Sign-to-911: Emergency Call Service for Sign Language Users with Assistive AR Glasses

Yunqi Guo*, Jinghao Zhao*, Boyan Ding*, Congkai Tan*, Weichong Ling*, Zhaowei Tan†, Jennifer Miyaki‡, Hongzhe Du*, Songwu Lu*

* UCLA Computer Science Department † UC Riverside ‡ UCLA Linguistics Department

ABSTRACT

Sign-to-911 offers a compact mobile system solution to fast and runtime American Sign Language (ASL) and English translations. It is designated as 911 call services for ASL users with hearing disabilities upon emergencies. It enables bidirectional translations of ASL-to-English and English-to-ASL. The signer wears the AR glasses, runs Sign-to-911 on his/her smartphone and glasses, and interacts with a 911 operator. The design of Sign-to-911 departs from the popular deep learning based solution paradigm, and adopts simpler traditional AI/machine learning (ML) models. The key is to exploit ASL linguistic features to simplify the model structures and improve accuracy and speed. It further leverages recent component solutions from graphics, vision, natural language processing, and AI/ML. Our evaluation with six ASL signers and 911 call records has confirmed its viability.

CCS CONCEPTS

• **Human-centered computing** → **Accessibility systems and tools**.

KEYWORDS

Mobile AR System, AR Glasses, American Sign Language, Mobile AI, Sign Language Translation, 911, Emergency Call

ACM Reference Format:

Yunqi Guo, Jinghao Zhao, Boyan Ding, Congkai Tan, Weichong Ling, Zhaowei Tan, Jennifer Miyaki, Hongzhe Du, Songwu Lu. 2023. Sign-to-911: Emergency Call Service for Sign Language Users with Assistive AR Glasses. In *The 29th Annual International Conference on Mobile Computing and Networking (ACM MobiCom '23), October 2–6, 2023, Madrid, Spain.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3570361.3613260

ACM MobiCom '23, October 2–6, 2023, Madrid, Spain

ACM ISBN 978-1-4503-9990-6/23/10.

https://doi.org/10.1145/3570361.3613260

1 INTRODUCTION

American Sign Language (ASL) is the primary communication language used by people with hearing disabilities in the United States and part of Canada. It has estimated 0.5 ~ 2 million users [43]. Unlike spoken and written languages, ASL is a visual language that relies on hand features and motions to express words and sentences. This poses both challenges and opportunities for translating ASL and other languages, say, under 911 emergency call situations. Our survey on 54 ASL users confirms that, neither of the existing two options for 911 services (i.e., typing messages to a 911 operator or interpreting via a video relay service) is deemed convenient and readily accessible.

In this work, we thus address a simple, yet important problem for the ASL community: *Can we build a compact mobile system solution for translation between ASL and English, running on wearable device and smartphones, without any cloud/edge support?* If addressed, we enable live, direct emergency call service between an ASL user and a 911 operator, both using their primary languages. Of course, the solution must address the limitations of large training data sets, heavy computations, and considerable energy overhead. Unfortunately, this will rule out most deep learning based proposals [33, 36], since they cannot run on mobile devices without incurring excessive processing and energy overhead.

We depart from the deep learning based solution paradigm, and design a lightweight system Sign-to-911 for fast ASL and English translation. We leverage the observation that deaf people have started to use Augmented Reality (AR) glasses as everyday wearables [44, 51]. Upon emergencies, (s)he makes a 911 call from their smartphone. The AR glasses capture live videos of the signer's sign motions. The video frames are then forwarded to the smartphone via Bluetooth for sign recognition and sentence translation. The translated English texts are converted to voices, which are sent to the 911 operator over the call. The voice responses from the operator are converted to ASL sentences and sign animations of a 3D avatar, which are rendered on the AR glasses.

To better fit an emergency scenario, Sign-to-911 adopts traditional AI/ML models. They use fewer model parameters, 2~3 orders of magnitude lower than recent deep learning models for ASL sign recognition. To offer high-quality translations, we do not treat ASL signs as arbitrary hand gestures

in space and over time for feature extraction and recognition. Instead, our novel solution leverages the ASL domain knowledge to ensure its translation accuracy: Each sign assumes well-defined motion patterns, and ASL has structured but simpler syntax rules. By extensively leveraging recent component algorithms from graphics, vision, natural language processing (NLP), and AI/ML, we may achieve fast and accurate recognition and rendering at both the sign level and sentence granularity.

We have implemented Sign-to-911 as user-space applications on commodity AR glasses (priced at about 350 USD) and Android phones. Our system can cover about 550 distinct signs (4.5× improvement over prior work) and support fingerspelling, sufficient for 911 emergency call situations. We further evaluate our design with 6 ASL signers. Our models achieve an average accuracy of 88.53% (for individual sign) and 91.37% (at the sentence level), with an average end-to-end latency of 0.55s for 550 signs (112× reduction from running prior proposals directly on smartphones [33]). Our study obtains IRB exemptions before actual tests, and also yields arguably the largest ASL trace set on AR glasses; we plan to release it in the future. The remaining sections of this paper elaborate on the details of Sign-to-911 system.

2 BACKGROUND AND MOTIVATION

We introduce the necessary background on sign language and motivate our work.

2.1 American Sign Language

This study focuses on American Sign Language (ASL), the popular sign language used by the Deaf and Hard of Hearing communities in the United States and some regions of Canada. According to the National Institute on Deafness and Other Communication Disorders, approximately 37.5 million American adults (15%) have some trouble with hearing [45]. Among them, approximately two million individuals are categorized as deaf. ASL is an indispensable communication paradigm for the deaf and hard-of-hearing community; it has an estimated half to two million users in the US [43]. Providing accessible and inclusive communications, such as an ASL-based emergency call system, is valuable to ensure equal access and effective interactions for this community, particularly in emergency scenarios.

ASL has its own vocabulary and grammar. ASL has two major word formats: signs and fingerspelling. Signs are iconic gestures used to represent words for common objects, actions, and concepts. The ASL dictionary [66] has recorded around 1600 signs, and a recent linguistic study [54] has documented 2723 signs. Fingerspelling is used to express words that do not have an equivalent sign in ASL. It utilizes designated handshapes to represent the English alphabet, spelling out words such as names, addresses, and locations. ASL has its own syntax system [37] different from that of English. For example, the English sentence "I need water" in subject-verb-object (SVO) structure can be expressed as "WATER I NEED" (OSV) in ASL. The capitalized words here are ASL glosses, which are English words representing corresponding signs. Like most natural languages, ASL has more than one right word (sign) order.

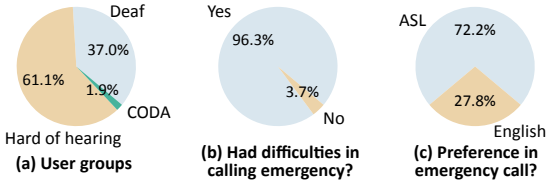

Figure 1: Emergency call survey for ASL users

In summary, sign language like ASL both poses design challenges and offers new opportunities. On one hand, its translation is challenging because both individual sign word recognition and sentence-level translation are needed. On the other hand, it uses well-defined human gestures and simpler, yet structured syntax. The fundamental problem is that, ASL is a visual and natural language, and solutions require techniques from NLP, vision and graphics.

2.2 Emergency Calls for Deaf People

911 provides emergency call service in the US. While offering multi-language support, the current 911 system does not supply an efficient communication channel for ASL users to communicate with 911 operators. This is largely due to the gap between ASL and other spoken languages; ASL is a visual communication language that requires runtime viewing for correct interpretation. Our interview with a local police officer further confirmed that an accessible solution is still lacking for deaf individuals to call 911.

To bridge the gap, sign language users conventionally use two alternative schemes to make their emergency calls: text-based communication, or relay services. Text-based communication services, such as real-time-text (RTT) [16] and teletypewriter (TTY) [17], transcribe voice to text and allow ASL users to input text during a call. Text-to-911 [18] allows individuals to send text messages to emergency services. However, these services are not applicable to many deaf individuals since not every sign language user has the same level of proficiency in a written language (say, English).

The other approach is to use relay services, such as video relay service (VRS), during a call. The user streams video to an interpreter who translates the sign language or into a voice message for the 911 operator. However, this approach requires high-speed network for video calls, which may be unavailable during emergencies. Moreover, the interpreter shortage makes this approach difficult to scale [64].

User Survey. We conduct a survey to learn about the 911 call experience for ASL users. We collect responses from 54 volunteers in the ASL community via anonymous online

ASL forums. Participation is open to all community members, with no material or financial incentives offered, except for a courtesy cup of coffee. As shown in Figure 1, among the participants, 37.0% are deaf, 61.1% are hard of hearing, and 1.8% are children of deaf adults (CODA). The survey shows that 96.3% of them have experienced difficulties communicating with an emergency call operator, mainly because text-based or VRS services are not readily accessible. Individuals who can speak but are deaf or hard of hearing often have to repeat their addresses and situations until the requested help arrives due to a lack of feedback. On the other hand, those who cannot speak are unable to make 911 voice calls. Furthermore, 72.2% of the participants prefer to communicate in ASL during emergency calls since ASL is their primary language in daily life and not all deaf individuals are fluent in English. This further validates that English proficiency cannot be presumed within the Deaf community, thus highlighting the crucial role of ASL for emergencies.

