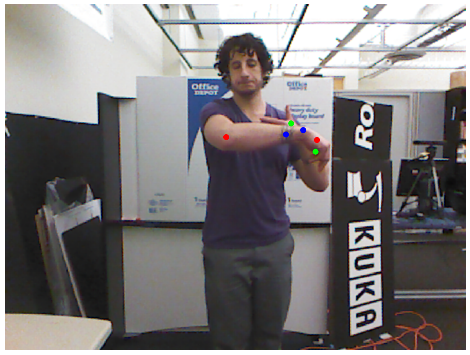
		Signers					
Hand detection	Handshape classification	#1	#2	#3	#4	#5	Overall%
MKST	Oracle	84.9	82.6	78.4	83.7	81.9	82.3%
DDRFR	Oracle	90.8	89.1	74.2	84.9	88.1	85.4%
MKST	Section 6.3.4	80.3	77.5	70.9	77.4	76.2	76.4%
DDRFR	Section 6.3.4	85.4	85.2	67.3	80.1	81.4	79.9%

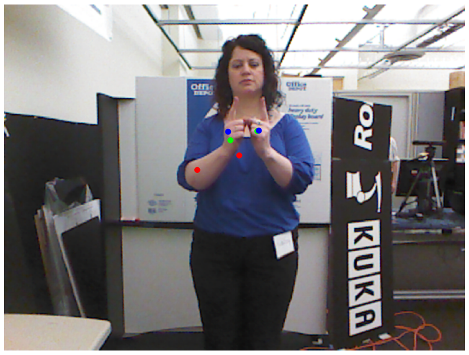
to be recognized as correct, the result at every step of the pipeline should be correct; if even one step produces the wrong result then the sentence is not recognized. We have seen earlier in Table 27, by using an oracle handshape classifier, which always outputs 100% accurate results, we can achieve sentence recognition accuracy of over 90% for all signers and an average accuracy of 93.4% when we use ground truth hand locations. If the handshape classifier described in Section 6.3.4 is used, 88.1% sentence recognition accuracy is achieved. This result gives us an upper bound on the sentence recognition accuracy as if we had a 100% accurate hand detector or tracker, all other methods for hand detection/tracking, in our case MKST and DDRFR, will probably yield lower sentence recognition accuracies.. We have discussed earlier in Section 6.5.1 the complex impact that a better hand detector can have on the results obtained from the recognition pipeline. A better hand detector will directly provide a positive impact on recognition of verb-of-movement classifier predicates. For noun signs a better hand detector can both directly and indirectly (by improving handshape classification) provide a positive impact on recognition. From Table 29 we clearly see that having better hand detection in the case of DDRFR with user-adaptive hand detection yields a better sentence recognition accuracy of 85.42% compared to 82.3% using the MKST when an oracle handshape classifier is used. If we switch to the handshape classifier from Section 6.3.4 sentence recognition accuracies of 79.9% and 76.4% are obtained in the case of DDRFR and MKST respectively.

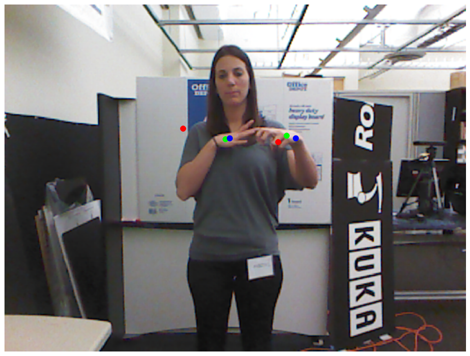
7.4.1.1 Findings

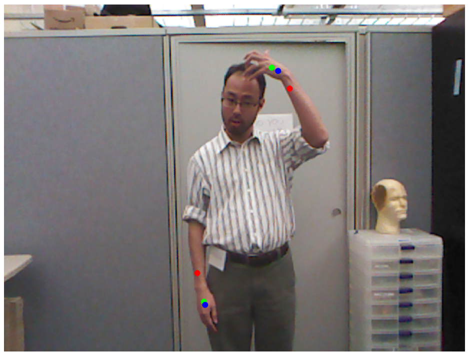
Figure 50 shows the comparison between DDRFR and MKST, for the cases shown earlier in Figure 46. The green, red and blue dots represent the ground truth, MKST and DDRFR locations respectively. We see that in most cases DDRFR does better than the MKST. In two cases we see the DDRFR predictor making errors. In Figure 50a we see that the MKST has a better location for the right hand compared to the DDRFR predictor. In Figure 50e both the left hand and right hand predictions are off by a small amount. Overall it appears that DDRFR has errors of smaller magnitude compared to the MKST.

