GASLA: Enhancing the Applicability of Sign Language Translation

Jiao Li, Yang Liu, Weitao Xu, Zhenjiang Li
Department of Computer Science, City University of Hong Kong, China

***Abstract*—This paper studies an important yet overlooked applicability issue in existing American sign language (ASL) translation systems. With excessive sensing data collected for each ASL word already, current designs treat every to-be-recognized sentence as new and collect their sensing data from scratch, while the amounts of sentences and the data samples per sentence are large usually. It takes a long time to complete the data collection for each single user, e.g., hours to a half day, which brings non-trivial burden to the end users inevitably and prevents the broader adoption of the ASL systems in practice. In this paper, we figure out the reason causing this issue. We present GASLA atop the wearable sensors to instrument our design. With GASLA, the sentence-level sensing data can be generated from the word-level data automatically, which can be then applied to train ASL systems. Moreover, GASLA has a clear interface to be integrated to existing ASL systems for overhead reduction directly. With this ability, sign language translation could become highly lightweight in both initial setup and future new-sentence addition. Compared with around 10 per-sentence data samples in current systems, GASLA requires 2–3 samples to achieve a similar performance.**

I. Introduction

To improve the life convenience and achieve better conversations for deaf people, a surge of American Sign Language (ASL) translation systems have been proposed, *e.g.*, using the motion data from wearable devices [34], [12], the skeleton data from cameras [10], the wireless signals from Wi-Fi or RFID devices [17], etc. These systems can "understand" the sign language performed by a deaf user and play its semantic content through a speaker (or show it on the phone), so that the deaf user can communicate with other people smoothly.

However, in this paper, we observe one important yet overlooked *applicability* issue in existing ASL system designs. To set up one such system, the sensing data samples are collected for each basic "word" first to build a word library. Next, for the more important ASL "sentences" that the deaf user wants the system to recognize and translate, we find current systems [10], [34], [12], [21] treat every sentence as new and collect the sensing data for each sentence from scratch to build the sentence library, given the fact that all the words (that can compose these sentences) are collected in the word library already. This causes the following applicability issues:

- *Expensive setup overhead.* To satisfy the daily communication's needs, the numbers of both 1) the sentences to be translated, *e.g.*, tens to hundreds of sentences, and 2) the amount of per-sentence sensing samples to collect, *e.g.*, at least 10 samples per sentence [10], [12], are large usually. It takes hours or even a half day for the data collection. This incurs non-trivial setup overhead, which could make novice or impatient users quit in the first place.
- *High maintenance cost.* Due to the same reason above, whenever a user wants to upgrade her system to add more new sentences, the same amount of effort needs to be paid to collect the sensing data for each new sentence. This is a life-long limitation in current designs, which calls for the ability to setup and update the system efficiently.
- *Unreliable system performance*. Enforcing high setup and maintenance overhead to the end users certainly lead to a higher chance to lower the user's patience during the data collection. This could cause a *continuous* performance drop when the low-quality sensing data are collected to train the system by the novice or impatient users.

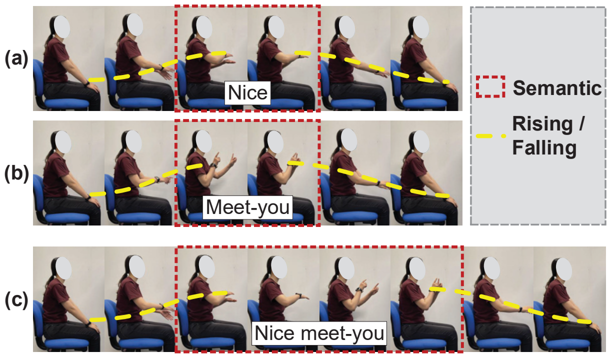

Fig. 1: Illustration of data collection for a wearable-based ASL system. (a) "Nice" and (b) "Meet-you" are two ASL words to be collected. (c) "Nice Meet-you" is an ASL sentence to be collected, which is "nice to meet you" in English. Each sensing data sample contains *rising*, *semantic* and *falling* three parts.

To overcome these issues, which we believe are one important reason preventing a wider adoption of ASL systems in practice, we need to understand why the sentence-level sensing data cannot be generated from the word-level ones in the prior systems first. When the sensing data samples are collected for each word, it is non-trivial to collect the data that correspond to the semantic meaning of this word exactly, because the sensing data for each word needs to be collected multiple times (to train the system) and a frequent device on/off switching is quite time consuming. Therefore, in current systems, a user starts and ends each word with respect to some pre-defined position usually, *e.g.*, on thigh in Fig. 1(a-b), so that sensing data can be collected continuously and later segmented easily. Of course, the negative impact is that each segmented word-level data sample contains not only the semantic part, which includes one *beginning* part and one *ending* part as well, *e.g.*, the rising and falling two parts in Fig. 1(a-b), respectively. However, when these two words are performed consecutively in one sentence, such extra parts from individual words do not exist in the sentence's semantic part, as Fig. 1(c) shows. This is why the word-level data cannot be concatenated to construct a sentence-level data sample directly.

With above understanding, in this paper, we aim to address this issue based on the following observation. For any two to-be-concatenated words, the falling part of the former word (*e.g.*, "nice") and the rising part of the latter word (*e.g.*, "meet-you") experience two approximately *opposite* moving traces. If we align the ending spot of "nice" and the starting spot of "meet-you", and then rotate one of the moving trace in the air to analyze the similarity of these two traces, we find that the falling part of "nice" and the rising part of "meet-you" share many similarities initially. After certain point, two traces start to diverge dramatically — their individual semantic parts start. Hence, we can view this point as a *separation point*, based on which we can exclude the sensing data in the falling and rising parts for the first and second words (Fig. 2(a-b)), respectively. The semantic parts of these two words can be then concatenated to form a two-word sentence (Fig. 2(c)), which are similar to the corresponding data when the user performs this sentence directly (Fig. 2(d)). This process can be repeated to concatenate more words to form a longer sentence and we can use it in the system training.

To harness this opportunity, we propose *GASLA* (Generative ASL translAtor) and integrate it with the wearable sensors to instrument our design. For any pair of words, their semantic parts can start at different positions with respect to the user's body, leading to different separation points. We thus propose a pipeline to achieve an effective signal processing and analyze the similarity of their traces to identify the separation point automatically. Moreover, we also make two practical considers in *GASLA*, including 1) handling the stiff changes of the sensing data at the separation point, which could lead to strong yet undesired features to dominate and undermine the ASL translation; and 2) abstracting a clear interface for *GASLA* to be compatible to the modules in the existing ASL systems for reducing their overhead.

