Animation of Fingerspelled Words and Number Signs of the Sinhala Sign Language

MALINDA PUNCHIMUDIYANSE, The Open University of Sri Lanka
RAVINDA GAYAN NARENDRA MEEGAMA, University of Sri Jayewardenepura

Sign language is the primary communication medium of the aurally handicapped community. Often, a sign gesture is mapped to a word or a phrase in a spoken language and named as a conversational sign. A fingerspelling sign is a special sign derived to show a single character that matches a character in the alphabet of a given language. This enables the deaf community to express words that do not have a conversational sign, such as a name, using a letter-by-letter technique. Sinhala Sign Language (SSL) uses a phonetic pronunciation mechanism to decode such words due to the presence of one or more modifiers after a consonant. Expressing numbers also have a similar notation, and it is broken down into parts before interpretation in sign gestures.

This article presents the variations implemented to make the 3D avatar-based interpreter system look similar to an actual fingerspelled SSL by a human interpreter. To accomplish the task, a phonetic English-based 3D avatar animation system is developed with Blender animation software. The conversion of Sinhala Unicode text to phonetic English and numbers written in digits to sign gestures is done with a Visual Basic.NET (VB.NET) application. The presented application has 61 SSL fingerspelling signs and 40 SSL number signs. It is capable of interpreting any word written using the modern Sinhala alphabet without conversational signs and interprets the numbers that go up to the billions. This is a helpful tool in teaching SSL fingerspelling and number signs of SSL to deaf children.

CCS Concepts: • **Computing methodologies** → **Computer graphics**; **Animation**; • **Applied computing** → **Arts and humanities**; **Language translation**

Additional Key Words and Phrases: Sinhala sign language, Sinhala fingerspelling, number gesture animation, 3D signing avatar

ACM Reference Format:
Malinda Punchimudiyanse and Ravinda Gayan Narendra Meegama. 2017. Animation of fingerspelled words and number signs of the Sinhala Sign Language. ACM Trans. Asian Low-Resour. Lang. Inf. Process. 16, 4, Article 24 (August 2017), 26 pages.
DOI: http://dx.doi.org/10.1145/3092743

1. INTRODUCTION

Sinhala Sign Language (SSL) is the preferred communication medium of the aurally handicapped Sinhala community in Sri Lanka. To communicate with a deaf person, a hearing person must either be conversant in SSL or have access to a sign interpreter.

This work is supported by the National Science Foundation of Sri Lanka, under the technology grant TG/2014/Tech-D/02. The Nvidia Titan X graphics card used for this research was donated by the NVIDIA Corporation.
Authors' addresses: M. Punchimudiyanse, Department of Computer Science, Faculty of Natural Sciences, The Open University of Sri Lanka, Nawala, Nugegoda, Sri Lanka, 10250; R. G. N. Meegama, Department of Computer Science, Faculty of Applied Sciences, University of Sri Jayewardenepura, Gangodawila, Nugegoda, Sri Lanka, 10250.

DOI: http://dx.doi.org/10.1145/3092743

Due to the limited number of sign interpreters, the deaf community faces communication hurdles when interacting with hearing people in day-to-day life.

There are two main gesture styles in SSL. The first style, the conversational style, has a set of sign gestures for common words and phrases that have one-to-one correspondence to words in Sinhala sentences. New conversational style signs are being adopted and developed by the deaf and interpreter community with the assistance of the National Institute of Education (NIE) in Sri Lanka, which publishes those signs in the form of books once or twice a year [1].

The second style uses the fingerspelling alphabet to decode Sinhala words that do not have sign gestures into a sequence of signs that correspond to the phonetic pronunciation of each letter of that particular word [2, 6]. For example, the name සුමිත් (Sumith) can be decoded into a character sequence as follows: ස් + ු + ම් + ි + ත් (S + U + M + I + TH).

Representing numbers using SSL requires the decoding of a given number into digits, decades, hundreds, thousands, millions, and billions unless it has a single sign gesture. The number signs used in SSL are given in [2].

The deaf schools scattered throughout Sri Lanka have several dialects comprising different sign gestures to match the same Sinhala word in the conversational style and different ways of interpreting unknown words. Some schools, such as the deaf school in Rathmalana, prefer to interpret names in English fingerspelling, while others, such as those in Balangoda and Matara, teach Sinhala fingerspelling to deaf children for the purpose of interpreting names and words that do not have sign gestures.