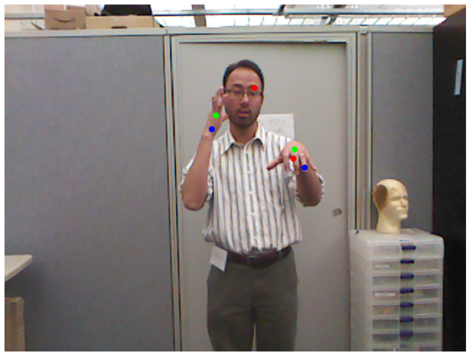
Table 30 shows the percentage occupancy of each signer in feature space. Recall that


(a)


(b)


(c)


(d)


(e)

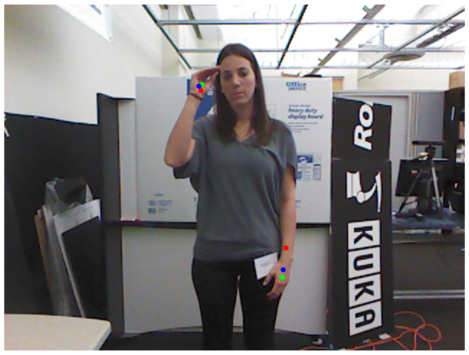

(f)

Figure 50: Comparison of hand locations reported by DDRFR to MKST. (ground truth locations are in green, MKST locations in red and DDRFR hand detection is shown in blue).

Table 30: Occupancy ratio in feature space for each signer.

Signer	Occupancy%
1	32.33%
2	33%
3	52.33
4	40.87%
5	42.95%

our features are constructed from a 60x40 grid in signer space (Figure 47a). We clearly see that for Signer 3 the occcupancy ratio is significantly different than other signers. Also we see that for every other signer there is one other signer who has a similar occupancy ratio but we find none such case for Signer 3. This could explain the reason why the DDRFR predictor does not do well for Signer 3, since there is not enough training data to make accurate predictions.

7.5 Discussion

The domain driven random forest regression detected hands better than the Microsoft Kinect Skeleton Tracker. However, there was one signer (Signer 3) for whom the method did worse than the MKST. We believe that the poor result for Signer 3 is mainly due to lack of training data that matches the Signer 3's signing style, which has been shown in Table 30 to be significantly different from the others. Although we used user-adaptive training, the predictions were inaccurate, highlighting that there was insufficient training data.

The motivation to do user-adaptive training is the ability to quickly collect a small amount of sample data from an unknown signer and use that data to retrain the prediction trees. The retrained trees will likely have better predictions than without user-adaptive training. In real-world applications this small amount of time sacrificed to collect new samples and retrain the system pays heavy dividends by improving robustness. However, it appears, based on our experiments, that when the signer has a significantly different style compared to other signers in the training data, a small amount of additional signing data to perform user-adaptive training will not result in improved performance. The only solution that remains is to train with larger number of signers accounting for several different body types and signing styles. A larger pool of signers in the training data mitigates the need for user-adaptive training but requires more effort to label the data. It is our belief that the DDRFR method for hand detection will consistently perform better than the MKST when trained with a larger set of signers, with different body types and signing styles.

Looking ahead from the perspective of building an ASL-classifier based game with a fully automatic recognition system, the results shown in Table 29 are promising. A switch to verification, as was done in the CopyCat game will provide the necessary boost in performance making it possible for the game to be employed in educational settings. We are cautiously encouraged from the fact that the DDRFR method for hand detection in combination with the handshape classification method from Section 6.3.4 has close to 80% sentence recognition accuracy as compared to 67% sentence recognition in the case of the earlier CopyCat system when tested on the Gwinnett dataset (see Table 5). However, one major difference is that signers in the Gwinnett dataset were children whereas in the Classifier-Kinect dataset the signers are adults. Still, even though recognition accuracy may drop with newer datasets with younger signers, due to higher signing variation and presence of disfluencies, sentence verification in an educational ASL-classifier game as opposed to recognition will improve the system's overall accuracy. As we have seen in the past, a live deployment of the CopyCat sentence verifier, that had 82% accuracy on the Gwinnett dataset (see Table 5) was sufficient to produce an educational effect with deaf children, on average almost doubling their test scores after playing the game for two weeks (see Section 3.3).

CHAPTER VIII

CONCLUSION AND FUTURE WORK

Educational sign language games can play an important role towards acquiring essential language skills especially for deaf children born to hearing parents, who constitute 95% of all deaf children born in the United States. Advancements in automatic sign language recognition (ASLR) have made it possible to build computer systems that automatically recognize finger spellings, isolated signs, facial expressions and interrogative words like WH-questions (e.g. who, what, where, and when). However, there is much to do in ASLR and we are far from developing an advanced sign language recognition system on par with current voice recognition systems that can recognize spoken English of hundreds of thousands of speakers. Some areas that are currently underexplored in ASLR are sentence verification and recognition of classifier-based grammatical structures of American Sign Language (ASL).