We develop a prototype of *GASLA* with LG smart watch, integrate our design to augment the state-of-the-art system SignSpeaker [12], and experiment on six volunteers with the ASL library of 41 sentences generated from 69 words. Extensive results show that SignSpeaker needs at least 10 samples for each sentence in the system training to achieve a high accuracy of 95%. Augmented by *GASLA*, with the generated sentence-level trace as a good base, each user only needs to include two to three native sentence-level samples to achieve a comparable performance, *e.g.*, 93%. In summary, this paper has made the following contributions:

- We identify an applicability issue commonly in prior ASL systems, which can cause the high setup and maintenance costs, as well as continuous performance loss potentially if errors are made in the labor-intensive data collection.
- We propose effective techniques to address this issue and realize them in *GASLA*. Moreover, since *GASLA* is positioned to enhance the applicability of ASL translation, instead of reinventing the entire system stack, we thus provide a clear interface for our techniques to be compatible to the existing ASL system.
- We develop a prototype of *GASLA* and conduct extensive experiments. The results show that *GASLA* can achieve comparable performance with the state-of-the-art design but reduce the overhead remarkably.[1]

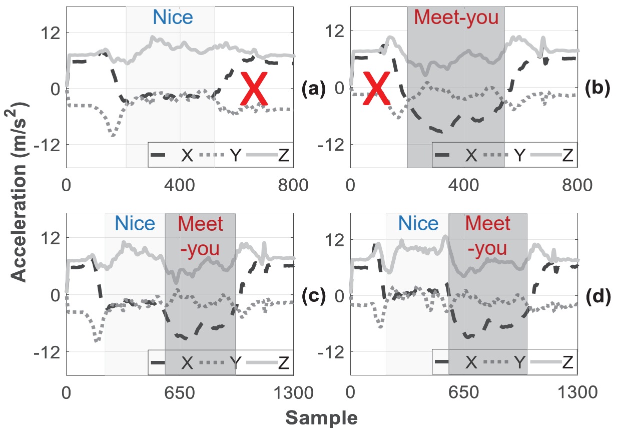

Fig. 2: Accelerometer data collected in wearable-based ASL systems. For two ASL words (a) "nice" and (b) "meet-you", the crossed areas represent the to-be-excluded parts based on the separation point. Afterwards, the sensing data for (c) the generated sentence "nice meet-you" is similar to that of (d) the sentence performed by the user directly.

II. Background and Overview

In this section, we introduce the background of the sign language translation and the overview of our *GASLA* design.

A. Sign Language Translation

1) Sign language users. There are a large population of deaf people globally, *e.g.*, approximately 48 millions in U.S. [12], 11 millions in U.K. [1], etc. Sign language is their major language. It is composed of a set of pre-defined hand gestures to represent the basic words, which can be further connected to form different sentences. Two sign-language "speakers" can communicate smoothly, while they usually have difficulties to communicate with the people without the impaired hearing or speech issues, which leads to tremendous inconvenience in their daily life, *e.g.*, shopping, commuting, seeing a doctor, etc. More severely, many of them may not be able to obtain normal educations and employments, which could cause them social

[1]The generated training data can train the ASL system directly. However, according to our experiments in Section IV, we suggest the users to collect two to three data samples from their direct performing of each sentence and mix them with the generated ones in the training data set. By doing so, the performance can be improved significantly. In the future, we plan to improve our design to further avoid the collection of such two to three data samples.

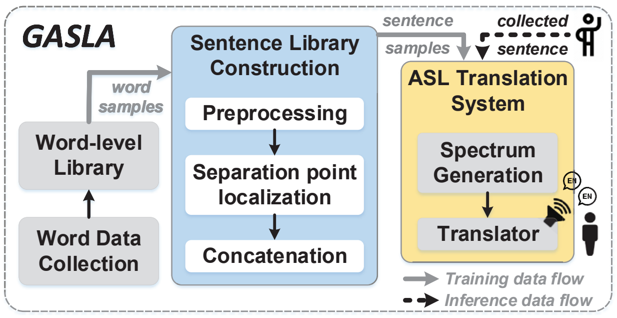

Fig. 3: Overview of the *GASLA* design.

isolation and loneliness [2]. As a result, the sign language translation is essential and necessary for them.

2) Wearable-based solutions. To benefit the deaf people, a variety of sign language translation systems have been proposed (reviewed in Section V), while we find they share one common issue to limit their applicability in practice. In this paper, we adopt the *wearable-based* solutions to instrument our design, due to their several advantages, including the wide availability of wearable devices (*e.g.*, smart watch, wrist band, etc.), the portable design in nature (*e.g.*, the user is not required to stand in front of a camera with the computer vision designs), and the lightweight computation (*e.g.*, the motion data is used usually and their computations can be afforded by the phone).

The principle of our solution is general, which can be turned to fit other types of designs potentially in the future.

B. System Overview

Fig. 3 overviews our design. Because *GASLA* is positioned to improve the applicability of current ASL systems instead of reinventing the entire system stack, we design it with a clear interface, so that the sentence-level data samples generated by *GASLA* can be used to train the underlying ASL system.

With the word-level data samples collected (*e.g.*, accelerometer and gyroscope data), a user provides a list of sentences to be recognized by the ASL system. Then, the *sentence library construction* component in *GASLA* starts to generate the data samples for these sentences using the word-level data samples from the word library. The generation goes through three steps, including the pre-processing, separation point localization and concatenation. All the sentence-level data samples generated by *GASLA* together are used to form the training data set.

To train one ASL system, the input training data samples are usually converted to the frequency domain first and their frequency spectrum is then applied to train the translator. Later, when the system is used after the training, it takes the sensing data from the sign language performed by the user as input (in the actual system usage, the input is the actual sensing data, not our generated ones) and recognizes its meaning, which can be played through a speaker or shown on the phone's screen, so that deaf people can communicate with other people smoothly.

III. System Design

We elaborate the *GASLA* design in this section.

A. Word-Level Data Collection

To setup an ASL system, the first step is to collect sensing data for a set of user-selected words to build a word-level library. For each word, the user performs it several times to collect multiple samples of the corresponding sensory data.[2] Using a wearable device, the sensory data are from *accelerometers* (acc) and *gyroscopes* (gyro) [12] usually. Therefore, each sensory data sample for any word x can be expressed as:

$$\mathbf{s}_x \quad = \quad < acc_x, \; gyro_x >, \qquad (1)$$

where acc_x and $gyro_x$ represent a sequence of the acc and gyro values respectively, *e.g.*, $acc_x = \{a_x(t)\}$ and $gyro_x = \{g_x(t)\}$, where t is the time index. All $\mathbf{s}_i$ together form the word-level library. Suppose that eight samples are collected for each word. If we want to concatenate words "x" and "y" to form a two-word sentence "xy", we can generate 64 ($= 8 \times 8$) different sensory data samples for this sentence, by numerating different combinations of $\mathbf{s}_x$ and $\mathbf{s}_y$ samples. For each combination, we concatenate the semantic parts of "acc_x and acc_y" as well as "$gyro_x$ and $gyro_y$". The detailed concatenation is introduced in the next subsection.

B. Sentence Library Construction

With the word-level library above, the user further decides a list of sentences to be recognized by the ASL system. With *GASLA*, we will generate the sensory data samples for each sentence to construct the sentence-level library. We note that if one sentence contains some word(s) not in the word-level library yet, the user can collect the sensory data for these missing words first, so that the sentences can be generated from the word-level library directly. As Fig. 3 depicts, the sentence generation contains three steps, including *Pre-processing*, *Separation point localization* and *Concatenation*. We introduce each of them in the following.

1) Pre-processing: As stated in Section I, each word-level sensory data sample contains *rising*, *semantic* and *falling* three parts (Fig. 1). To concatenate two words x and y, we will examine the similarity between the falling part of the former word "x" and the rising part of the latter word "y". Then we can exclude these transitional sensory data and connect their semantic parts to obtain one generated sensory data sample for the sentence "xy".