We have presented an avatar system to handle conversational style of SSL in our work [3], which also includes limited work on fingerspelling. However, it does not include the complete fingerspelling alphabet (Figure 6), the signing variations of fingerspelling and signing numbers in digits that are discussed in this study. This article focuses on presenting the features of the SSL fingerspelling alphabet; discussing undocumented strategies used in SSL fingerspelling; interpreting full numbers in SSL and a mechanism of animating fingerspelling and numbers in SSL using a computer-animated character (3D human). Furthermore, the techniques proposed in this research to sequence and animate all the variations in SSL fingerspelling and number signing are based on the feedback obtained from three professional sign interpreters with several years of experience in SSL interpretation and teaching SSL. Two of them have involved with NIE in developing SSL to the stage where we have it today from the 1980s. The presented system processes Sinhala Unicode text to convert it to appropriate fingerspelled signs and number signs, and it includes the number sign set of 40 signs and the entire fingerspelling sign gesture set of 61 characters related to the modern Sinhala alphabet and several additional signs required to represent modifiers in SSL.

2. REVIEW OF LINGUISTIC AND TECHNICAL BACKGROUND

2.1. English Sign Language Fingerspelling System

Fingerspelling is the technique used to display names and words that do not have conversational gestures in any sign language. For English, 26 fingerspelling sign gestures are used to spell a name or an unknown word, and it is a straightforward task of one-to-one mapping. For example, to spell the name “John,” English fingerspelling signs in British sign language (BSL) corresponding to “j,” “o,” “h,” and “n” can be shown, as given in Figure 1.

Furthermore, BSL uses both hands in a single posture to show characters in the English alphabet as shown in Figure 5 except for the characters J and H where hand movement is required to denote the letters J and H. The American Sign Language

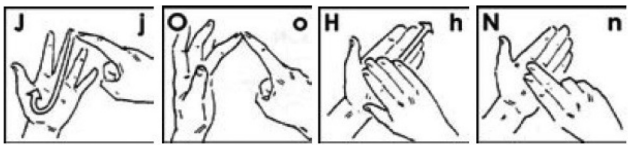

Fig. 1. English fingerspelling signs for the name “John.”.

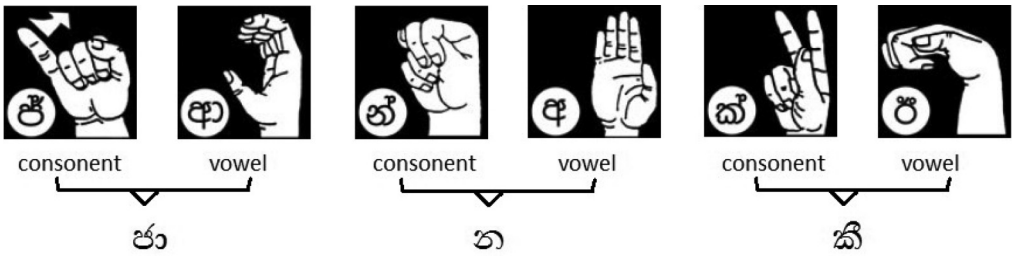

Fig. 2. Sinhala fingerspelling of the name “Janaki.”

(ASL) alphabet for English uses a single hand and a single posture for fingerspelling for most characters except for the letters J and Z, which use hand movements to denote signs. In contrast, SSL uses one’s single hand for fingerspelling signs and signs may have single posture or multiple postures based on the different variants of characters in the Sinhala alphabet. Most of the multi-posture SSL signs are having either rotational or sliding movements between postures.

The generally accepted convention is that the right hand of a person is used to show the fingerspelling signs of SSL, but some sign interpreters who are left handed use their left hand to show the fingerspelling signs.

2.2. Sinhala Sign Language Fingerspelling System

When a Sinhala word is fingerspelled, the decoding of the word into a character sequence becomes complex, because the Sinhala fingerspelling system works according to the phonetic pronunciation mechanism. Let us briefly look at how the phonetic pronunciation system works in the Sinhala language.

A consonant written with the ් (al) modifier is considered as the base or pure form of the consonant, and a hidden “a” sound is attached to the consonant to derive the standard consonant in the Sinhala alphabet. For example, ක (ka) is a standard consonant, while ක් (k) is the base form of the same consonant. There are modifiers in the Sinhala language of which one or more can be attached to the base form of a consonant to provide different sounds, making it a derived consonant. Most of the modifiers in Sinhala have an associated vowel sound [5].

When a word is phonetically decoded, the vowel associated with the modifier is written instead of writing the modifier after the consonant [6]. For example, the name ජානකී (Janaki) is pronounced in Sinhala as Jaanakiee and phonetically decoded as ජ් + ආ + න් + අ + ක් + ී (jh + aa + n + a + k + ee). The phonetic decoding of the word is processed in sequence as a consonant followed by a relevant vowel to get the corresponding SSL fingerspelling sequence, as depicted in Figure 2.