In our previous work we developed CopyCat, an educational ASL game that requires children to engage in a progressively more difficult expressive signing task as they advance through the game. This game helps children improve their short term memory and increases their "wordspan," the ability to recall and repeat signs/words immediately after they have seen a tutorial video of the ASL sentence. CopyCat was the first game of its kind in providing deaf children the ability to practice their sign language skills in the absence of a conversational partner. We have demonstrated in Chapter 3 that by leveraging context we can use verification, in place of recognition, to boost machine performance for determining if the signed responses in the expressive signing tasks of the CopyCat game, are correct or incorrect. Although CopyCat included only English-like sentences it provided us a template to design future games and build enhancements that would make it possible to recognize other linguistic aspects of ASL.

A critical component of any educational ASL game that employs automatic recognition/verification is the ability to provide feedback to the signer about errors in signing. To successfully provide feedback the automated system should have the ability to accurately segment parts of the ASL sentence including the signs for future reference. In Chapter 4 we have demonstrated that the quality of a machine verifier's ability to identify the boundary of the signs can be improved by using a two-pass technique that combines signed input in both forward and reverse directions.

So far our focus had been on advancing ASLR research to support sentence recognition without paying attention to economics of scale, which is paramount for an educational game like CopyCat to attain large scale adoption and fulfill the needs of our target population. We estimated that the custom hardware required for the CopyCat game, which includes one desktop/laptop computer, a pair of colored gloves with attached accelerometers, a IEEE1394 camera and a kiosk setup would cost close to \$2000. The cost of the hardware setup definitely presented a barrier for adoption. Advances in imaging technology have resulted in the availability of inexpensive off-the-shelf depth cameras. The widely available Microsoft Kinect Camera costs a mere \$250 and has the flexibility of being able to connect either to a desktop/laptop computer or to the Microsoft Xbox gaming system. In Chapter 5 we have shown that we can reduce CopyCat's dependency on custom manufactured hardware by using an off-the-shelf Microsoft Kinect depth camera to achieve similar verification performance compared to the earlier hardware.

As we mentioned earlier, recognition of classifier-based grammatical structures in ASL is still relatively underexplored. The combination of RGB and depth information available through the Microsoft Kinect camera is particularly suitable for recognizing the visual-spatial constructions of ASL classifiers. In Chapter 6 we give details about an ASL classifier dataset (Classifier-Kinect) we designed and collected in consultation with ASL linguists and fluent signers. We have presented recognition results on this dataset by using a method that combines hand tracking information provided by the Microsoft Kinect camera and a technique for handshape classification. In Chapter 7 we show that the generic human pose estimator provided the Microsoft Kinect is not suitable for hand tracking in ASL datasets and propose a method to detect hand positions using domain-driven random forest regression. We show that the new hand detector provides reliable hand detection, which results in better recognition accuracy.

In the future we hope to collect classifier-based ASL datasets, with larger sign vocabulary and additional ASL classifiers, using young deaf signers as opposed to fluent adult signers in the Classifier-Kinect dataset. One issue with the Classifier-Kinect dataset was the scarcity of signers, which meant that variations in signing and different body types of the signers, needed to train a robust hand tracker and for training better hidden Markov models to model the noun signs and verb-of-movement classifier predicates, were not adequately captured. For collection of future classifier-based ASL datasets we will recruit a significantly larger number of young deaf singers, which will introduce variation in signing and body type.

From the point of view of the recognition pipeline, we will retrain the hand detector, train a new hand shape classifier to handle additional classifier handshapes and build new hidden Markov models to recognize signs and classifier predicates in the expanded vocabulary. Our hope is to bundle the new recognizer in to a CopyCat-like ASL classifier game application and release the game through channels, such as Microsoft Xbox Live, that facilitate content delivery directly to people's homes, as well as direct download for installation on personal computers.

REFERENCES

[1] Aran, O., Ari, I., Akarun, L., Sankur, B., Benoit, A., Caplier, A., Campr, P., Carrillo, A., and Fanarda, F., "SignTutor: An Interactive system for sign language tutoring," *IEEE MultMedMag*, pp. 81–93, January 2009.

[2] Aran, O., Ari, I., Akarun, L., Sankur, B., Benoit, A., Caplier, A., Campr, P., Carrillo, A. H., and Fanard, F.-X., "Signtutor: An interactive system for sign," 2009.

[3] Athitsos, V., Alon, J., Sclaroff, S., and Kollios, G., "Boostmap: A method for efficient approximate similarity rankings," in *In proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004 (CVPR 2004)*, vol. 2, pp. II–268–II–275 Vol.2, June 2004.

[4] Athitsos, V. and Sclaroff, S., "3d hand pose estimation by finding appearance-based matches in a large database of training views," in *IEEE Workshop on Cues in Communication*, 2001.

[5] Athitsos, V. and Sclaroff, S., "Estimating 3D hand pose from a cluttered image," *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 2, pp. 432–432–9 vol.2, 2003.

[6] Bauer, B., Hienz, H., and Kraiss, K., "Video-based continuous sign language recognition using statistical methods," in *ICPR 2000*, pp. 463–466, Vol II.