1.a) Problem. Ideally, if we had the wearable device's exact moving trajectories when the user is performing these two words, we can compare these two trajectory traces to fulfill above design idea. However, the wearable sensory data (*e.g.*, acc and gyro) do not provide the device's location information directly. Some recent studies [15], [23], [24] are able to use the motion sensor data from a single wearable device to recover the moving trace of a user's arm, but the overhead to integrate them is large. More importantly, the accuracy of these existing designs are moderate merely, *e.g.*, about 10 cm tracking errors, which are not precise enough for this concatenation design.

[2] In addition to generate sentences, these words can also be used to train a dedicated neural network to recognize each single word performed by the user [34], [10], which is orthogonal to *GASLA* and thus omitted in this paper.

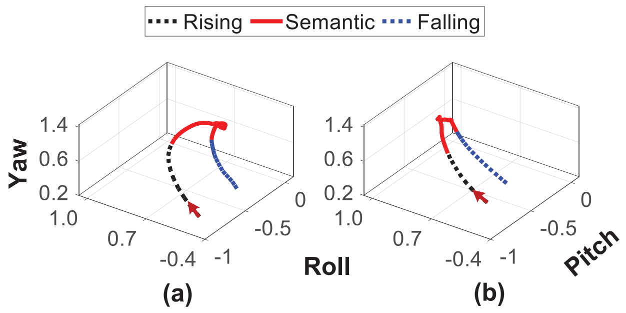

Fig. 4: Orientation $\mathbf{ori}_x$ traces of two ASL words (a) "Nice" and (b) "Meet-you" respectively, which are derived from the acceleration and the angular velocity of the wearable device.

1.b) Proposed strategy. To collect the sensory data for each word, the user usually starts and ends each word with respect to some pre-defined position usually, *e.g.*, on thigh as shown in Fig. 1. We find that the **orientation** of the device (on the wrist) could serve as a good approximation to reflect how the user's arm is moving. Since this information is used to locate the *separation point* merely (in step two), such an approximation of the device's actual moving trace suffices for *GASLA*.

Orientation [37] is derived from device's acceleration (acc) and the angular velocity [3]. It is measured by *Euler angle* θ and can be obtained by using the wearable OS's API directly.[3] Therefore, for each sensory data sample $\mathbf{s}_x$ in Eqn. (1), we can further obtain its corresponding orientation trace:

$$\mathbf{ori}_x \quad = \quad \{\theta_x(t)\}_{t=1}^T = \{< R_x(t),\ Y_x(t),\ P_x(t) >\}_{t=1}^T,$$

where T is the number of time steps and $\theta_x(t)$ is the Euler angle at time t, which can be presented by roll (R), yaw (Y) and pitch (P) three directions. Fig. 4(a) and (b) depict $\mathbf{ori}_x$ for the words "nice" and "meet-you", respectively.

We note that the orientation $\mathbf{ori}_x$ is used to locate the separation point merely, based on which we can remove the sensory data (acc and gyro) for the falling part of word x and the rising part of word y, respectively. The orientation itself is not used by the ASL system. Before we conduct the separation point search, we perform two pre-processings for $\mathbf{ori}_x$ first.

Signal cropping. Signal cropping removes the *idle* and *transitional* portions in the sensory data sample of each word, since we find that keeping them will decrease the efficiency of the separation point search and the final sentence recognition performance. We use $\mathbf{ori}_x$ to find the two cropping boundaries, based on which we crop for both acc and gyro data.

For each axis of the Euler angle $A_x(t)$, where A is R, Y or P, the cropping boundaries are the *first* and *last* local minimum points of $A_x(t)$, *e.g.*, Fig. 5 marks them for $P_x(t)$. However, due to the inevitable noises and jitters, it is non-trivial to select a proper window size to conduct the local minimum search, *e.g.*, at each time t, the average value in a window before and after t is computed to determine whether t is a local minimum point. Hence, we propose to conduct the first *derivative* for each $A_x(t)$ and all the local minimum points

[3]Euler angle indicates the attitude of the wearable device at each timestamp in a reference coordinate system [3].

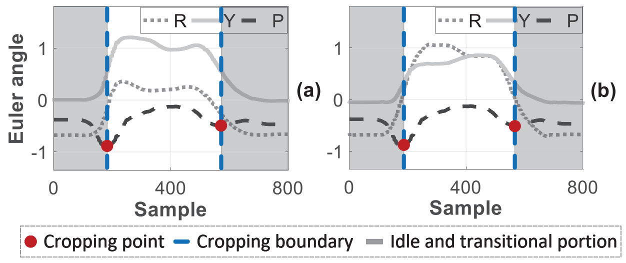

Fig. 5: Illustration of signal cropping on the orientation traces for (a) "Nice" and (b) "Meet-you" these two ASL words.

correspond to the *zero-crossing* points, meanwhile whose left part is *negative* and right part is *positive*, in the first-derivative trace (Fig. 6). We can then select the first and last such points without using any pre-defined window for averaging, which is more reliable. For each word, we conduct this operation for roll, yaw and pitch three axes individually first. Then,

- for the rising part, if three axes lead to three different cropping points, we select the one with the largest time index, *i.e.*, the most right-hand-side one in Fig. 5.
- for the falling part, we select the cropping point with the minimum time index, *i.e.*, the most left-hand-side one.

Re-sampling. Due to the unstable sampling rate of wearable device [18], we further re-sample each $\mathbf{s}_x$ (after above signal cropping) and normalize their length in time to be the same.

2) Separation Point Search: For two to-be-connected words "x" and "y", the falling part of "x" and the rising part of "y" experience two approximately *opposite* moving traces within a certain range. We denote the *farthest* point of this range as their **separation point**. Beyond this point, the semantic part of each word will start gradually.

According to this rationale, the key idea of the separation point search design is to **align** the ending spot on the falling part of the former word "x" with the starting spot on the rising part of word "y". Then, we fix one trace (*e.g.*, $\mathbf{ori}_x$) and gradually rotate another one (*e.g.*, $\mathbf{ori}_y$) to examine the similarity of their (highly) overlapped portion. After certain point, when two traces start to diverge dramatically, we view this point as the separation point. As a result, the separation point search includes the following steps:

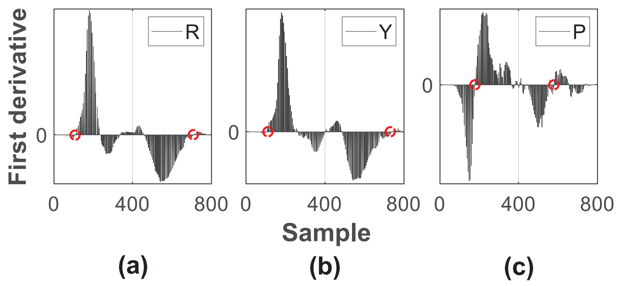

Fig. 6: First derivative of $A_x(t)$, where A is for (a) R, (b) Y and P. The cropping points are highlighted. They are zero-crossing points. Meanwhile, their left and right parts are negative (corresponding to Euler angle's decreasing) and positive (corresponding to Euler angle's increasing), respectively.