Goals. The survey result motivates us to devise an effective solution for the ASL community to make emergency calls. Specifically, the solution should have four features: (1) Accurate bi-directional translation: it must support accurate, two-way communications between the signer and a 911 operator: ASL signs-to-spoken English and spoken English to ASL signs. (2) Fast translation: the bi-directional translation must be fast enough to ensure liveness and interactiveness of the call conversation [39]. (3) Easy to use and carry: the solution should be easy to use and carry with the signers, since a significant portion of emergency situations arise on the road, or at remote or not readily-accessible locations. (4) Operation in the absence of high-speed Internet access: we do not assume infrastructure support for real-time video transfers and interpretation of ASL, except for a conventional voice call service. This is a common scenario for emergencies in remote or not readily accessible regions.

Limitation of Current Machine Translations for ASL Signers. The current machine-based solutions for ASL signers cannot meet all goals and well serve the emergency calls. The fundamental problem is that, ASL is a visual language by nature, and any translation must capture and recognize each sign (or fingerspelling) in an accurate and timely fashion. Existing software solutions, such as SL-GCN [33] and I3D [36], heavily rely on computer vision techniques. Therefore, they require high-end GPUs for fast processing, thus unsuitable for signers without access to cloud/edge services. Network communications with cloud/edge servers may incur long latency and compromise call interactiveness.

The proposed hardware solutions, such as gloves [77] and smartwatches [30], capture and recognize signs with sensors and hardware processing. However, they are deemed impractical for everyday wear and lack the necessary resolution to fully capture sign features. Moreover, they only provide unidirectional communication, which does not fit the two-way communication between the caller and the 911 operator.

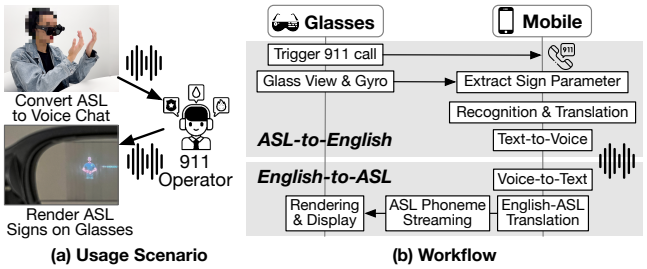

Figure 2: Workflow of Sign-to-911

3 SYSTEM OVERVIEW

We now describe Sign-to-911. The system components and workflow for Sign-to-911 are shown in Figure 2. The signer wears an assistive pair of Augmented Reality (AR) glasses, which interact with his/her smartphone. With a click on the 911 icon on the glasses, the user makes an emergency call, which is initiated through his/her smartphone. The smartphone subsequently works with the AR glasses for sign-to-English translation. The translated spoken English will be sent to the 911 operator via the 911 call. The voice response from the operator will be translated to ASL signs, which are rendered on the AR glasses. It thus offers bidirectional call communication between the ASL signer and a 911 operator.

We select the AR glasses (illustrated in Figure 2), rather than other wearable hardware (e.g., gloves or smartwatches), for ASL signers. They have a lightweight design and non-distractive displays, and limited on-glass processing capability, thus allowing users to wear them like sunglasses or normal eyeglasses. They typically have a built-in camera, a color display, a gyroscope sensor, and a speaker (for music listening), as well as a Bluetooth interface that readily connects to nearby smartphones at speed up to 150KB/s [65]. The assistive glasses keep a signer's hands free for signing, while allowing him/her to see the operator's responses in ASL sign animations on the display. Furthermore, assistive AR glasses of this type are quite affordable, with current prices ranging from 350 USD [31] to 1300 USD [71]. Some deaf people have already incorporated the usage of AR glasses into their daily routines [44, 51], although the current models do not yet support ASL accessibility. Based on our survey of 105 sign language users, 91.4% of respondents indicated their willingness to use glasses as a communication aid.

Our wearable system takes a software-centric approach. We thus design two pipelines to enable two-way communications: ASL-to-English and English-to-ASL (Figure 2). On the ASL-to-English translation, we capture each ASL sign by exploiting the sign parameters and simple machine-learning models. These parameters also provide a standard description of the signs. We thus enable fast recognition of signs

based on the ASL domain structures. Once signs are recognized, we leverage the emergency call context and syntax model to convert sign sequences to English sentences. On the English-to-ASL translation, we first translate each English sentence to the corresponding ASL gloss with the syntax model. We then produce the ASL sentence using the basic language unit: phoneme. We further compress phonemes leveraging kinematic correlations, in order to feed the ASL streaming into the low-rate Bluetooth connectivity. Finally, the glasses decompress the phoneme stream and render the ASL signs and sentences in front of the user's eyes.

Given the glass-smartphone setup and simple workflow for each user, we address two key challenges with novel domain-driven designs to enable ASL-to-English translation and English-to-ASL production.

Accurate and Fast Sign Translation. A primary challenge in our Sign-to-911 scheme is to capture and recognize each ASL sign without relying on edge/cloud infrastructures. Sign language is a natural, visual language that is conveyed through a sequence of gestures. We thus have to use machine learning models that can analyze a substantial number of sign features in a timely fashion. This may in turn increase model complexity further. Note that the used models must be processed on mobile and wearable devices in our system setting. Consequently, it remains difficult to develop and operate a lightweight solution that translates sign language with high fidelity at runtime. To address this issue, we depart from the popular paradigm of deep-learning based models. Instead, we propose a novel method that exploits sign parameters derived from linguistics and extensive domain knowledge in ASL. Our method enables us to capture sign gestures efficiently and ensure accurate and fast recognition.

Efficient Sign Production from English. Many ASL users consider ASL to be their primary language. However, producing ASL signs from English sentences and rendering them on AR glasses can be challenging. Each sign involves the coordinated actions of fingers, palms, and arms. Moreover, the AR glasses do not have enough processing capability, and the Bluetooth connectivity between the glasses and the smartphone cannot transfer and render each produced sign gesture on time. Our proposed solution utilizes the phonetic parameters of ASL to generate accurate signs. We further leverage kinematic correlations for efficient sign compression. Consequently, we achieve high-fidelity ASL production on smartphones and accurate rendering on glasses.

4 ASL-TO-ENGLISH TRANSLATION

Our first task is to provide translation of ASL into English during a 911 call given the glass-mobile setting. Figure 3 shows the three main steps. First, we capture the sign parameters for recognition (§4.1). Second, we leverage ASL domain knowledge to perform fast sign recognition with the parameters (§4.2). We further construct coherent sentences from the sign sequences and translate them into English (§4.3). Our lightweight translation pipeline works with limited capabilities of smartphones and AR glasses.

Our approach differs from all prior proposals [23, 33, 36]. The prior approaches treat signs as a spatial-temporal sequence of hand movements and uses computer vision techniques for sign recognition. However, these methods often result in large, complex models that require significant computational resources for both feature extraction and recognition. For example, I3D [36] uses video as its input and requires 4.8GB GPU memory. SL-GCN [33] takes skeleton sequences as input but still requires 16 million parameters and 5 seconds to recognize one sign with a powerful GPU. Consequently, these models are not suitable for this setting.

4.1 Capturing Sign Parameters

We devise a novel scheme to extract sign parameters from the video frames capturing the signer's gestures by the AR glasses. The on-glasses camera has a field-of-view (FoV) encompassing the common signing space [73]. Our approach exploits the premise that ASL is a visual, yet natural language with linguistic features similar to those found in a spoken language. Like the consonants and vowels used in spoken languages, ASL signs follow a set of decomposable patterns and rules (i.e., sign factors). Such factors set the foundation for differentiating signs from arbitrary human gestures.

Linguistic factors for ASL signs ASL signs are generally classified into one-handed and two-handed signs. ASL further uses four main linguistic factors to define a sign: handshape, palm direction, location, and hand movement [63][1]. The handshape describes the configuration a hand assumes when making a sign. The ASL dictionary lists 40 handshapes to organize signs [63] (see Figure 4 for illustrative examples). Palm orientation indicates the orientation on how the palm of a hand is turned. It has six choices (up, down, left, right, front, or back) in ASL. The location specifies where a sign is formed. It is expressed with respect to close body areas when a sign is performed (say, "in front of face," "near right ear," "right cheek," etc.). Movement specifies the direction and trajectory of how a sign moves in space. It also includes repetition, motion magnitude, and speed. Hand movements are not arbitrary in ASL. They exhibit three patterns [19]: (1) based on the movement directions, signs can be classified as static (a.k.a. fingerspell), unidirectional and repeated signs; (2) for two-handed signs, both hands may move or only the dominant hand moves; (3) if both hands move, the movement pattern can be classified as symmetric, parallel, or alternating (Figure 5). In summary, from the ASL linguistic

[1]Nonessential parameters, e.g., body and facial expressions, provide additional information such as tone.