[7] Bauer, B., Hienz, H., and Kraiss, K., "Video-based continuous sign language recognition using statistical methods," pp. Vol II: 463–466, 2000.

[8] Bauer, B., Hienz, H., and Kraiss, K., "Video-based continuous sign language recognition using statistical methods," in *Proceedings of the 15th International Conference on Pattern Recognition*, vol. 2, pp. 463–466, September 2000.

[9] Bellugi, U., OGrady, L., Lillo-Martin, D., OGrady Hynes, M., Hoek, K., and Corina, D., "Enhancement of spatial cognition in deaf children," in *From Gesture to Language in Hearing and Deaf Children* (Volterra, V. and Erting, C., eds.), vol. 27 of *Springer Series in Language and Communication*, pp. 278–298, Springer Berlin Heidelberg, 1990.

[10] Black, M. J. and Jepson, A. D., "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation," *International Journal of Computer Vision*, vol. 26, no. 1, pp. 63–84, 1998.

[11] Brashear, H., *Improving the Efficacy of Automated Sign Language Practice Tools.* PhD thesis, Georgia Institute of Technology, College of Computing, 2010.

[12] Brashear, H., Henderson, V., Park, K.-H., Hamilton, H., Lee, S., and Starner, T., "American sign language recognition in game development for deaf

children," in *Proceedings of the 8th international ACM SIGACCESS conference on Computers and accessibility*, pp. 79–86, ACM, 2006.

[13] Brashear, H., Park, K.-H., Lee, S., Henderson, V., Hamilton, H., and Starner, T., "American Sign Language recognition in game development for deaf children," in *ASSETS 06*, (New York, NY, USA), pp. 79–86, 2006.

[14] Brashear, H., Starner, T., Lukowicz, P., and Junker, H., "Using multiple sensors for mobile sign language recognition," in *ISWC '03*, (Washington, DC, USA), pp. 45–52, IEEE Computer Society, October 2003.

[15] Brashear, H., Zafrulla, Z., Presti, P., Starner, T., Hamilton, H., and Lee, S., "A corpus for verifying American Sign Language during game play by deaf children," in *LREC 2010*, (Malta), 2010.

[16] Breiman, L., "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[17] Cameron, C. R., DiValentin, L. W., Manaktala, R., McElhaney, A. C., Nostrand, C. H., Quinlan, O. J., Sharpe, L. N., Slagle, A. C., Wood, C. D., Zheng, Y. Y., and others, "Hand tracking and visualization in a virtual reality simulation," in *Systems and Information Engineering Design Symposium (SIEDS), 2011 IEEE*, pp. 127–132, IEEE, 2011.

[18] Chai, X., Li, G., Lin, Y., Xu, Z., Tang, Y., Chen, X., and Zhou, M., "Sign language recognition and translation with kinect," in *IEEE Conf. on AFGR*, 2013.

[19] Chen, C.-P., Chen, Y.-T., Lee, P.-H., Tsai, Y.-P., and Lei, S., "Real-time hand tracking on depth images," in *Visual Communications and Image Processing (VCIP), 2011 IEEE*, pp. 1–4, IEEE, 2011.

[20] Cooper, H. and Bowden, R., "Sign language recognition: Working with limited corpora," in *Universal Access in Human-Computer Interaction. Applications and Services 5th International Conference, UAHCI 2009, held as part of HCI International 2009*, pp. 472–481, 2009.

[21] de Campos, T. and Murray, D., "Regression-based hand pose estimation from multiple cameras," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 1, pp. 782–789, June 2006.

[22] de La Gorce, M., Fleet, D., and Paragios, N., "Model-based 3D hand pose estimation from monocular video," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, pp. 1793–1805, Sept 2011.

[23] Doliotis, P., Athitsos, V., Kosmopoulos, D., and Perantonis, S., "Hand shape and 3D pose estimation using depth data from a single cluttered frame," in *Advances in Visual Computing* (Bebis, G., Boyle, R., Parvin, B., Koracin, D., Fowlkes, C., Wang, S., Choi, M.-H., Mantler, S., Schulze, J., Acevedo, D., Mueller, K., and Papka, M., eds.), vol. 7431 of *Lecture Notes in Computer Science*, pp. 148–158, Springer Berlin Heidelberg, 2012.

[24] Droeschel, D., Stückler, J., and Behnke, S., "Learning to interpret pointing gestures with a time-of-flight camera," in *Proceedings of the 6th international conference on Human-robot interaction*, pp. 481–488, ACM, 2011.

[25] Emmorey, K., Kosslyn, S. M., and Bellugi, U., "Visual imagery and visual-spatial language: Enhanced imagery abilities in deaf and hearing asl signers," *Cognition*, vol. 46, no. 2, pp. 139–181, 1993.