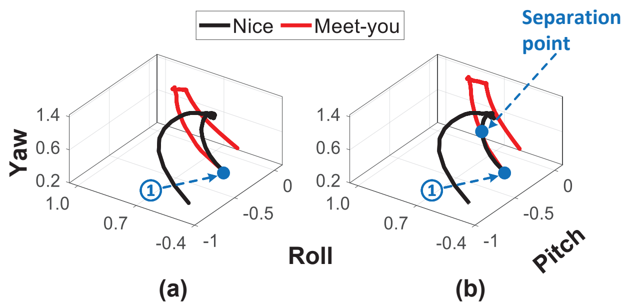

Fig. 7: (a) Two orientation traces are aligned at spot ①; and (b) the separation point found for these two ASL words.

Step 1) Alignment: Because the orientation trace represents how the device is rotated with respect to the previous time step, two opposite orientation traces cannot be compared directly to analyze their similarity, while we can reverse one of them (*e.g.*, $\mathbf{ori}_x$) along the time, denoted as $\mathbf{ori}'_x$, *i.e.*,

$$\mathbf{ori}'_x \quad = \quad \{\theta_x(T-t+1)\}_{t=1}^{T} = \{\theta'_x(t)\}_{t=1}^{T}, \qquad (2)$$

and use this reversed $\mathbf{ori}'_x$ to get aligned and compared with $\mathbf{ori}_y$. In Fig. 7, spot ① is the starting spot of $\mathbf{ori}_{meet-you}$, which has been aligned with the ending spot of $\mathbf{ori}'_{nice}$.

Step 2) Search: To find the the location of the separation point, search is performed by in an iterative manner. Initially, we compare two orientation traces within a conservatively small window w (*e.g.*, w=200), and then we gradually increase the window size until the separation point is observed. In the first (initial) iteration, each of two orientation traces includes w Euler angles:

$$\mathbf{ori}'_x = \{\theta'_x(t)\}_{t=1}^{w}; \text{ and } \mathbf{ori}_y = \{\theta_y(t)\}_{t=1}^{w}. \qquad (3)$$

Then, we fix $\mathbf{ori}'_x$ and rotate $\mathbf{ori}_y$ (with respect to their aligned spot) to search the best overlapping. This operation is essentially to multiply a rotation matrix r_y to $\mathbf{ori}_y$, so that the distance $d(\cdot)$ between $\mathbf{ori}'_x$ and the rotated $r_y \cdot \mathbf{ori}_y$. In general, for each iteration, the best overlapping can be quantified by

$$\min_{\{r_y \in space\}} d(\mathbf{ori}'_x,\ r_y \cdot \mathbf{ori}_y), \qquad (4)$$

where the search space of r_y and the distance measure $d(\cdot)$ in Eqn. (4) are determined as follows.

Search space. The search space is a cone-like range, whose central axis passes through the first and last Euler angle points $\theta'_x(1)$ and $\theta'_x(w)$ from $\mathbf{ori}'_x$, as illustrated in Fig. 8. The orientation trace $\mathbf{ori}_y$ is rotated within this search space with a small discrete step along Roll, Yaw and Pitch three directions. For each rotation r_y, we compute its distance $d(\cdot)$ to $\mathbf{ori}'_x$.

Distance. Considering that the time stamps are not precisely synchronized between the two traces, we employ dynamic time warping (DTW) [4] to tolerate such inconsistency by warping $\mathbf{ori}_y$ first and then compute its distance to $\mathbf{ori}'_x$:

$$d(\cdot) \quad = \quad \frac{1}{w}\sum_{t=1}^{w} ||\ \theta'_x(t) - r_y \cdot DTW(\theta_y(t))\ ||_2. \qquad (5)$$

Algorithm 1: Sentence-level Sensory Data Generation

```
1  input: $acc_x,\ gyro_x,\ ori_x,\ acc_y,\ gyro_y,\ ori_y$;
2  output: $acc_{xy}, gyro_{xy}$;
3  $(ori_x, ori_y) = pre-process(ori_x, ori_y)$;
4  $(ori'_x, ori_y) = align(ori_x, ori_y)$;
5  $initializing\ w;\ isFnd = false$;
6  while $w \le W$ and $!isFnd$ do
7      $\tilde{r}_y = \min_{\{r_y \in space\}} d(ori'_x,\ r_y \cdot ori_y)$;
8      $\mathcal{D}(t) = \{\Delta(t)\}_{t=1}^{w}$;
9      while $\tilde{t} \in [1,\ w]$ do
10         if $\frac{1}{\tilde{t}}\sum_{l=1}^{\tilde{t}}\Delta(l) < \alpha \cdot \frac{1}{w-\tilde{t}+1}\sum_{l=\tilde{t}}^{w}\Delta(l)$ then
11             $(acc_{xy}, gyro_{xy}) =$
                 $concatenate(acc_x, gyro_x, acc_y, gyro_y)$;
12             $isFnd = true;\ break$;
13         else
14             $increase\ \tilde{t}\ by\ one$;
15     $increase\ w\ size$;
```

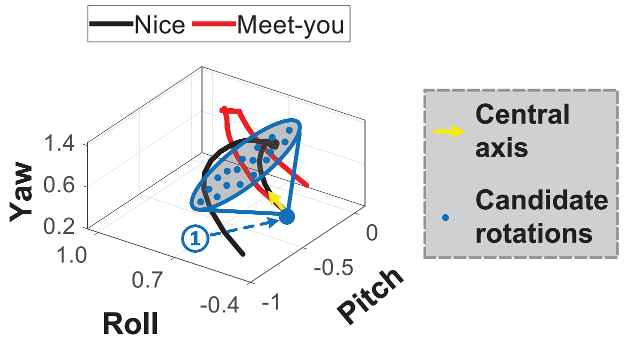

Fig. 8: Illustration of the cone-like search space.

Step 3) Checking: After the optimal rotation matrix $\tilde{r}_y$ is determined from Eqn. (4), we need to check whether the separation point is included in the current window w. In the equation of $d(\cdot)$ above, we denote $\Delta(t) = \theta'_x(t) - \tilde{r}_y \cdot DTW(\theta_y(t))$, which is the distance between two corresponding points from two orientation traces, and we store all $\Delta(t)$s in a set:

$$\mathcal{D}(t) \quad = \quad \{\Delta(t)\}_{t=1}^{w}. \qquad (6)$$

If the separation point is in current window w and we denote its time index is $\tilde{t}$, we find that before time $\tilde{t}$, the average of the $\Delta(t)$ distances is generally small (highly overlapped), while it climbs up quickly after $\tilde{t}$ (starting to diverge from each other). Such an intuition can be realized by the following mechanism:

$$\min_{\tilde{t} \in [1,w]} \tilde{t}, \qquad (7)$$

$$s.t. \quad \frac{1}{\tilde{t}}\sum_{l=1}^{\tilde{t}}\Delta(l) < \alpha \cdot \frac{1}{w-\tilde{t}+1}\sum_{l=\tilde{t}}^{w}\Delta(l), \qquad (8)$$

where α is an empirical parameter and it is set to 0.35 in our current implementation. After solving Eqns. (7) and (8),

- if the optimal $\tilde{t}$ is found between 1 and w, it implies that the separation point is in current window w and we can move to the next concatenation module.