Parameter	Meaning	Dimension #
Hand number	One-handed or two-handed	2
Handshape	Handshape sequence over time	$40 \times 2 \times t$
Wrist trajectory	Wrist position over time	$3 \times 2 \times t$
Palm orientation seq.	Palm direction over time	$3 \times 2 \times t$

Table 1: Sign Parameters

standpoint, each sign can be well specified using the above four main factors. Furthermore, each linguistic factor only assumes a limited number of choices for defining a sign.

Issues with direct utilization of ASL linguistic parameters However, we cannot directly use the above four sign factors to capture and recognize a sign. First, the 40 handshapes are well defined by the ASL dictionary [63]. However, most proposed systems cannot obtain such handshape information directly. Instead, they only have access to the visual information obtained through the sequence of images or video frames when a signer makes a sign (say, through the AR glasses in our case). Second, location (with respect to close body parts, e.g., "below right shoulder") is an important linguistic factor to define a sign in ASL. This is also deemed to be difficult in reality, since sensors (such as AR glasses) cannot capture the full view of the signer's body. Therefore, they cannot learn the relative positioning of body parts versus hand sign. In fact, we are unaware of any prior scheme that uses the location factor in ASL to classify a sign.

Sign Parameters for ML. We have identified four types of sign parameters (see Table 1) that can be used by our ML algorithms. These parameters are independent of the signer's body shape and hand size, thus being robust across signers:

(1) **Hand number:** A two-dimensional vector denoting the probability of the sign being one-handed or two-handed.
(2) **Handshape:** Handshape sequence over time t. Each is represented by probabilities of 40 basic ASL handshapes.
(3) **Wrist trajectory:** The wrist's coordinates (x, y, z) over time t, captured through hand detection and depth estimation based on the hand size.
(4) **Palm orientation sequence:** The palm's orientation (α, β, γ) over time t.

These four parameters provide an ML-friendly representation of the sign language. We next explain how to extract these parameters on the glass-mobile setting.

Sign Parameter Extraction. We extract the above sign parameters from the video frames captured by the AR glasses. The extraction procedure has three steps: skeleton extraction, segmentation, and adaptive extraction.

Skeleton Extraction. Extracting hand skeletons from video frames is supported by mobile libraries [42], which are able to estimate the 3D joint positions of fingers even when some occlude from each other. However, on-glasses processing can only achieve 3 FPS, much lower than the natural signing speed. We thus offload processing to the signer's smartphone to extract hand skeletons. Since the signer wears the AR glasses as (s)he makes a sign, one new issue arises: head movements affect skeleton positioning in the camera view. We leverage the gyroscope data to address this issue. With such data being further streamed from the glasses to the phone, the signer's hands can be calibrated relative to a fixed position, thus reducing the impact of head movements.

Segmentation. The captured wrist trajectory sequences need to be segmented into signs. We exploit the idle states of the hands in segmentation. Prior ASL study [19] shows that, pause and neutral position are critical to identifying pacing between sequential signs. Specifically, the signer may pause after the current sign, before transitioning to the next sign. Alternatively, the signer's hands may return to a neutral position (i.e., where the hands remain relaxed, typically at the waist level in front of the body [63]), before starting the next. We thus use the pause time and the hand's neutral position to detect the borderline of sequential signs. The sign parameters are then ready for final extraction.

Adaptive Extraction. The hand number and movement can be directly extracted from the hand skeletons. Palm orientation can be computed from the normal vector of the palm plane. The extraction of handshape from each frame is more involved. We first calculate the angles of finger joints [74] to minimize the impact of different hand sizes among signers. We then match joint rotations with the 40 base ASL handshapes using a multilayer perceptron classifier[28], which assumes a hidden layer of size 64. Since each hand may contain one or two handshapes during the sign [24], we further merge the handshape sequence to four vectors (two for each hand-start and end). This merging process helps to reduce recognition errors that may occur with individual frames.

To further optimize processing at the smartphone, we design an adaptive handshape extraction scheme. Our approach involves taking different sample rates on dominant and non-dominant hands for handshape recognition. In ASL, the dominant and non-dominant hands have different impacts on sign meaning. The non-dominant hand typically undergoes fewer changes in position and shape than the dominant one, thus requiring lower sample rate. A signer may set his/her dominant hand via configurations; the default dominant hand is the right hand. This adaptive extraction further reduces processing time by about 25%.

4.2 Sign Recognition from Parameters

The signer's gestures are thus converted into a sequence of sign parameters. The next task is to perform sign recognition from such parameters. Note that our recognition must classify word signs from fingerspelling-based terms.

We take a two-step approach. We first categorize a sign candidate into one of a few categories. We then compare the similarity of the collected sign parameters with all signs in the category. This two-level, hierarchical recognition scheme

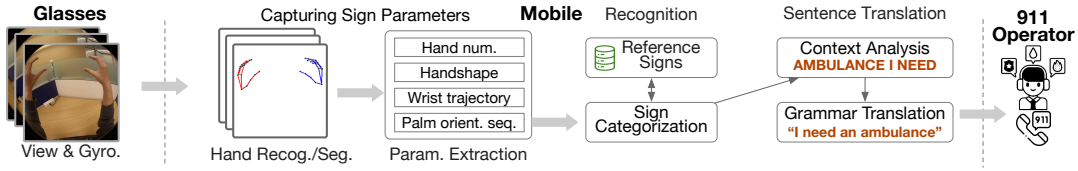

Figure 3: ASL-to-English Pipeline

Figure 4: Basic handshapes from glass view (40 in total)

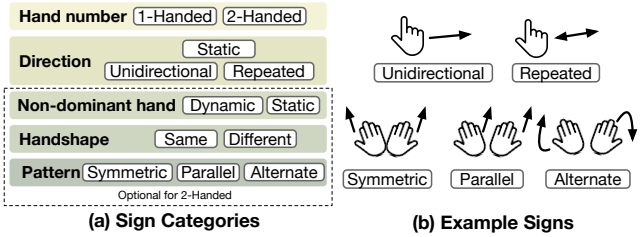

Figure 5: Sign Categories and Examples

scales better than flat recognition, where parameters of a new sign are directly compared with all candidate signs (i.e., hundreds or thousands of signs). It reduces processing complexity and enables fast recognition on mobile devices.

Sign Categorization. As shown in Figure 5(a), signs are first classified along five dimensions: (1) *Hand number:* signs are first classified into 1-handed and 2-handed with the hand number parameter; (2) *Direction:* using wrist trajectory, signs are classified into unidirectional, repeated, and static. Note that, the static sign (a.k.a. fingerspell) denotes a single letter in ASL; a corresponding fingerspelling module (to be elaborated next) is triggered for further recognition; (3) *Non-dominant hand behavior:* we further decide whether the non-dominant hand is dynamic or static, by using the wrist trajectory of this hand; (4) *Handshape*: the handshapes of both hands are further identified based on the 40 candidate handshape set; (5) *Patterns:* signs are finally classified by movement patterns mandated by ASL: symmetric, parallel, and alternate. The movement patterns are extracted by combing the wrist trajectory and palm orientation sequences. Figure 5(b) shows a few examples of different sign categories. The above categorization can be readily obtained from the sign parameters during the training phase, and stored in a reference sign database.

Our evaluation shows that, the above step could reduce the search space by an order of magnitude. The sign category enables us to recognize signs accurately, even when there are slight variations in sign parameters due to differences in signing habits and speeds.

Recognition of Word Signs. We next use a fast dynamic time warping algorithm [52] to match the captured hand trajectory with those candidate signs in the same category, and sort out the Top-k candidates based on the weighted similarity of all sign parameters. The weights are learned with a linear regression algorithm to suppress noises. Our evaluation shows that, 96.5% of signs can be correctly identified when k=5. The accuracy can be further improved using context information (§4.3).

Recognition of Fingerspelling-based Terms. We exploit two distinctive features of fingerspelling to classify such terms from word signs in ASL. Fingerspelling is typically performed by the dominant hand, with the wrist retaining a stationary position, thus providing a distinctive feature from other signs. Moreover, the whole word, but not individual letters, must be fully expressed with fingerspelling [56].

Our recognition module for fingerspelling thus works in two steps. Each letter is recognized from each frame using palm orientation and handshape. Multiple letters are then merged into a word. Since ASL fingerspelling uses wrist movement to indicate repeated letters in the word (e.g., Z-O-O), we apply wrist movement checker to decide the repeated letters. We further call the English spell checker [59] to minimize misrecognition. Our system is designed to trigger fingerspelling recognition under two situations: (1) during a conversation, such as when a user needs to sign a name, and (2) when the system fails to recognize a sign. In the first case, fingerspelling recognition is initiated by placing the dominant hand in a fixed position in front of the glasses. For the second case, the system utilizes confidence scores to detect uncertain recognition and allows the user to correct signs using fingerspelling.

4.3 Sentence Translation

Given the recognized ASL signs, we next translate them into an English sentence. We address two issues in the translation. First, ASL follows its own grammar. The grammar differences arise in two main aspects: different word order and simplified structure. For the basic English word order of subject-verb-object (SVO), it often shows up as subject-object-verb (OSV) in ASL. For example, the sentence "I need an ambulance" in English SVO order is translated into "AMBULANCE I NEED" in ASL. Moreover, ASL adopts a simplified structure. For instance, ASL does not have "be verbs" (i.e., am/is/are). ASL also does not use separate signs for articles (a/an/the). To solve the grammar issue, we have devised a grammar translation model that converts ASL sign sequences into an English sentence. The model works even though signers may have different signing habits in terms of word order. The

second issue is that ASL signs may appear as homophones, which assume identical sign parameters but convey different meanings [29]. To address this issue, we disambiguate such signs using context information on the usage scenario. A context model is used to identify the correct sign out of multiple choices in the given context.