[26] Eng-Jon, O. and Bowden, R., "A boosted classifier tree for hand shape detection," in *Proceedings of the Sixth IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 889–894, May 2004.

[27] Erenshteyn, R. and Laskov, P., "A multi-stage approach to fingerspelling and gesture recognition," 1996.

[28] Fanelli, G., Gall, J., and Van Gool, L., "Real time head pose estimation with random regression forests," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 617–624, IEEE, 2011.

[29] Fang, G., Gao, W., and Zhao, D., "Large vocabulary sign language recognition based on hierarchical decision trees," in *International Conference on Multimodal Interfaces*, pp. 125–131, 2003.

[30] Fang, G., Gao, W., and Zhao, D., "Large-vocabulary continuous sign language recognition based on Transition-Movement Models," *SMC-A*, vol. 37, pp. 1–9, January 2007.

[31] Frati, V. and Prattichizzo, D., "Using kinect for hand tracking and rendering in wearable haptics," in *World Haptics Conference (WHC), 2011 IEEE*, pp. 317–321, IEEE, 2011.

[32] Gao, W., Fang, G., Zhao, D., and Chen, Y., "Transition movement models for large vocabulary continuous sign language recognition (CSL)," in *Sixth IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 553–558, 2004.

[33] Grobel, K. and Assan, M., "Isolated sign language recognition using hidden Markov models," in *IEEE International Conference on Systems, Man, and Cybernetics: Computational Cybernetics and Simulation*, vol. 1, pp. 12–15, October 1997.

[34] Habili, N., Lim, C. C., and Moini, A., "Segmentation of the face and hands in sign language video sequences using color and motion cues," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 14, no. 8, pp. 1086–1097, 2004.

[35] Haiping, L., Konstantinos N, P., Anastasios N, V., and others, "A full-body layered deformable model for automatic model-based gait recognition," *EURASIP Journal on Advances in Signal Processing*, vol. 2008, 2007.

[36] Henderson, V., Lee, S., Brashear, H., Hamilton, H., Starner, T., and Hamilton, S., "Development of an American Sign Language Game for Deaf Children," in *IDC '05: Proceeding of the 2005 Conference on Interaction Design and Children*, (New York, NY, USA), ACM Press, June 2005.

[37] Hernandez-Rebollar, J. L., Kyriakopoulos, N., and Lindeman, R. W., "A new instrumented approach for translating American Sign Language into sound and text," in *Proceedings of the Sixth IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 547–552, 2004.

[38] Huddleston, W. E., Lewis, J. W., Phinney, R. E., and Deyoe, E. A., "Auditory and visual attention-based apparent motion share functional parallels," *Perception & Psychophysics*, vol. 70, pp. 1207–1216, Oct. 2008.

[39] Huenerfauth, M., *Generating American Sign Language Classifier Predicates For English-To-ASL Machine Translation.* PhD dissertation, University of Pennsylvania, 2006.

[40] Huenerfauth, M., "Spatial, temporal, and semantic models for american sign language generation: Implications for gesture generation.," *Int. J. Semantic Computing*, vol. 2, no. 1, pp. 21–45, 2008.

[41] Imagawa, K., Lu, S., and Igi, S., "Color-based hands tracking system for sign language recognition," in *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, pp. 462–467, IEEE, 1998.

[42] Jiang, H., "Confidence measures for speech recognition: A survey," *Speech Communication*, vol. 45, no. 4, pp. 455 – 470, 2005.

[43] Kadous, M. W. and Engineering, C. S., "Machine recognition of auslan signs using powergloves: Towards large-lexicon recognition of sign language," in *Proceedings of the Workshop on the Integration of Gesture in Language and Speech*, pp. 165–174, 1996.

[44] Keskin, C., Kirac, F., Kara, Y., and Akarun, L., "Real time hand pose estimation using depth sensors," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pp. 1228–1234, Nov 2011.

[45] Keskin, C., Kiraç, F., Kara, Y. E., and Akarun, L., "Real time hand pose estimation using depth sensors," in *Consumer Depth Cameras for Computer Vision*, pp. 119–137, Springer, 2013.

[46] Khoshelham, K. and Elberink, S. O., "Accuracy and resolution of kinect depth data for indoor mapping applications," *Sensors*, vol. 12, no. 2, pp. 1437–1454, 2012.

[47] Kim, J. S., Jang, W., and Bien, Z., "A dynamic gesture recognition system for the Korean Sign Language KSL," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 26, no. 2, pp. 354–359, 1996.

[48] Lamberti, L. and Camastra, F., "Real-time hand gesture recognition using a color glove," in *Image Analysis and Processing–ICIAP 2011*, pp. 365–373, Springer, 2011.

[49] Lee, S., Henderson, V., Hamilton, H., Starner, T., Brashear, H., and Hamilton, S., "A gesture-based american sign language game for deaf children," in *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '05, (New York, NY, USA), pp. 1589–1592, ACM, 2005.