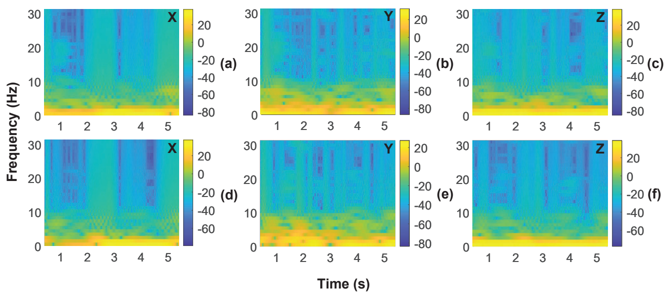
Fig. 9: (a)-(c) Spectrum of 3-axis acc for the generated sentence “nice meet-you” is similar to (d)-(f) that of 3-axis acc for the sentence performed by the user directly.

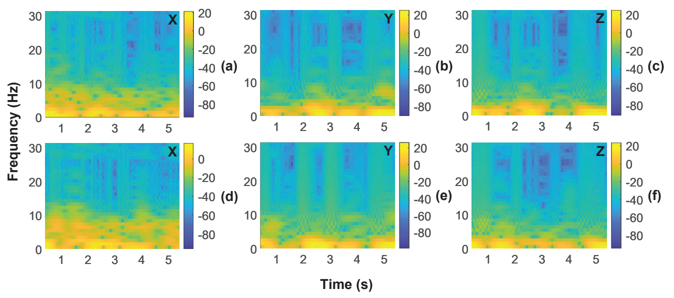
Fig. 10: (a)-(c) Spectrum of 3-axis gyro for the generated sentence “nice meet-you” is similar to (d)-(f) that of 3-axis gyro for the sentence performed by the user directly.

- if there is no $\tilde{t}$ to make constraint Eqn. (8) to hold, it implies that the separation point is not in current window w. In this case, we increase the window size and continue the search for the next iteration. In current *GASLA*, the window size is increased by 50 each time.

3) Concatenation: After the time index $\tilde{t}$ of the separation point is found, we can process the sensory data $acc = \{a(t)\}_{t=1}^{T}$ and $gyro = \{g(t)\}_{t=1}^{T}$ for words “x” and “y”:

- For word “x”: we remove all its acc and gyro data (in the falling part) from time $T - \tilde{t}$ to T. In other words, we keep $\{a_x(l)\}_{l=1}^{\tilde{t}}$ for acc and $\{g_x(l)\}_{l=1}^{\tilde{t}}$ for gyro.
- For word “y”: we remove the acc and gyro data from the beginning to time $\tilde{t}$. In other words, we keep $\{a_y(l)\}_{l=\tilde{t}}^{T}$ for acc and $\{g_y(l)\}_{l=\tilde{t}}^{T}$ for gyro.

However, if we connect $\{a_x(l)\}_{l=1}^{\tilde{t}}$ and $\{a_y(l)\}_{l=\tilde{t}}^{T}$ directly to form the acc trace for “xy” (similar for gyro), it usually leads to a sudden sensory value change at the concatenation point, which will incur strong frequency responses to dominate the frequency spectrum of the sensory data (the neural networks of ASL systems take such spectrum as input usually). To tackle this issue, we apply a small transitional window between $\{a_x(l)\}_{l=1}^{\tilde{t}}$ and $\{a_y(l)\}_{l=\tilde{t}}^{T}$ (similar for gyro), and conduct interpolations to add a set of interpolated sensory values for a smooth transition in between without sudden changes.

4) Summary: So far, we have introduced the concatenation for two words “x” and “y” to form a two-word sentence “xy”, as summarized in Algorithm 1. If we need to connect “xy” with another word “z”, we can repeat above process for “y” and “z”, so that the sensory data sample for sentence “xyz” can be generated. We can continue this process to form the sensory data sample for the longer sentences.

On the other hand, for each sentence, by using different sensor data samples from each word, we can generate different samples for the same sentence. Finally, all the generated sensor data samples for the user-selected sentences will form the sentence-level library to train the ASL system.

C. ASL Translation

Now, with the sentence-level library, we can train an ASL neural network to recognize different sentences.

1) Network input: For each sentence k, we have multiple samples of the generated sensory data and each sample includes both the acc and gyro data, *e.g.*, $\mathbf{s}_k =< acc_k,\ gyro_k >$.

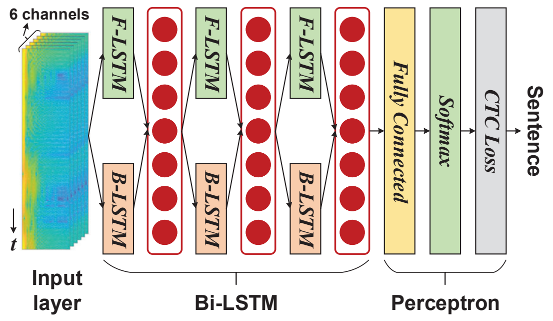

Fig. 11: Structure of the neural network for ASL translation.

Because the features of the motion sensory data for different words or sentences are mainly reflected at the frequency domain, similar as the prior methods [12], we convert acc_k and $gyro_k$ to the frequency domain and employ such frequency spectrum to train the neural network.

Fig. 9 and 10 show the spectrum of the three-axis acc_k and $gyro_k$ for sentence “nice meet-you”, respectively. In each figure, the first row is the spectrum of our generated sensory data. As a comparison, the second row shows the spectrum of the sensory data when this sentence is performed by user directly. We can see that our generated sensory data capture the main frequency characteristics of the targeted sentence, which can thus be used to train the neural network directly.

2) Neural network: As stated in Section I, *GASLA* aims to the generate the sentence-level sensory data from the word-level ones, instead of re-inventing the entire ASL system stack. Therefore, with the training data obtained from *GASLA*, we apply it for the neural network from the state-of-the-art ASL system, SignSpeaker [12], directly, which can enable *GASLA* to be an independent component to be integrated to existing ASL systems. Fig. 11 illustrates the network structure, which includes three major parts:

- *Input*: the network takes six spectrums as input from both the acc and gyro sensory data.[4] Each spectrum is a 198 by 1245 2D image in our implementation.
- *Bi-LSTM layers*: since sensory data have strong temporary correlations within each sentence, Bi-LSTM layers are used in the network design to capture such relation.

[4]SignSpeaker also takes the spectrum of linear acceleration as input, which are derived from the acc data. We thus omit it in our current implementation.

- *Connectionist Temporal Classification (CTC)*: CTC layer is employed to associate each word's label in a sentence to the corresponding part in the spectrum automatically.

IV. System Evaluation

In this section, we evaluate the performance of *GASLA*.

A. Experiment Setup

Hardware and software. We develop a prototype system of *GASLA* with the LG Watch (Android 7.1 Wear OS with Invensense MPU-6515 six-axis motion sensors), SAMSUNG Galaxy S7, and a desktop of Intel i7-8700K CPU and Nvidia GTX 2080Ti GPU to train the neural network of the ASL translation system. The neural network is developed on TensorFlow (1.15.0). After the development and training, we deploy the executable network on the smartphone.