Grammar Translation. For grammar translation, we build an ASL syntax model to provide mappings between the ASL grammar orders and the corresponding English grammar orders, such as the "OSV↔SVO." To train the model, we extract the ASL syntax order and map it to the English syntax order using a parallel corpus, which includes the ASL-English translations. During grammar translation, our model first parses the sign sequence and identifies the best alignment of its syntax order. For example, as shown in Figure 6, the incoming signs "AMBULANCE I NEED" are mapped to the order "OSV." The model then maps it to the English grammar order "SVO." Second, it fills missing elements in ASL, such as the "be verbs" and articles. If no exact match is found in the known ASL order, the most similar order matching the incoming sign sequence will be applied. This can be done by matching the subtree structures [13].

Our syntax model is trained using a public parallel corpus from the authoritative ASL resource Signing Savvy [58]. The dataset contains 1233 translations between ASL gloss sequences and their corresponding English sentences. Figure 7 shows some example sentences. We thus have learned 209 mappings between ASL and English. This process results in an accurate and efficient translation that produces grammatically correct English sentences from ASL sign sequences.

Leveraging Context Information. Our system further leverages 911 context information from the real-life emergency call conversations to refine the translation, as well as recognition of uncertain signs. We construct our context model using the following process: (1) classify the 911 conversations into five emergency types based on the answers to the type of emergency, (2) track questions asked by the 911 operator, and (3) create a context model using the tuple of [Topic, Question, Response]. The weights of sign candidates are then adjusted based on each tuple value. For example, if the question is on the color of the victim's clothes, the context model selects color-related words in the recognized candidates as the final response. By incorporating contextual information into our recognition and translation, we improve the system accuracy. This becomes important in cases where signs are ambiguous or unclear.

After the translation, the resulting English sentence is streamed to the glasses via Bluetooth. The AR glasses generate corresponding voice message, which is fed in by the ongoing call at the smartphone and sent to the 911 operator.

4.4 Miscellaneous Issues

We next discuss miscellaneous issues on Sign-to-911 design.

Compound Signs. In ASL, a compound sign combining two or more individual signs can be used to convey a single meaning. For instance, the *parents* sign is a compound one, by combining the signs for *mother* and *father*. By breaking down into their individual components, we can efficiently handle such compound signs without increasing design complexity.

Voice/Text Interface. Most smartphones do not allow generating voices for phone calls directly from a mobile application due to security concerns. We thus send the translated English from the phone to the glasses. The AR glasses use their local text-to-speech to produce the voice, which is output to the speaker on the glasses. The glasses further play sounds to let the operator know that the user is signing. This way, the signer communicates with the 911 operator. Furthermore, we provide text captions for both the signer and operator's messages, which are displayed on the AR glasses for enhanced clarity and understanding.

Extreme Case Handling. In the extreme case the signer cannot make signs (e.g., got the hands/arm hurt in an accident), our system generates an automated, on-the-spot description. If no sign is detected after a 911 call, the glasses produce a voice message containing essential information, including name, age, and location, as well as surrounding objects/buildings learned from object recognition. This offers a critical lifeline for deaf or hard-of-hearing individuals during emergencies.

5 ENGLISH TO ASL PRODUCTION

During an emergency call, an ASL signer also needs to comprehend voice responses from the 911 operator. Current speech recognition schemes [14] can readily convert the operator's voices into English texts. However, directly presenting such texts may hinder usability, as many ASL users primarily communicate through the sign language rather than in English. To address this challenge, we construct an English-to-ASL pipeline (Figure 8). The pipeline generates signs from the operator's responses, animates the produced signs using a 3D avatar, and renders them on the AR glasses. This ensures that signers can directly understand the responses via signs, resulting in more effective call conversations.

There are two intuitive approaches to animate ASL signs on the glasses: one is to perform ASL translation and rendering on the smartphone and stream the resulting animation video to the glasses; the other is to perform text-to-ASL translation and sign animation production on the glasses. However, neither of the two is desirable. The first approach incurs long delay due to encoding/decoding latency and prolonged transfer of video frames over the low-rate Bluetooth connectivity between the glasses and the phone. The second

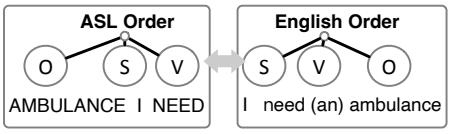

Figure 6: Grammar Mapping

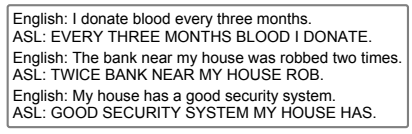

Figure 7: English-ASL parallel corpus

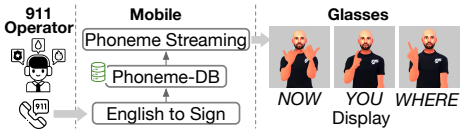

Figure 8: English to ASL Pipeline

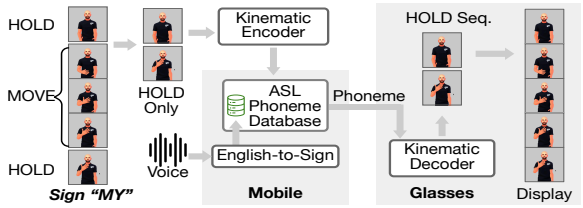

Figure 9: Phoneme DB construction & streaming

approach is too heavy to be operated on AR glasses with limited processing capacity and power budget.

We take a novel approach to lightweight production of ASL sign animations on AR glasses. To this end, we exploit the well-accepted MOVE-HOLD model [38] to describe the temporal units of signs, and achieve a higher degree of compression for sign animation.

Specifically, a HOLD state records a static gesture, while a MOVE state captures the transition between two HOLD states. An ASL sign can be reconstructed using one to five HOLD states (ASL phonemes). Each phoneme effectively takes a snapshot on all involved sign parameters at the time instant for a HOLD state. It can be acquired by taking snapshots from the sign parameters discussed in §4.1. These phonemes serve as "key frames" during sign production.

Consequently, we design a three-step pipeline for ASL production based on phonemes (see Figure 8). The pipeline converts the incoming voice messages to phonemes on the smartphone, and streams the generated phonemes to the glasses for on-glass animation. The core of our pipeline is a module that produces ASL signs using phoneme sequences and compresses phonemes with kinematic correlations. Our scheme enables high-fidelity transfer of ASL signs via low-rate wireless connectivity (as low as 3.8KB/s).

English-to-Sign. In the first step, incoming voices are converted to English sentences using the speech-to-text conversion module on the smartphone. We then translate the English text to ASL gloss (i.e., word sign) sequences. Note that ASL and English have different word orders. To ensure correctness of the produced gloss order, we reuse the syntax model in §4.3 and convert English sentences to corresponding gloss sequences. We match the English syntax order and ASL order, while dropping "be verbs" and articles. For example, the English sentence "What is your emergency" is translated as "YOUR EMERGENCY WHAT." Finally, we convert the gloss sequence to sign animations. Note that, each gloss must be matched by a sign of the same word type, since a gloss may assume multiple word types. For example, "SECOND" can be either an adjective or a noun, denoted with different sign gestures. Moreover, our animation database contains all frequently used signs. In case a gloss does not match any sign in the database, we create the fingerspelling animations from individual letters of the word.

Sign Production with Phoneme Sequences. We use the phoneme sequences to visually produce each recognized gloss. Direct usage of sign parameters of §4 does not yield an efficient method for producing a sign. This is because sign parameters record complete hand skeleton movements over time. Transmitting this entire skeleton sequence to the glasses results in prolonged latency. In contrast, phonemes offer a more concise way to convey visual information for a sign. Phonemes follow the MOVE-HOLD model [38], which includes HOLD states. A phoneme captures a static snapshot of the sign parameters at a particular timestamp. To capture these HOLD states, we first use body and hand recognition algorithms [12, 74] to extract the 3D skeleton sequence from sign language videos [36]. We then identify HOLD states in the sequence based on movement speed and direction, which are verified by human experts to ensure accuracy. With these phonemes at HOLD states, our sign rendering uses spherical interpolation algorithms [50] to approximate the MOVE state between two HOLD states.

Compression with Kinematic Correlations. We further compress phonemes by leveraging kinematic correlations. Note that, hand joint movements must comply with human kinematic constraints [22]. For instance, a single flexor tendon can control the flexion of multiple joints, such as the dip joint and pip joint [40]. With the kinematic constraints, each hand could be represented with a subset of 10 out of the full 15 hand joints. The degree of freedom of each joint rotation could also be reduced from 3 to 2 or 1 (depending on specific finger joints). We build a kinematic encoder/decoder with a Linear Regression (LR) algorithm. Only 10 joints and corresponding reduced rotations need to be transmitted during rendering. The remaining joints and rotations are inferred from the LR at the glasses. Our approach retains production accuracy under the limited bandwidth.