[50] Lempitsky, V., Verhoek, M., Noble, J. A., and Blake, A., "Random forest classification for automatic delineation of myocardium in real-time 3d echocardiography," in *Functional Imaging and Modeling of the Heart*, pp. 447–456, Springer, 2009.

[51] Li, Z. and Jarvis, R., "Real time hand gesture recognition using a range camera," in *Australasian Conference on Robotics and Automation*, pp. 21–27, 2009.

[52] Liang, R. and Ouhyoung, M., "A real-time continuous gesture recognition system for sign language," in *Third International Conference on Automatic Face and Gesture Recognition*, pp. 558–565, 1998.

[53] Liwicki, S. and Everingham, M., "Automatic recognition of fingerspelled words in British Sign Language," in *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pp. 50–57, 2009.

[54] Martin, Joy, A., Sera, and Maria, D., "The acquisition of spatial constructions in American Sign Language and English," *Journal of Deaf Studies and Deaf Education*, vol. 11, pp. 391–402, 2006.

[55] Mathan, L. and Miclet, L., "Rejection of extraneous input in speech recognition applications, using multi-layer perceptrons and the trace of hmms," in *Acoustics, Speech, and Signal Processing, 1991. ICASSP-91., 1991 International Conference on*, pp. 93 –96 vol.1, apr 1991.

[56] Matsuo, H., Igi, S., Lu, S., Nagashima, Y., Takata, Y., and Teshima, T., "The recognition algorithm with non-contact for Japanese Sign Language using morphological analysis," *Gesture and Sign Language in Human-Computer Interaction: Proceedings of the International Gesture Workshop*, vol. 1371, 1997.

[57] Mayberry, R. I. and Eichen, E. B., "The long-lasting advantage of learning sign language in childhood: Another look at the critical period for language acquisition," *Journal of Memory and Language*, vol. 30, pp. 486–498, 1991.

[58] Metaxas, D., Liu, B., Yang, F., Yang, P., Michael, N., and Neidle, C., "Recognition of nonmanual markers in american sign language (asl) using non-parametric adaptive 2D-3D face tracking," in *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*, pp. 2414–2420, May 2012.

[59] Michael, N., Metaxas, D., and Neidle, C., "Spatial and temporal pyramids for grammatical expression recognition of american sign language," in *Proceedings of the 11th international ACM SIGACCESS conference on Computers and accessibility*, Assets '09, (New York, NY, USA), pp. 75–82, ACM, 2009.

[60] Minnen, D. and Zafrulla, Z., "Towards robust cross-user hand tracking and shape recognition," in *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 1235–1241, 2011.

[61] Mitchell, R. and Karchmer, M., "Chasing the Mythical Ten Percent: Parental Hearing Status of Deaf and Hard of Hearing Students in the United States," *Sign Language Studies*, vol. 4, pp. 138–163, Winter 2004.

[62] Mittal, A., Zisserman, A., and Torr, P. H., "Hand detection using multiple proposals.," in *In Proceedings of the British Machine Vision Conference, (BMVC 2011)*, pp. 1–11.

[63] Mo, Z. and Neumann, U., "Real-time hand pose recognition using low-resolution depth images," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2 (CVPR 2006)*, pp. 1499–1505.

[64] Moeslund, T. B., Hilton, A., and Krüger, V., "A survey of advances in vision-based human motion capture and analysis," *Computer vision and image understanding*, vol. 104, no. 2, pp. 90–126, 2006.

[65] Nayak, S., Sarkar, S., and Loeding, B., "Automated extraction of signs from continuous sign language sentences using iterated conditional modes," in *IEEE Conference on Computer Vision and Pattern Recognition, (CVPR 2009)*, pp. 2583–2590.

[66] Neidle, C., Michael, N., Nash, J., and Metaxas, D., "A Method for Recognition of Grammatically Significant Head Movements and Facial Expressions, Developed through use of a Linguistically Annotated Video Corpus," in *Proceedings of the Workshop on Formal Approaches to Sign Languages, held as part of the 21st European Summer School in Logic, Language and Information*, (Bordeaux, France), July 2009.

[67] NIH, "American Sign Language," 2011. Publication No. 11-4756, http://www.nidcd.nih.gov/health/hearing/pages/asl.aspx (Accessed September 5 2012).

[68] Ong, S. and Ranganath, S., "Automatic sign language analysis: a survey and the future beyond lexical meaning," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, pp. 873 – 891, jun 2005.

[69] Oz, C. and Leu, M. C., "American Sign Language word recognition with a sensory glove using artificial neural networks," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 7, pp. 1204–1213, 2011.

[70] Parasnis, I. and Samar, V. J., "Parafoveal attention in congenitally deaf and hearing young adults," *Brain and Cognition*, vol. 4, no. 3, pp. 313 – 327, 1985.