Data collection. Since there is no public motion sensor data set from smart watch for sign language [12], we conduct a data collection according to the well-known American Sign Language guidance website [5]. We select 69 popular words in our daily lives to construct the word-level library. These word-level sensory data are further used to generate the sensory data for 41 commonly used sentences, *e.g.*, "Food enough", "I like hamburger", "Your favorite book what", used for evaluation.

We recruit six volunteer users (3 females and 3 males) to participate into the data collection and each user wears the smart watch on the right wrist. This study has obtained the university's ethical approval. Before the data collection starts, we provide a 10-minute tutorial for each user about the usage of device, the procedure of data collection, etc. We start from the word-level data collection. For each word, the video illustration from [5] is played. The user sits on a chair and rehearses until getting ready. During the actual collection, every word is performed five times continuously (without pausing the collection in the middle). As shown in Fig. 1, the default hand position before and after performing a word each time is on the user's thigh. After all the words are completed, we move on to the sentence-level data collection with a similar procedure. In total, we have collected 2460 (= $6 \times 41 \times 10$) pieces of sentence-level data, which are used to train the ASL system without *GASLA* for comparison purpose, and 2070 (= $6 \times 69 \times 5$) pieces of word-level data, which are used to generate 3936 (= $6 \times 41 \times 16$) pieces of sentence-level data for training the ASL system with *GASLA*. With the training data set, we train only one neural network for each method (stated below) and test the performance of each method on all six users. For each method, we use 80% of sentence-level data to form the training data set.

Metric. We adopt the same metric — word error rate (r_e) — as [12], which is widely used to evaluate ASL systems and defined as $r_e = \frac{D+I+S}{D+C+S}$, where I is the minimum number of word insertions, D is the minimum number of word deletions, S is the minimum number of substitutions and C is the number of correctly recognized signs, between the recognized sentence by the network and the ground truth. With r_e, we adopt the **accuracy**, *i.e.*, $(1 - r_e) \times 100\%$, in our evaluation.

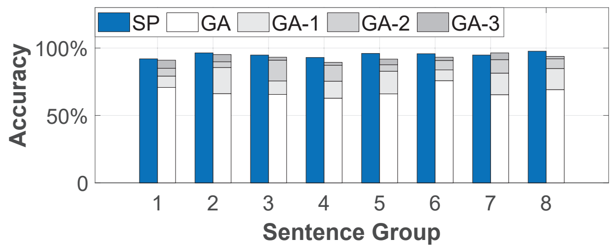

Fig. 12: Accuracy comparisons between SignSpeaker (SP) and *GASLA* (GA) under different settings by adding one to three piece(s) of native sensory data in the training.

Methods. We compare following methods in the evaluation:

- **SignSpeaker** (SP): the state-of-the-art wearable-based ASL system [12], which is trained with eight sensory data samples for each sentence when the sentence is performed by users directly (without our generated sensory data).
- ***GASLA*** (GA): our design atop SignSpeaker, in which the network is trained by using our generated sensory data.

B. Overall Performance

To facilitate our discussion, we denote:

- **Native sensory data**: as the sensory data collected when a sentence is performed by user directly;
- **Generated sensor data**: as the sensory data generated by using the word-level ones by our method.

Overall performance. For a clear illustration, we group 41 sentences into eight groups with each group of around five sentences in Fig. 12. The accuracy of SP among eight groups is from 92% to 97.67%, and the overall average accuracy is 95%, which is similar to the accuracy reported in the original paper of SignSpeaker [12]. For GA, we only use the generated sensory data to train the network and test it on native sensory data directly. We can see that it achieves a reasonable accuracy from 62.8% to 75.8% among these groups. In the experiment, we observe that as long as we add very few native sensory data in the training, the system performance can be improved dramatically. In particular, when only one native sample is added, GA (denoted as GA-1) can achieve the accuracy from 76% to 85.6%. With the number of native samples is increased to three, GA-3 can achieve comparable performance with SP, *e.g.*, 93% on average. In the rest experiments, we adopt GA-3 as the default setting.

Fig. 12 suggests that GA already provides a good baseline. The major reason limiting its performance is the dis-similarity occurred in the sensory data generation process, which leads to certain differences on the obtained spectrum inputs. Fortunately, with the performance obtained in Fig. 12, we can foresee a good potential to employ more advanced domain transfer learning techniques to remove such differences for improving the accuracy, and we will try this opportunity in the future. Even with the current design, the data collection overhead can be reduced by 62.5% compared with that in SP.

SignSpeaker with fewer training data. In Fig. 13, we further investigate the performance of SignSpeaker if it also uses very few (two to three) sensory data samples directly

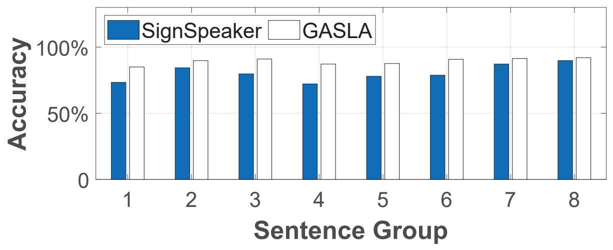

Fig. 13: Accuracy of *GASLA* and SignSpeaker when few native sensory data are used in the system training.

from the user. From the result, we can see that the performance drop of SignSpeaker is obvious compared with Fig. 12, *e.g.*, the accuracy reduction is 10.2% to 27.8%, which indicates that the importance of the amount of the sensory data samples used to train an ASL system. Different from SignSpeaker, with the generated sensory data as the major training data (they are generated from the word-level sensory data automatically and the word-level library is available in most ASL systems), *GASLA* can achieve a much higher accuracy with the same overhead of the native sentence-level sensory data collection.

C. Quality of Generated Sensory Data

To ensure the good performance of *GASLA*, one important requirement is that the generated sensory data for each sentence must be similar enough to the corresponding native data. In this experiment, we study the similarity between these two types of sensory data. To quantify the similarity, because the spectrum inputs of the neural network are 2D images, we utilize the structural similarity (SSIM) index, which is a mature indicator to measure the similarity between two images [16].

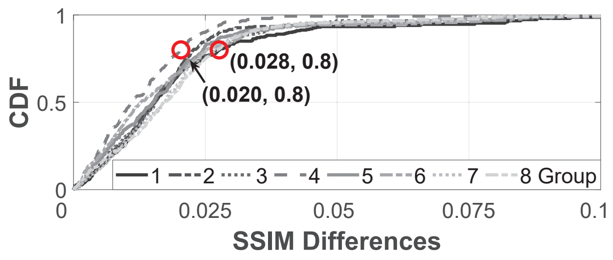

Fig. 14: CDF of the SSIM differences.

For each sentence, we first pick one native data sample as a base and calculate: 1) the average SSIM of all other native samples of this sentence with respect to the base, and 2) the average of SSIM of all the generated samples of this sentence with respect to the same base. Then, we compute the difference of these two average SSIM values. A smaller difference indicates the higher similarity between them. For each word, we repeat this process by using every native sample as the base. Fig. 14 shows the CDF of all the sentences among the eight groups. From the result, we can see that 80% of the SSIM differences are from 0.02 to 0.028, which are very small. Overall, 80% of this difference is less than 0.025 and the average is 0.018, which indicate that the generated sensory data samples are indeed similar to the native samples.