Figure 9 illustrates the construction and usage of the ASL phoneme database for phoneme streaming. We first extract and compress the HOLD states of each sign using the kinematic encoder, which is then stored in a local database at the smartphone. During translation, the phone looks up each gloss in the phoneme database and streams its compressed version to glasses. The AR glasses decode the body/hand movements with the LR and produce sign motion sequences

by interpolating the phonemes of HOLD states [50]. Finally, the glasses render the signs with a local 3D avatar in real-time. The approach reduces the required bandwidth by 50x compared with directly streaming hand skeleton sequences, ensuring both low latency and high fidelity.

6 SYSTEM IMPLEMENTATION

The implementation of Sign-to-911 is in software only. It operates as apps on the phone and the AR glasses, respectively. The main components are shown in Figure 10.

Application on AR glasses. The application on glasses runs as an Android app, as shown in Figure 12, on the lightweight Android Go Edition [5]. It displays ASL animations, text transcriptions of messages from both sides, and visual cues for the camera's FoV. The app includes 3015 lines of Java and 1149 lines of Kotlin. For ASL-to-English, the app acquires real-time camera views and gyroscope data with Android APIs [4, 7]. The glasses record camera views into *H.264* videos at 15 FPS, which are later decoded by the smartphone for sign translation. Both encoding and decoding use Android MediaCodec [6]. For ASL reproduction, the glasses receive encoded ASL phonemes through the communication manager, and reproduce signs using a reproduction module. This module loads a 3D avatar from local files produced by Mixamo [72]. To reduce rendering overhead, we downsize the texture and decimate polygons to create an avatar with 66 bones, 132 joints, and 28,106 triangles baked as a 2.4MB file in .glb format. The reproduction module animates this avatar with skeleton transformation and renders it with Google Filament [25].

The AR glass uses a Communication Manager module (550 lines of Java) to enable glass-phone data communications. We use Bluetooth RFCOMM [3] for reliable data transfer. We have designed packet abstraction on top of the RFCOMM stream. The abstraction has two types of packets: control and data. Control packets coordinate between the AR glasses and the phone during 911 call initiation or termination. Data packets may carry three types of information: video frames, gyroscope data, and ASL phonemes. This communication manager is also reused by the smartphone app.

Smartphone App. We have developed an Android application for smartphones. This app has 4610 lines of Java and 2076 lines of Python code. The communication manager receives ASL-to-English data from the glasses and streams English-to-ASL phonemes. We use Android-MediaPipe [42] to capture the skeleton and run it on mobile GPU. Our translation module performs bi-directional translations between ASL and English. It is implemented with Andronix [9], thus allowing Python code and libraries to run on phones with similar performance as native apps. To build our recognition and translation modules, we utilize scikit-learn [49] for machine learning, and NLTK [13] for natural language processing in Python. Both the reference sign database and the phoneme database are embedded and preloaded in the app.

Moreover, we have implemented a 911 interface to make emergency calls. To capture the audio stream from a phone call, system permission is typically required due to Android privilege management [41]. To bypass this roadblock, we bound our mobile app with an Android accessibility service [1], thus enabling us to capture the audio stream through the *VOICE_RECOGNITION* audio source. Our app thus runs continuously as long as the service is activated, and provides swift response upon emergencies. The captured audio stream is processed by speech recognition using VOSK library [14]. For ASL-to-English, we call Android's built-in text-to-speech engine [8] to generate English voices from the text.

7 EVALUATION

We next evaluate Sign-to-911 on commodity smartphones and assistive AR glasses. We first introduce our evaluation methodology. We describe our model training results and assess both ASL translation and production. We further conduct a user study on signer's experiences.

7.1 Methodology

Experimental Setup. We evaluate Sign-to-911 on off-the-shelf AR glasses and smartphones, as shown in Figure 11. Specifically, we use INMO AIR [31] assistive AR glasses running Android 10 Go. The glasses are equipped with a quad-core Cortex-A53 processor (1.4GHz), 2 GB RAM, and 32 GB flash memory. The smartphone, OnePlus 10 Pro 5G, runs Android 12 with a Qualcomm SM8450 Snapdragon 8 Gen 1 processor, 8GB RAM, and 256G storage. We further use an Eversame USB Digital Tester to gauge power consumption.

Glass-view Sign Traces. Our ASL-to-English model training requires: 1) ASL video and gyroscope data captured from the AR glasses; 2) the sign sequences and corresponding English text translations. However, there are no such public traces to the best of our knowledge. Consequently, we decide to collect our own glass-view ASL traces.

As our solution is designed for emergency calls, we use two text-based 911 content sources to generate glass-view traces. The first from Montgomery County [15] contains over 600K emergency call records, each of which offers a summary report for the 911 call. The second is the 911 response template from Eugene Police Department [20]. It contains 46 questions covering 4 emergency scenarios (medical, fire, police, and hybrid). 911 operators use it to quickly understand the situation by asking appropriate questions.

We invite signers to drive ASL conversations in these two datasets. To ensure accurate representation of natural signing habits and experiences, we select a group of six signers in our

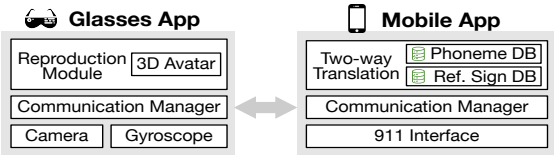

Figure 10: Sign-to-911 Implementation

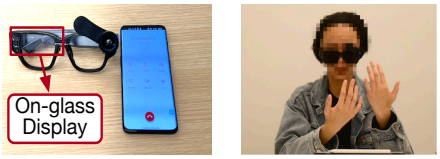

Figure 11: Experimental Setup

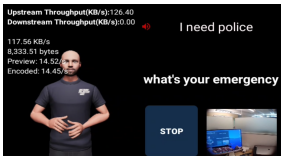

Figure 12: Glass App

study. This group includes two native ASL signers, an ASL linguistic expert, and three ASL students with over two years of signing experiences. The signers sign the sentences, and their signing motions and gyroscope readings are collected by two types of devices: AR glasses [31] and a head-mounted action camera [32]. We label the collected data with the sign sequences and corresponding English sentences.

The above collection process results in three glass-view ASL datasets, covering 249 GB of video traces and more than 11.5-hour user-signing actions. Dataset-1 (D1) logs 478 distinct ASL word signs using the first source [15] to cover the 911-related sign words. Dataset-2 (D2) is built from the 911 response template [20]. For each question, multiple answers are generated to cover various real-world scenarios. In total, we generate 180 Q&As, which use 202 distinct ASL signs. All questions and answers are validated by signers.

D1 and D2 together cover more than 550 distinct word signs used in 911 emergency calls. In addition to word signs, the remaining contents, e.g., names, addresses, and numbers, are recorded with fingerspelling motions. We use D1+D2 (i.e., the superset by merging D1 and D2) to train the sign recognition model. We further use D2 to evaluate the grammar/syntax model in the 911 settings.

We construct Dataset-3 (D3) to evaluate our overall system. D3 contains detailed 911 call conversations, both real ones and machine-generated ones. We download 4 real-life 911 call recordings [46, 47] and translate these conversations into ASL. Furthermore, we use ChatGPT-3.5 [48] to generate 30 artificial conversations with 150 Q&As for emergency calls on three topics (medical, fire, and police). All conversation contents are verified by our signers. To our knowledge, we have produced the largest glass-view ASL dataset to date.

Phoneme Database for ASL Production. Different from the glass-view sign traces, the phoneme database requires front views of signs, just the opposite to the glass views. We construct this database by extracting phonemes from the public ASL video datasets [36]. For each sign, we label its HOLD-state frames, extract phonemes, and compress them using kinematic correlations. Our database covers 3100 signs, larger than those provided by ASL directory [63] and linguistic study [54]. We collect more compound signs, while prior efforts [54, 63] did not. The words not covered by these signs are expressed through fingerspelling.

Ethical Considerations. This research received approval from UCLA Institutional Review Board under IRB#23-000239. We provided information sheets to all participants, and their confidentiality and privacy were ensured. The user study followed ethical guidelines outlined by our institute. The used glass-view camera captured the sign motions view without any facial features of a signer. Furthermore, no actual 911 calls were made during this study. In summary, our research was carried out in an ethical and responsible manner.

7.2 Component Evaluation

In this section, we evaluate three main components of Sign-to-911: sign recognition model, grammar/syntax model, and English-to-ASL production.

Sign Recognition Model. We use the combined datasets of D1 and D2 to train the sign recognition model. The combined dataset D1+D2 is split into training (80%) and test subsets (20%). We evaluate our model training using 5-fold cross-validation on the training subset. The training yields an average accuracy of 91.72%, with a standard deviation of 1.27% over the training subset.

We compare the accuracy with two state-of-the-art recognition models, I3D [36] and SL-GCN [33]. Both take the image view as input. For fair comparisons, these two models are also trained with the same training data subset as ours.