[71] Parasnis, I., Samar, V. J., Bettger, J. G., and Sathe, K., "Does deafness lead to enhancement of visual cognition in children? Negative evidence from deaf nonsigners.," *Journal of Deaf Studies and Deaf Education*, vol. 1(2), pp. 142–152, 1996.

[72] Park, S., Yu, S., Kim, J., Kim, S., and Lee, S., "3D hand tracking using kalman filter in depth space," *EURASIP Journal on Advances in Signal Processing*, vol. 2012, no. 1, pp. 1–18, 2012.

[73] Prime Sense, *The PrimeSensor^{TM}Reference Design 1.08.* April 2011. http://www.PrimeSense.com.

[74] Pugeault, N. and Bowden, R., "Spelling it out: Real-time asl fingerspelling recognition," in *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 1114–1119, 2011.

[75] Rabiner, L., "A tutorial on hidden markov models and selected applications in speech recognition," vol. 77, pp. 257–286, Feb 1989.

[76] Rehg, J. and Kanade, T., "Visual tracking of high dof articulated structures: An application to human hand tracking," in *Computer Vision ECCV '94* (Eklundh, J.-O., ed.), vol. 801 of *Lecture Notes in Computer Science*, pp. 35–46, Springer Berlin Heidelberg, 1994.

[77] Ren, Z., Yuan, J., and Zhang, Z., "Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera," in *Proceedings of the 19th ACM international conference on Multimedia*, pp. 1093–1096, ACM, 2011.

[78] Rose, R., Juang, B., and Lee, C., "A training procedure for verifying string hypotheses in continuous speech recognition," in *International Conference on Acoustics, Speech, and Signal Processing, (ICASSP-1995)*, vol. 1, pp. 281 –284.

[79] Saffari, A., Leistner, C., Santner, J., Godec, M., and Bischof, H., "Online random forests," in *12th IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 1393–1400, IEEE, 2009.

[80] Sagawa, H. and Takeuchi, M., "A method for recognizing a sequence of sign language words represented in a Japanese Sign Language sentence," in *Proceedings of the 4th IEEE International Conference on Automatic Face and Gesture Recognition*, (Grenoble, France), pp. 434–439, March 2000.

[81] Shotton, J., Sharp, T., Kipman, A., Fitzgibbon, A., Finocchio, M., Blake, A., Cook, M., and Moore, R., "Real-time human pose recognition in parts from single depth images," *Communications of the ACM*, vol. 56, no. 1, pp. 116–124, 2013.

[82] Sonka, M., Hlavac, V., and Boyle, R., *Image Processing, Analysis, and Machine Vision.* Toronto, Ontario, Canada: Thomspon Learning, third ed., 2008.

[83] Starner, T. and Pentland, A., "Visual recognition of American Sign Language using hidden Markov models," in *Proceedings of the International Workshop on Automatic Face and Gesture Recognition*, 1995.

[84] Starner, T., Weaver, J., and Pentland, A., "Real-time American Sign Language recognition using desk and wearable computer based video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 12, pp. 1371–1375, 1998.

[85] Starner, T. and Pentland, A., "Real-time american sign language recognition from video using hidden markov models," in *Motion-Based Recognition*, pp. 227–243, Springer, 1997.

[86] Stenger, B., Mendonca, P., and Cipolla, R., "Model-based 3D tracking of an articulated hand," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR 2001).*, vol. 2, pp. II–310–II–315.

[87] Stokoe, W., "Sign language structure: An outline of the visual communication systems of the American Deaf," *Studies in Linguistics, Occasional Papers 8*, 1960.

[88] Stuckler, J. and Behnke, S., "Combining depth and color cues for scale- and viewpoint-invariant object segmentation and recognition using random forests," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4566 –4571, October 2010.

[89] Suarez, J. and Murphy, R. R., "Hand gesture recognition with depth images: A review," in *The 23rd IEEE International Symposium on Robot and Human Interactive Communication.*, pp. 411–417, 2012.

[90] Sukkar, R. and Lee, C.-H., "Vocabulary independent discriminative utterance verification for nonkeyword rejection in subword based speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 4, pp. 420 –429, nov 1996.

[91] Sukkar, R. and Wilpon, J., "A two pass classifier for utterance rejection in keyword spotting," in *IEEE International Conference on Acoustics, Speech, and Signal Processing, (ICASSP-93)*, vol. 2, pp. 451 –454 vol.2, apr 1993.

[92] Talbot, K. F. and Haude, R. H., "The relation between sign language skill and spatial visualization ability: Mental rotation of three-dimensional objects," *Perceptual and Motor Skills*, vol. 77, pp. 1387–1391, Dec 1993.

[93] Ten Holt, G., Hendriks, P., and Andringa, T., "Why don't you see what I mean? Prospects and limitations of current automatic sign recognition research," *Sign Language Studies*, vol. 6, Summer 2006.

[94] Valli, C., Lucas, C., and Mulrooney, K., *Linguistics of American Sign Language: An Introduction.* Washington D.C.: Gallaudet University Press, 4th ed., 2005.