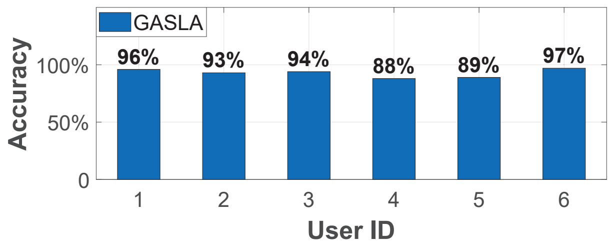

Fig. 15: Accuracy among six different users.

D. Micro-Benchmarks

We further examine a series of micro-benchmarks for a more comprehensive understanding of *GASLA*'s performance.

Different users. Fig. 15 illustrates the performance of *GASLA* among different users. The accuracy among different users is from 88% to 97%, and the average accuracy is 92.83%. These results indicate *GASLA* can achieve a consistently good performance cross all the users.

Different wrist movement speeds. Next, we examine the system performance under different wrist movement speeds in Fig. 16(a). In particular, the normal speed is about 1.5 seconds to perform one word, and we investigate the faster and slower speeds with about 1.0 and 1.9 seconds to perform a word, respectively. The network keeps unchanged (trained by the normal-speed data only as before) and we test it using the input data under different speeds. Fig. 16(a) shows that the accuracy under faster and slower wrist movement speeds is 87% and 93%, respectively. Since we have normalized the length of the input sensory data prior to processing, the speed difference is compensated and its impact is not significant.

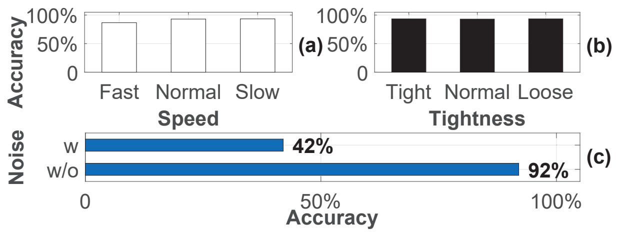

Fig. 16: Accuracy under different (a) wrist movement speeds and (b) wearing tightness levels. (c) Accuracy with and without noises in the sentence-level sensory data.

Different wearing tightness levels. We then investigate the impact of the wearing tightness of the smart watch in Fig. 16(b). In particular, the user wears the smart watch loosely or tightly intentionally. The network is still unchanged (trained by the normal-tightness data). We can see that the accuracy is stable under different wearing tightness levels, which indicates that the system is robust to user's different wearing styles.

Different quality of data. Without the *GASLA* design, the data collection overhead is high to setup an ASL system and it is possible to collect low-quality sensory data. If such low-quality sensory data are adopted in the system training, the system performance can be impacted. To understand this impact, we intentionally mix some noisy motion sensory data

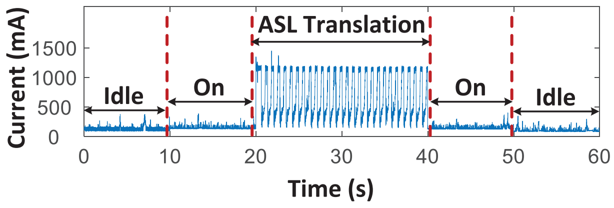

Fig. 17: Energy consumption of different states when the ASL system is executed on smart phone.

into the training data set. As Fig. 16(d) shows, the average accuracy with and without noises is 42% and 92%, respectively. Therefore, a lightweight design is desired in practice.

E. System Overhead

Latency and energy consumption. Finally, we examine the latency and energy consumption of *GASLA*. For the sentence-level sensory data generation, we have measured the latency for each technical module on the desktop. The result shows that the latency of pre-processing, search and concatenation is 1.05, 9.32 and 0.002 sec, respectively. The total latency is 10.372 sec, which is acceptable in practice. After the system is deployed on the smart phone, our measurement shows that the average time to translate one ASL sentence is 0.4 sec and the memory usage on the phone is 256 MB, which can be accommodated easily by existing mobile platforms. We measure the system's energy consumption on the phone by Monsoon power monitor in Fig. 17. In particular, the working current (mA) in the idle state and screen-on state is about 120 mA and 150 mA respectively. Fig. 17 shows that the average working current of the system is 700 mA. Such energy consumption can lead to nearly 4.29-hour battery life with a continuous execution on SAMSUNG Galaxy S7, which is acceptable for the practical usage of the ASL system.

V. Related Work

We review the related works of *GASLA* in this section.

ASL translation systems. To facilitate the communication for the deaf people, many different types of the ASL systems have been developed. The popular examples employ cameras or depth sensors with computer vision techniques, wireless, bio-metric sensors and wearable sensors to fulfill the design.

DeepASL [10] is an ASL system by using Leap Motion. The system in [19] is based on the on-board depth camera from a smartphone. The system proposed in [7] takes the video frames captured by an RGB camera as input. However, depth sensors or cameras usually have a fixed deployment, which limit the service area of the ASL system. In addition, they also require the line of sight between the use and device, as well as a good lighting condition. By using wireless signals, like SignFi [17] and mmASL [21], such requirements can be avoided, while their usage is still limited in a vicinity of the wireless device. To overcome these limitations, the bio-metric sensors and wearable sensors can provide portable and more ubiquitous solutions. For example, the system in [35] utilizes Photoplethysmography (PPG) sensors for the ASL translation. Some recent works further adopt the more commonly wearable sensors from different kinds of wearable devices in their system designs, *e.g.*, by using smart gloves [26], armband [34] and smart watch [12]. Parallel to the studies above, we observe a common applicability issue in existing ASL systems in this paper. We propose *GASLA* to address this issue to enhance the applicability and usability of ASL translation systems.

Motion sensor data processing. The *GASLA* design is also related to the motion sensor data processing, especially for accelerometers and gyroscopes. In addition to ASL, motion sensors are also used for other applications in the literature. For instance, various features are extracted from motion sensors for human activity recognition [32], [8], [29], [33], [11], user authentication [25], [6], [9], [36], security analysis to unveil the potential privacy leakage when users type sensitive information [14], [38], [28], [27], etc. Some recent works [15], [24], [23] are able to recover the moving trajectory of the user's arm using the motion sensor data from a single smart watch. *GASLA* is orthogonal to these existing works, which do not address the challenges encountered in this paper.

Data augmentation. Data augmentation [30] is a related technique to construct signals in *GASLA*, which mainly includes GAN (Generative Adversarial Network)-based methods [20], [13] and data processing methods [31], [22]. GAN-based methods usually train a generator to generate signals similar to the real signals [20], [13], while they require many real sensing signals to ensure the network's reliability and generalization. Data processing methods usually generate variants of the real signals through several transformations after the time and frequency domain analysis on the signals, such as cropping, warping, jittering, shifting, flipping and so on [31], [22]. However, they may not be effective to solve the applicability issue of the existing ASL systems. In this paper, we propose *GASLA* to address this issue.

VI. Conclusion

This paper presents *GASLA* to address a meaningful applicability issue for existing ASL systems. With *GASLA*, the sensory data of each to-be-recognized ASL sentence can be generated by using the word-level ones automatically. The generated sentence-level sensory data can be applied in the system training directly, which largely reduces the setup and maintenance overhead of ASL systems, because the amounts of the sentences and the data samples per sentence are large usually. We present *GASLA* atop the wearable sensors to instrument our design. To examine the effectiveness of this system design, we develop a *GASLA* prototype and conduct extensive experiments to evaluate its performance.