The accuracy results are shown in Table 2. It is clear our model has comparable accuracy in sign recognition as the SL-GCN model, but higher accuracy than the I3D model. The root cause analysis shows that, the inaccuracy of our model largely stems from those signs that are prone to sliding to the view boundary or outside the camera frame. For example, the signs for words BROWN and RED are close to the face, thus being on the view boundary. Consequently, the hand-tracking algorithm fails to accurately detect the skeleton for both signs. Minor adjustments to the camera view angle of the AR glasses may solve the problem.

Note that, we only compare our model with the vision-based solutions. Other sensor-based schemes are trained and tested using spatial-temporal sensory data, and cannot be trained using our visual datasets. Moreover, they have only tried with small datasets of 50-100 signs [23, 30], while our model covers about 550 distinct signs (4.5× improvements).

Syntax Model. We train our syntax model using the public dataset [58] with the approach described in §4.3. To evaluate the quality of syntax translation, we use D2 that has both English and ASL conversations. We evaluate the syntax model for both ASL-to-English and English-to-ASL translations, using the word accuracy metric (i.e., WAcc [35]). In

Model	Accuracy (%)
I3D	82.03
SL-GCN	89.06
w/o-cat.	85.72
Sign-to-911	88.53

Table 2: Sign Recog. Acc.

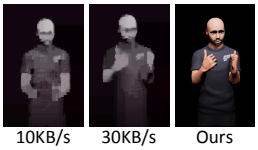

Figure 13: Visual Quality

ASL-to-English translation, we convert the gloss sequences using the syntax model, and compare them with the corresponding English sentence. In English-to-ASL translation, we convert the English text to the gloss sequence, and compare the obtained gloss sequence with the ground-truth one.

The evaluation results show that, the WAcc for our ASL-to-English translation is 93.96%, and the WAcc for English-to-ASL is 95.60%. Errors in ASL-to-English translation were primarily due to difficulties in filling in articles (a/the), while the errors in English-to-ASL translation were due to differences in word order in ASL, which did not affect the meaning. Overall, our syntax model shows high accuracy in mutual translations between ASL and English.

ASL Production. We compare the quality of ASL production based on video streaming and that using our glass-based animation. Figure 13 shows the visual effects produced under 10KB/s video, 30KB/s video, and our on-glass production. The video-based production results in poor video quality, making it difficult to recognize signs. In contrast, phoneme streaming and on-glass production achieve high-fidelity production with low-rate, mobile-glass communication.

To quantify the consumed bandwidth, we compare three streaming methods: full skeleton sequence streaming, phoneme streaming with/without kinematic compression. Experiments show that, phoneme streaming consumes the least bandwidth by transmitting only the essential HOLD states. Full skeleton sequence streaming uses 78.6KB/s bandwidth, while phoneme streaming without kinematic compression only consumes 11.2KB/s. With kinematic compression, the used bandwidth is reduced by 20.7× to merely 3.8KB/s.

ASL Animation Quality. We assess the quality of ASL animations by examining their recognizability, sign parameter accuracy, and the hand distance between the human signer and the avatar. First, our user survey reveals that all participants successfully identified the sign animation sentences in D3, despite their individual preferences for certain signing variants (§8). Second, a comparison of the sign parameters with their ground truth reveals that a significant 95.0% were accurately represented in the animation, according to the expert feedback. Finally, the average difference in hand position between the generated signs and the reference human signs (i.e., ground truth) is less than 4cm. This discrepancy largely resulted from variances in body shapes between the human signer and the avatar. In summary, our animations successfully emulate key movements and sign parameters. The aspects for further improvement include fluidity and reference accuracy; they can be enhanced with more human feedback and updated collision detection.

7.3 System Evaluation

Sentence Translation. We first assess sentence translation in both accuracy and robustness. We use word accuracy (WAcc) as the metric for accuracy, which is calculated for both gloss sequences and English sentences. We compare ours with I3D and SL-GCN. Since I3D and SL-GCN do not support segmentation and syntax translation, we apply the same scheme as ours in these steps. All models are trained with the combined D1+D2 set, and then evaluated on D3. Note that, D3 is never used in training and records real-world 911 conversations.

Table 3 lists the comparison between our approach and the two related models. Sign-to-911 yields 5.06% higher in WAcc than SL-GCN, and 11.19% higher than I3D. Our detailed analysis reveals that the context model indeed helps; it improves WAcc from 84.49% to 91.37%.

We also evaluate translation robustness with respect to new signers. We collect the sign videos from a new signer under emergency call settings. The signer has not appeared in the training datasets. Results show that Sign-to-911 still achieves 82.40% accuracy (shown in Table 3). Both sign parameters and categorization help to reduce the impact of varying sign habits and speeds. In contrast, I3D and SL-GCN barely achieve 37.50% and 56.32% accuracy, respectively. In contrast to the other two DNN models, our solution concentrates on essential parameters derived from the ASL domain. As a result, it mitigates the risk of overfitting when training with limited datasets.

Robustness Upon Emergencies Emergency situations introduce uncertainties beyond those in regular scenarios. Both signer and environmental factors affect data capture, potentially reducing sign recognition accuracy. Our solution must therefore address these challenges. We assess recognition performance in different settings, including fast-paced walking at 4 miles per hour, low-light environments with illumination of 40 Lux, and outdoors. Across all settings, we evaluated recognition accuracy on the same set of sign interactions from D3. Table 4 shows the comparison against the baseline lab environment with the illumination of 500 Lux. The recognition accuracies fall within the ±2%, indicating our solution is robust enough to handle broad signer and environmental conditions.

End-to-End Latency. We quantify the end-to-end latency for both ASL-to-English and English-to-ASL pipelines. For ASL-to-English, we measure the interval from when a sign is completed to when the corresponding voice is generated. We define translation latency as the time it takes for the model to produce a translation. The remaining components

Model	D3	New Signer
I3D	80.18	37.49
SL-GCN	86.31	56.32
Sign-to-911	**91.37**	**82.40**

Table 3: Word Accuracy (%) for Sentence Translation

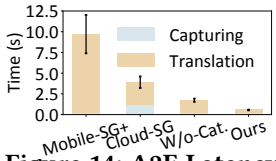

Figure 14: A2E Latency

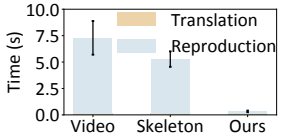

Figure 15: E2A Latency

Setting	Lab	Low-light	Walking	Outdoor
WAcc.(%)	90.7	91.7	90.9	88.9

Table 4: WAcc. in Different Settings

Approach	Accs.	Usab.	Oval.
Text-based	2.9	2.8	2.8
VRS	3.4	3.5	3.6
Sign-to-911	**4.2**	**4.3**	**4.6**

Table 5: User Study

(video encoding/decoding, transmission, etc.) are collectively referred to as capturing latency. We compare the latency with SL-GCN model on mobile (Mobile-SG+) and the cloud (Cloud-SG). For Cloud-SG, the phone transmits the video to the cloud through 5G for translation. The cloud runs the SL-GCN model with an Nvidia RTX 3090 GPU. The SL-GCN model running on mobile takes an average of 62 seconds due to its slow skeleton extraction module. Alternatively, we extract the skeletons with [42] and feed them to the model. This modified version is referred to as Mobile-SG+. We also gauge our model without sign categorization.

The results are shown in Figure 14. Our solution achieves an average time of 0.55 seconds, resulting in 17.4× reduction for Mobile-SG+ (9.7s). Compared to directly running SL-GCN on the mobile, our solution reduces 112× of latency. Although Cloud-SG reduces processing time with its powerful GPU, it still takes 7× longer than our solution. Compared with the pipeline without sign categorization, our solution reduces ASL-to-English translation latency by 8×, and achieves 3.1× reduction in end-to-end latency.

We quantify the English-to-ASL latency from when the voice is received to when the glasses render the sign. As shown in Figure 15, we compare three solutions: streaming generated video at 480P, streaming full skeleton sequences, and our phoneme streaming. The translation on mobile takes 122ms on average, and phoneme streaming only takes 206ms to render the animation. In contrast, video-based and skeleton-based animations incur 22× and 16× latency, respectively. Our phoneme streaming thus ensures low-latency sign production despite using Bluetooth.

Model Complexity & System Overhead. The deep learning models for sign recognition use millions of parameters in their neural networks [33, 36]. We use traditional AI/ML models with domain knowledge with only ~4,000 parameters to be trained in our two-way, ASL-English communications. It significantly prunes the search space with ASL domain knowledge and offers a lightweight solution.

We further show the applicability of our solution on mobile phones with different hardware capabilities. In addition to OnePlus 10 Pro 5G (~550 USD), we run Sign-to-911 on Mate 20 (<300 USD) that runs Android 10 with a Kirin 980 processor, 6GB RAM, and 128GB storage. Results show that Sign-to-911 works with both high-end and mid-range smartphones. The average translation latency is 650ms, still a 14.9× reduction from Mobile-SG+.

We measure the system overhead by recording the power, CPU, and memory consumption over time during the operation of the entire setup on each device. Our system's power consumption is comparable to that of other camera applications or media players, less than 1.89W on glasses and less than 2.75W on smartphones. The mobile application uses less than 35% of CPU and takes up 350MB of memory, while the glass application uses less than 50% of CPU processing and takes up 65MB of memory. These results suggest that our application is efficient and should not significantly drain device resources, making it compatible with a wide range of commodity devices.