[95] Vogler, C. and Metaxas, D., "ASL recognition based on a coupling between HMMs and 3D motion analysis," in *ICCV98*, pp. 363–369, 1998.

[96] Vogler, C. and Metaxas, D., "Parallel hidden Markov models for American sign language recognition," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 1, pp. 116–122, 1999.

[97] Vogler, C., Sun, H., and Metaxas, D., "A framework for motion recognition with applications to American Sign Language and gait recognition," in *Proceedings of Workshop on Human Motion*, pp. 33–38, 2000.

[98] Vogler, C. and Metaxas, D., "Adapting hidden markov models for asl recognition by using three-dimensional computer vision methods.," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, (Orlando, FL), pp. 156–161, October 1997.

[99] Vogler, C. and Metaxas, D., "Handshapes and Movements: Multiple-channel American Sign Language recognition," in *Gesture-Based Communication in Human-Computer Interaction*, vol. 2915, pp. 247–258, Springer-Verlag, January 2004. Lecture notes in Artificial Intelligence.

[100] Wang, R. Y. and Popović, J., "Real-time hand-tracking with a color glove," in *ACM Transactions on Graphics (TOG)*, vol. 28, p. 63, ACM, 2009.

[101] Weaver, K. A., Hamilton, H., Zafrulla, Z., Brashear, H., Starner, T., Presti, P., , and Bruckman, A., "Improving the language ability of Deaf signing children through an interactive American Sign Language-based video game," in *Proceedings of 9th International Conference of the Learning Sciences*, June 2010.

[102] Westeyn, T., Brashear, H., Atrash, A., and Starner, T., "Georgia Tech Gesture Toolkit: Supporting experiments in gesture recognition," in *ICMI '03*, (New York, NY, USA), pp. 85–92, ACM Press, 2003.

[103] Xu, C. and Cheng, L., "Efficient hand pose estimation from a single depth image," in *IEEE International Conference on Computer Vision (ICCV)*, pp. 3456–3462, Dec 2013.

[104] Yang, C., Jang, Y., Beh, J., Han, D., and Ko, H., "Gesture recognition using depth-based hand tracking for contactless controller application," in *IEEE International Conference on Consumer Electronics (ICCE)*, pp. 297–298, IEEE, 2012.

[105] Yang, H., Sclaroff, S., and Lee, S., "Sign language spotting with a threshold model based on conditional random Fields," *PAMI*, vol. 31, pp. 1264–1277, July 2009.

[106] Yang, M.-H. and Ahuja, N., "Recognizing hand gestures using motion trajectories," in *Face Detection and Gesture Recognition for Human-Computer Interaction*, pp. 53–81, Springer, 2001.

[107] Yang, R., Sarkar, S., and Loeding, B., "Enhanced level building algorithm for the movement epenthesis problem in sign language recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2007.

[108] Yang, R., Sarkar, S., and Loeding, B., "Handling movement epenthesis and hand segmentation ambiguities in continuous sign language recognition using nested dynamic programming," *The IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 462–477, March 2010.

[109] Yoo, B., Han, J.-J., Choi, C., Yi, K., Suh, S., Park, D., and Kim, C., "3D user interface combining gaze and hand gestures for large-scale display," in *CHI 2010, Extended Abstracts on Human Factors in Computing Systems*, pp. 3709–3714, ACM, 2010.

[110] Young, S. J., Evermann, G., Gales, M. J. F., Hain, T., Kershaw, D., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V., and Woodland, P. C., *The HTK Book, version 3.4.* Cambridge, UK: Cambridge University Engineering Department, 2006.

[111] Young, S., "Detecting misrecognitions and out-of-vocabulary words," *IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, pp. 21–24, 1994.

[112] Yu, D., Ju, Y. C., and Acero, A., "An effective and efficient utterance verification technology using word n-gram filler models," in *Proceedings of Interspeech 2006—ICSLP: 9th International Conference on Spoken Language Processing, Pittsburgh, PA, USA*, 2006.

[113] Zafrulla, Z., Brashear, H., Hamilton, H., and Starner, T., "A novel approach to American Sign Language (ASL) phrase verification using reversed signing," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW 2010)*, pp. 48–55.

[114] Zafrulla, Z., Brashear, H., Hamilton, H., and Starner, T., "Towards an American Sign Langauge verifier for educational game for deaf children," in *Proceedings of International Conference on Pattern Recognition, (ICPR 2010).*

[115] Zafrulla, Z., Brashear, H., Starner, T., Hamilton, H., and Presti, P., "American Sign Language recognition with the kinect," in *Proceedings of the 13th international conference on multimodal interfaces, (ICMI 2011)*, pp. 279–286.

[116] Zhang, Z., "Microsoft kinect sensor and its effect," *IEEE MultiMedia*, vol. 19, pp. 4–10, Apr. 2012.