Acknowledgement

This work is supported by the GRF grant from Research Grants Council of Hong Kong (Project No. CityU 11217420), the CityU APRC grant (Project No. 9610485), CityU SRG-Fd (Project No. 7005666), and a grant from Chow Sang Sang Group Research Fund sponsored by Chow SangSang Holdings International Limited (Project No. 9229062). Zhenjiang Li is the corresponding author.

References

[1] https://www.gov.uk/government/publications/understanding-disabilities-and-impairments-user-profiles/saleem-profoundly-deaf-user.

[2] https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss.

[3] https://developer.android.com/reference/android/hardware/SensorEvent#values.

[4] https://en.wikipedia.org/wiki/Dynamic_time_warping.

[5] https://www.lifeprint.com/.

[6] N. Al-Naffakh, N. Clarke, and F. Li. Continuous user authentication using smartwatch motion sensor data. In *Proc. of IFIPTM*, 2018.

[7] K. Bantupalli and Y. Xie. American sign language recognition using deep learning and computer vision. In *Proc. of IEEE Big Data*, 2018.

[8] H. Bi, J. Zhang, and Y. Chen. Smartge: identifying pen-holding gesture with smartwatch. *IEEE Access*, 2020.

[9] Y. Cao, Q. Zhang, F. Li, S. Yang, and Y. Wang. Ppgpass: nonintrusive and secure mobile two-factor authentication via wearables. In *Proc. of IEEE INFOCOM*, 2020.

[10] B. Fang, J. Co, and M. Zhang. Deepasl: Enabling ubiquitous and non-intrusive word and sentence-level sign language translation. In *Proc. of ACM Sensys*, 2017.

[11] X. Guo, J. Liu, and Y. Chen. Fitcoach: Virtual fitness coach empowered by wearable mobile devices. In *Proc. of IEEE INFOCOM*, 2017.

[12] J. Hou, X.-Y. Li, P. Zhu, Z. Wang, Y. Wang, J. Qian, and P. Yang. Signspeaker: A real-time, high-precision smartwatch-based sign language translator. In *Proc. of ACM Mobicom*, 2019.

[13] C. Li, Z. Liu, Y. Yao, Z. Cao, M. Zhang, and Y. Liu. Wi-fi see it all: generative adversarial network-augmented versatile wi-fi imaging. In *Proc. of ACM Sensys*, 2020.

[14] Y. Liu and Z. Li. aleak: Privacy leakage through context-free wearable side-channel. In *Proc. of IEEE INFOCOM*, 2018.

[15] Y. Liu, Z. Li, Z. Liu, and K. Wu. Real-time arm skeleton tracking and gesture inference tolerant to missing wearable sensors. In *Proc. of ACM Mobisys*, 2019.

[16] D. Ma, A. Ferlini, and C. Mascolo. Oesense: employing occlusion effect for in-ear human sensing. *arXiv preprint arXiv:2106.08607*, 2021.

[17] Y. Ma, G. Zhou, S. Wang, H. Zhao, and W. Jung. Signfi: Sign language recognition using wifi. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2018.

[18] J. Nilsson. Improving the security of the android pattern lock using biometrics and machine learning, 2017.

[19] H. Park, Y. Lee, and J. Ko. Enabling real-time sign language translation on mobile platforms with on-board depth cameras. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2021.

[20] G. Ramponi, P. Protopapas, M. Brambilla, and R. Janssen. T-cgan: Conditional generative adversarial network for data augmentation in noisy time series with irregular sampling. *arXiv preprint arXiv:1811.08295*, 2018.

[21] P. S. Santhalingam, A. A. Hosain, D. Zhang, P. Pathak, H. Rangwala, and R. Kushalnagar. mmasl: Environment-independent asl gesture recognition using 60 ghz millimeter-wave signals. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2020.

[22] J. Sedmidubsky and P. Zezula. Augmenting spatio-temporal human motion data for effective 3d action recognition. In *Proc. of IEEE ISM*, 2019.

[23] S. Shen, M. Gowda, and R. R. Choudhury. Closing the gaps in inertial motion tracking. In *Proc. of ACM MobiCom*, 2018.

[24] S. Shen, H. Wang, and R. Roy Choudhury. I am a smartwatch and i can track my user's arm. In *Proc. of ACM MobiSys*, 2016.

[25] F. Sun, C. Mao, X. Fan, and Y. Li. Accelerometer-based speed-adaptive gait authentication method for wearable iot devices. *IEEE Internet of Things Journal*, 2018.

[26] N. Tubaiz, T. Shanableh, and K. Assaleh. Glove-based continuous arabic sign language recognition in user-dependent mode. *IEEE Transactions on Human-Machine Systems*, 2015.

[27] C. Wang, X. Guo, Y. Wang, Y. Chen, and B. Liu. Friend or foe? your wearable devices reveal your personal pin. In *Proc. of ACM ASIACCS*, 2016.

[28] C. Wang, J. Liu, X. Guo, Y. Wang, and Y. Chen. Wristspy: Snooping passcodes in mobile payment using wrist-worn wearables. In *Proc. of IEEE INFOCOM*, 2019.

[29] S. Wang, G. Zhou, A. Watson, L. Xie, M. Sun, and W. Jung. Wearable motion sensor-based chewing side detection. *Smart Health*, 2021.

[30] Y. Wang, J. Shen, and Y. Zheng. Push the limit of acoustic gesture recognition. In *Proc. of IEEE INFOCOM*, 2020.

[31] Q. Wen, L. Sun, F. Yang, X. Song, J. Gao, X. Wang, and H. Xu. Time series data augmentation for deep learning: A survey. *arXiv preprint arXiv:2002.12478*, 2020.

[32] J. Wu and R. Jafari. Orientation independent activity/gesture recognition using wearable motion sensors. *IEEE Internet of Things Journal*, 2018.

[33] L. Xie, X. Dong, W. Wang, and D. Huang. Meta-activity recognition: A wearable approach for logic cognition-based activity sensing. In *Proc. of IEEE INFOCOM*, 2017.

[34] Q. Zhang, D. Wang, R. Zhao, and Y. Yu. Myosign: enabling end-to-end sign language recognition with wearables. In *Proc. of ACM IUI*, 2019.

[35] T. Zhao, J. Liu, Y. Wang, H. Liu, and Y. Chen. Towards low-cost sign language gesture recognition leveraging wearables. *IEEE Transactions on Mobile Computing*, 2019.

[36] T. Zhao, Y. Wang, J. Liu, Y. Chen, J. Cheng, and J. Yu. Trueheart: Continuous authentication on wrist-worn wearables using ppg-based biometrics. In *Proc. of IEEE INFOCOM*, 2020.

[37] P. Zhou, M. Li, and G. Shen. Use it free: Instantly knowing your phone attitude. In *Proc. of ACM MobiCom*, 2014.

[38] T. Zhu, L. Fu, Q. Liu, Z. Lin, Y. Chen, and T. Chen. One cycle attack: Fool sensor-based personal gait authentication with clustering. *IEEE Transactions on Information Forensics and Security*, 2020.