7.4 User Study

To evaluate the performance of our system in real-world scenarios, we conduct a user study. For emergency calls in D3, we ask 12 participating signers to rate the quality of experience (QoE), in terms of accessibility, usability, and overall experience on a 5-point scale: Excellent (5), Good (4), Fair (3), Poor (2), and Very Poor (1). We further consider the diversity of the signers' background. We conducted a survey among users aged 20 to 80, which included the deaf, hard-of-hearing individuals, and students learning ASL. The accessibility reflects how readily the system can be learned by new users and activated upon emergencies. The usability reflects correctness, liveliness, and human likeness; it is well accepted in translation assessment [60]. Signers further rate their overall experiences. They also use and rate two other emergency call solutions: text-based and video relay services (VRS). As shown in Table 5, our solution achieves the best QoE on both accessibility (4.2) and usability (4.3). The overall experience is 4.6, which indicates improved user experience compared with the text-based scheme (2.8) and VRS (3.6). Through interactions with users, we discovered that opinions about text-based communication significantly vary, mainly because many deaf individuals are not yet familiar with the new Text-to-911 service [18]. The user feedback highlights the importance of having a single-click solution for deaf people to make emergency calls. It confirms the demand for accessible emergency call services by the deaf community.

8 DISCUSSION

Iconic Classifiers. Our current solution does not fully account for signs that modify the movement path to communicate different semantics. This usually applies to a limited

number of iconic signs [67], which denote the visual features (e.g., shape and size) of its referent. For example, the path of FIRE can be modified in its production to convey various levels of fire. Since the set of commonly used classifiers is limited and variations mostly arise from path movement [68], our model, with further diversification of training samples, holds promises to handle such sign variants.

Next Step. The cultural and linguistic diversity of the U.S. Deaf community results in varied sign languages, including Signed Exact English and Black ASL, each of which possesses unique vocabularies and grammatical rules. These ASL variants will be incorporated in our next release. We also plan to include more use cases, e.g., daily communication, traveling, and content creation. This will involve integrating additional language models, e.g., ChatGPT and personalization strategies, to improve quality across various contexts. We will collaborate with smart glass vendors to improve the AR glass hardware, e.g., camera's FoV and voice interface.

9 RELATED WORK

Sign Language Translation. Sign language recognition has been an active research topic in recent years. The existing solutions are either vision-based (I3D [36], SAM-SLR [33], and DeepASL [23]) or sensor-based (using gloves [2], smart watches [30], earphones [34], EMG sensors [75], etc.). These solutions, regardless of how to extract sign features, apply deep-learning-based models (e.g., RNN and its variants) on temporal data for sign recognition. Given the model complexity, the trained model could only cover a vocabulary set size of around 100 signs. In contrast, our solution departs from the deep learning based approach. We use simpler, traditional AI/ML models, while exploiting rich ASL domain knowledge. We thus recognize about 550 signs. Our solution runs on commodity smartphones for recognition without cloud/edge support. We further reduce translation latency by an order of magnitude. Recent research [55] proposes a nice application of the Hamburg Notation System (HNS) [27] to teach sign languages. However, HNS is language-agnostic and uses a complex combinational approach of parameters to represent beyond ASL. Its increased complexity is not suitable for our mobile setting. We thus use a variant of the Stokoe system [63] with 40 base handshapes only. Moreover, [55] did an excellent job of parameter extraction. We show more work is needed in both sign recognition and translation. For recognition, we did not use deterministic representations but probabilistic ones from parameters.

Moreover, existing solutions cannot handle sentence-level translations for ASL well. They either offer no sentence-level translation at all [33, 36], or rely on complex temporal model training on large sentence corpora [2, 23, 30, 34, 75]. In contrast, we explicitly consider ASL grammar and embed such domain knowledge into our sentence-level translation models. In summary, we pursue an explainable AI model approach, thus departing from the blackbox deep learning schemes. Our models are simpler and run on smartphones.

Sign Language Production. For sign language production, current apps, such as HandTalk [26] and Sign Language Translator [57], cannot often produce the correct grammar order. They also cover limited signs, rely on clouds/edges for sign production [10, 69], or require heavy GPU processing for rendering and generation [53, 61, 62]. In contrast, we devise a grammar model using ASL syntax knowledge, thus providing both accurate and high-fidelity ASL reproduction. Our solution runs on mobile devices, and renders signs and ASL sentences on AR glasses.

Assistive AR/VR. AR/VR technologies have been used to develop assistive applications for people with disabilities [21]. For instance, [76] and [11] enable navigation for people with visual impairments, [70] leverages AR for social and emotional learning, etc. We report the first system using assistive AR glasses to support ASL communications. Our software-based solution on commodity AR glasses makes it readily accessible to ASL users.

10 CONCLUSION

Providing sign language support is vital to people with hearing disabilities in modern society. They need to interact with other people, and use sign language such as ASL as their natural and primary language, particularly in emergencies. Unfortunately, most current solutions are deemed inconvenient or inaccessible. In this work, we describe Sign-to-911, which exploits traditional AI/ML models but incorporates ASL linguistic domain knowledge. As a result, we simplify model complexity but retain high accuracy and speed in translations. Our evaluation with real signers has confirmed the effectiveness of our solution.

We believe the techniques are more widely applicable to other common usage scenarios for ASL users. We select the 911 call service since it offers a system stress test for our approach. It not only requires high recognition accuracy but also demands low latency and bidirectional translations. All such requirements must be met on mobile and wearable devices without assuming cloud/edge support.

ACKNOWLEDGMENT

We are deeply grateful to participants from GLAD Inc., NCOD at CSUN, UCLA HandsOn Club, and the broader Los Angeles Deaf community for their invaluable contributions and insights. Our warm thanks go to all survey respondents for their indispensable feedback. We express our sincere gratitude to the anonymous shepherd and reviewers for their insightful comments and suggestions. This work is partly supported by NSF CNS-2008026.

REFERENCES

[1] Accessibility service. https://developer.android.com/reference/android/accessibilityservice/AccessibilityService. Accessed: 2022-11-29.

[2] Ahmed, M. A., Zaidan, B. B., Zaidan, A. A., Salih, M. M., and Lakulu, M. M. b. A review on systems-based sensory gloves for sign language recognition state of the art between 2007 and 2017. *Sensors 18*, 7 (2018), 2208.

[3] Andersson, C., Nilsson, I., Olsson, P., Slot, G., Sörensen, J., Liechty, D., Hunter, R., Kambhatla, S., Adermann, S., M., C., Godia-Caner, P., Mettälä, R., and Box, M. Rfcomm with ts 07.10, 2001.

[4] Android camera api. https://developer.android.com/guide/topics/media/camera, Mar. 2023.

[5] Android (go edition). https://www.android.com/versions/go-edition/, Mar. 2023.

[6] Android mediacodec. https://developer.android.com/reference/android/media/MediaCodec, Mar. 2023.

[7] Android motion sensors. https://developer.android.com/guide/topics/sensors/sensors_motion, Mar. 2023.

[8] Android texttospeech. https://developer.android.com/reference/android/speech/tts/TextToSpeech, Mar. 2023.

[9] Andronix app. https://andronix.app/. Accessed: 2023-03-09.

[10] Asl translator. https://apps.apple.com/us/app/asl-translator/id421784745?correlationId=fc9f5193-5430-4cf1-8249-0b7052ee005c, Mar. 2023.

[11] Bandukda, M., and Holloway, C. Audio ar to support nature connectedness in people with visual disabilities. In *Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers* (2020), pp. 204–207.

[12] Bazarevsky, V., Grishchenko, I., Raveendran, K., Zhu, T., Zhang, F., and Grundmann, M. Blazepose: On-device real-time body pose tracking. *arXiv preprint arXiv:2006.10204* (2020).

[13] Bird, S., Klein, E., and Loper, E. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.", 2009.

[14] Cephei, A. Vosk offline speech recognition api. https://alphacephei.com/vosk/. Accessed: 2022-11-29.

[15] CHIRICO, M. Emergency - 911 calls. https://www.kaggle.com/datasets/mchirico/montcoalert. Accessed: 2022-11-29.

[16] Commission, F. C. Real-time text (rtt). https://www.fcc.gov/real-time-text, 2023. Accessed: 2023-07-30.

[17] Commission, F. C. Telecommunications relay service (trs). https://www.fcc.gov/consumers/guides/telecommunications-relay-service-trs, 2023. Accessed: 2023-07-30.

[18] Commission, F. C. Text to 911: What you need to know. https://www.fcc.gov/consumers/guides/what-you-need-know-about-text-911, 2023. Accessed: 2023-07-30.

[19] Coulter, G. R. *Current Issues in ASL Phonology: Phonetics and Phonology, Vol. 3*, vol. 3. Academic Press, 2014.

[20] Department, E. P. 9-1-1 call scripts. https://www.eugene-or.gov/2892/9-1-1-Call-Scripts. Accessed: 2022-11-20.

[21] Dick, E. Current and potential uses of ar/vr for equity and inclusion. Tech. rep., Information Technology and Innovation Foundation, 2021.

[22] ElKoura, G., and Singh, K. Handrix: animating the human hand. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2003), pp. 110–119.

[23] Fang, B., Co, J., and Zhang, M. Deepasl: Enabling ubiquitous and non-intrusive word and sentence-level sign language translation. In *Proceedings of the 15th ACM conference on embedded network sensor systems* (2017), pp. 1–13.

[24] Friedman, L. A. *Phonology of a soundless language: phonological structure of the American sign language.* University of California, Berkeley, 1976.

[25] Google. Filament. https://github.com/google/filament, 11 2022.

[26] Hand talk: Your website accessible in asl. https://www.handtalk.me/en/, Jan 2023.

[27] Hanke, T. Hamnosys-representing sign language data in language resources and language processing contexts. In *LREC* (2004), vol. 4, pp. 1–6.

[28] Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H. *The elements of statistical learning: data mining, inference, and prediction*, vol. 2. Springer, 2009.

[29] Homonyms in sign language. https://www.handspeak.com/learn/209, Mar. 2023.

[30] Hou, J., Li, X.-Y., Zhu, P., Wang, Z., Wang, Y., Qian, J., and Yang, P. Signspeaker: A real-time, high-precision smartwatch-based sign language translator. In *The 25th Annual International Conference on Mobile Computing and Networking* (2019), pp. 1–15.

[31] Inmo air. https://vr-compare.com/headset/inmoair, Mar. 2023.

[32] Insta 360 go 2. https://www.insta360.com/cn/product/insta360-go2/, Mar. 2023.

[33] Jiang, S., Sun, B., Wang, L., Bai, Y., Li, K., and Fu, Y. Skeleton aware multi-modal sign language recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2021), pp. 3413–3423.

[34] Jin, Y., Gao, Y., Zhu, Y., Wang, W., Li, J., Choi, S., Li, Z., Chauhan, J., Dey, A. K., and Jin, Z. Sonicasl: An acoustic-based sign language gesture recognizer using earphones. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 5*, 2 (2021), 1–30.

[35] Koehn, P. *Statistical machine translation.* Cambridge University Press, 2009.

[36] Li, D., Rodriguez, C., Yu, X., and Li, H. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision* (2020), pp. 1459–1469.

[37] Liddell, S. K. *Grammar, gesture, and meaning in American Sign Language.* Cambridge University Press, 2003.

[38] Liddell, S. K., and Johnson, R. E. American sign language: The phonological base. *Sign language studies 64*, 1 (1989), 195–277.

[39] Macháček, D., Žilinec, M., and Bojar, O. Lost in interpreting: Speech translation from source or interpreter? *arXiv preprint arXiv:2106.09343* (2021).

[40] MacKenzie, C. L., and Iberall, T. *The grasping hand.* Elsevier, 1994.

[41] Manifest.permission. https://developer.android.com/reference/android/Manifest.permission. Accessed: 2022-11-29.

[42] Mediapipe framework on android. https://developers.google.com/mediapipe/framework/getting_started/android, Mar. 2023.

[43] Mitchell, R. E., Young, T. A., Bachelda, B., and Karchmer, M. A. How many people use asl in the united states? why estimates need updating. *Sign Language Studies 6*, 3 (2006), 306–335.

[44] News, C. Captioned smart glasses let deaf people see, rewind conversations. https://www.cbsnews.com/miami/news/captioned-smart-glasses-let-deaf-people-see-rewind-conversations/, 2022. Accessed: 2023-07-30.

[45] on Deafness, N. I., and Disorders, O. C. Quick statistics about hearing. https://www.nidcd.nih.gov/health/statistics/quick-statistics-hearing, Mar. 2023.

[46] OpenAI. 71 pct police involved shooting 911 transcript 4/4/2018. https://www.nyc.gov/assets/nypd/downloads/pdf/public_information/911-transcripts-police-involved-shooting-040418.pdf, 2020. Accessed: 2023-02-25.

[47] OpenAI. Transcript of 911 call placed by jason ravnsborg on saturday, september 12, 2020. https://dps.sd.gov/application/files/7216/0260/

1522/911-call-transcribed.pdf, 2020. Accessed: 2023-02-25.

[48] OpenAI. chatgpt. https://chat.openai.com/chat, 2022. Accessed: 2023-02-29.

[49] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research 12* (2011), 2825–2830.

[50] Pennec, X. *Computing the mean of geometric features application to the mean rotation.* PhD thesis, INRIA, 1998.

[51] PlasticsToday. Ar glasses transcribe and display spoken language for hearing impaired. https://www.plasticstoday.com/medical/ar-glasses-transcribe-and-display-spoken-language-hearing-impaired. Accessed: 2023-06-20.

[52] Salvador, S., and Chan, P. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis 11*, 5 (2007), 561–580.

[53] Saunders, B., Camgoz, N. C., and Bowden, R. Progressive transformers for end-to-end sign language production. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16* (2020), Springer, pp. 687–705.

[54] Sehyr, Z. S., Caselli, N., Cohen-Goldberg, A. M., and Emmorey, K. The asl-lex 2.0 project: A database of lexical and phonological properties for 2,723 signs in american sign language. *The Journal of Deaf Studies and Deaf Education 26*, 2 (2021), 263–277.

[55] Shao, Q., Sniffen, A., Blanchet, J., Hillis, M. E., Shi, X., Haris, T. K., Liu, J., Lamberton, J., Malzkuhn, M., Quandt, L. C., et al. Teaching american sign language in mixed reality. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 4*, 4 (2020), 1–27.

[56] Shi, B., Del Rio, A. M., Keane, J., Michaux, J., Brentari, D., Shakhnarovich, G., and Livescu, K. American sign language fingerspelling recognition in the wild. In *2018 IEEE Spoken Language Technology Workshop (SLT)* (2018), IEEE, pp. 145–152.

[57] Sign language translator. https://apps.apple.com/us/app/sign-language-translator/id1458992650, Mar. 2023.

[58] Signing savvy. https://www.signingsavvy.com/, Mar. 2023.

[59] Sondej, F. Autocorrect. https://github.com/filyp/autocorrect, Mar. 2023. Version: 2.0.4, Accessed: 2023-03-01.

[60] Specia, L., Raj, D., and Turchi, M. Machine translation evaluation versus quality estimation. *Machine translation 24* (2010), 39–50.

[61] Stoll, S., Camgoz, N. C., Hadfield, S., and Bowden, R. Text2sign: towards sign language production using neural machine translation and generative adversarial networks. *International Journal of Computer Vision 128*, 4 (2020), 891–908.

[62] Stoll, S., Hadfield, S., and Bowden, R. Signsynth: Data-driven sign language video generation. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16* (2020), Springer, pp. 353–370.

[63] Tennant, R. A., Gluszak, M., and Brown, M. G. *The American sign language handshape dictionary.* Gallaudet University Press, 1998.

[64] The asl interpreter shortage and its impact on accessibility in college settings. https://nationaldeafcenter.org/news-items/the-asl-interpreter-shortage-and-its-impact-on-accessibility-in-college-settings/, Dec. 2022.

[65] Transmission rate vs. bandwidth in bluetooth technology. https://resources.pcb.cadence.com/blog/2022-transmission-rate-vs-bandwidth-in-bluetooth-technology, Mar. 2023.

[66] Unmarked and marked handshapes in sign language. https://www.handspeak.com/learn/index.php?id=439. Accessed: 2023-02-29.

[67] Valli, C., and Lucas, C. *Linguistics of American sign language: An introduction.* Gallaudet University Press, 2000.

[68] Vicars, B. Classifiers. https://www.lifeprint.com/asl101/pages-signs/classifiers/classifiers-main.htm, 2023. Accessed: 2023-07-15.

[69] Vivo sign language translator. https://assist.vivo.com/detail/translate, Mar. 2023.

[70] Voiss. https://www.projectvoiss.org/, Mar. 2023.

[71] Vuzix blade 2. https://vr-compare.com/headset/vuzixblade2, Mar. 2023.

[72] Wikipedia contributors. Mixamo — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Mixamo&oldid=1106970206, 2022. Accessed: 7-December-2022.

[73] Wikipedia contributors. Signing space — Wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Signing_space, 2023. Accessed: 2023-07-30.

[74] Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C.-L., and Grundmann, M. Mediapipe hands: On-device real-time hand tracking. *arXiv preprint arXiv:2006.10214* (2020).

[75] Zhang, Q., Jing, J., Wang, D., and Zhao, R. Wearsign: Pushing the limit of sign language translation using inertial and emg wearables. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 6*, 1 (2022), 1–27.

[76] Zhao, Y., Bennett, C. L., Benko, H., Cutrell, E., Holz, C., Morris, M. R., and Sinclair, M. Enabling people with visual impairments to navigate virtual reality with a haptic and auditory cane simulation. In *Proceedings of the 2018 CHI conference on human factors in computing systems* (2018), pp. 1–14.

[77] Zhou, Z., Chen, K., Li, X., Zhang, S., Wu, Y., Zhou, Y., Meng, K., Sun, C., He, Q., Fan, W., et al. Sign-to-speech translation using machine-learning-assisted stretchable sensor arrays. *Nature Electronics 3*, 9 (2020), 571–578.