There are two special modifiers in Sinhala, රකාරාංශ “rakaransha” and සඤ්ඤක “sanghaka,” which we refer to as RK and SK in this article. An RK modifier is written as a half circle below the consonant and adds the sound “r” to the consonant along with the hidden vowel sound “a.” For example, when the RK modifier is added to the base consonant ක් (k), the derived consonant ක්‍ර (kra) is formed, as shown in Figure 3(a). The

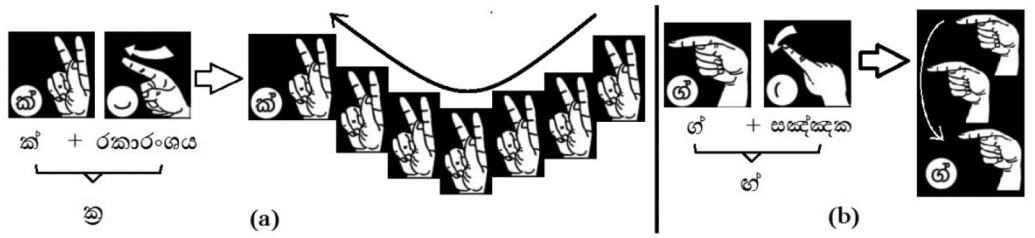

Fig. 3. Fingerspelling examples for (a) RK and (b) SK modifiers.

SK modifier is written as an adjoining half circle to the left of a consonant, as shown in Figure 3(b), which adds the "n" sound (not the "en" sound) to the consonant, making its pronunciation more nasal. Adding the SK modifier to the consonant ග (ga) converts it to the ඟ (nga) consonant. The pure and standard consonants with SK modifiers belong to the modern Sinhala alphabet, and therefore, they are classified as consonants (not derived consonants). When another modifier is added to a consonant with an SK modifier, then it becomes a derived consonant.

In SSL, there are variations in the fingerspelling of words with RK and SK modifiers, because the sign gesture related to the consonant is modified with the movement related to the respective RK or SK modifier, as depicted in Figures 3(a) and 3(b).

If a consonant has a second modifier in addition to the RK or SK modifier, then an appropriate vowel sign related to the second modifier is displayed. For example, the letter ක්‍ර (kra) only has an RK modifier, while the letter ක්‍රී (kri) has two modifiers ක් + රකාරංශය + ී (k + RK modifier + i).

A common variation exists when the same character is presented twice as adjoining letters of a particular word. This generally happens when a consonant with the ් (al) modifier is followed by the same consonant with or without a different modifier. For example, the word සද්දෙයි (sadhdheyi, meaning noisy), decoded as ස් + අ + ද් + ද් + එ + ය් + ි (s + a + dh + dh + e + y + i), has two ද් (dh) consonants together. In such situations, professional sign interpreters show the sign gesture for the first character closer to the body, then move the hand slightly away from the body, maintaining the posture of the first character to express that the second character is equivalent to the first character.

When processing pauses between words, the hands of the interpreter generally move back to the idle position before displaying the starting character of the next word.

Two types of fingerspelling signs exist in the Sinhala fingerspelling alphabet [2, 6]. The first type is a single posture sign where one hand posture determines the character, while the other type of sign has multiple postures with a movement between postures determines a particular sign.

The අල්ප ප්‍රාණ "alpa-prana" (AP) and මහ ප්‍රාණ "maha-prana" (MP) difference of certain consonants in the Sinhala alphabet is a key character classification considered by the designers of Sinhala fingerspelling signs. While they have similar pronunciations, the difference between AP and MP consonants is that there is more emphasis on the pronunciation of MP consonants than AP consonants. Sinhala has several such AP and MP letter pairs [7]. The most notable pairs are න, ණ (with the "na" sound) and ල, ළ (with the "la" sound).

The designers of Sinhala fingerspelling signs have adopted the convention of using a single posture for an AP consonant, while its relative MP consonant has a multiple posture sign. An MP character sign is constructed with a rotation or a slight movement of the hand while keeping the posture of the AP consonant. For example, the AP consonant න (na) is a single posture sign, while the MP consonant ණ (na) has a

Fig. 4. (a) Single posture sign for න (na) and (b) multi-posture sign ණ (Na).

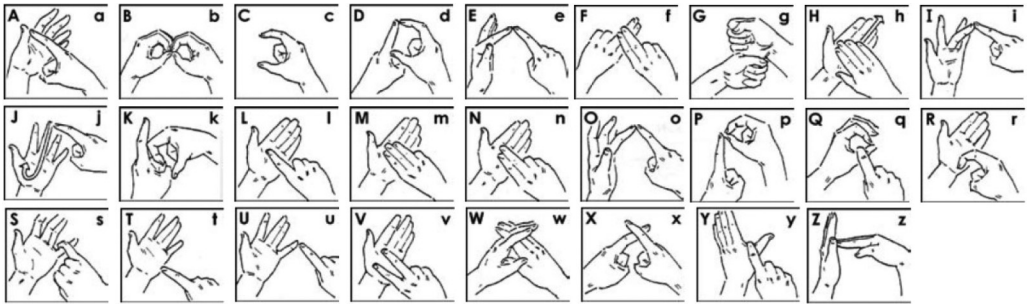

Fig. 5. English fingerspelling alphabet in BSL.

multi-posture sign, and those signs are differentiated only by a rotation, as shown in Figure 4.

The English fingerspelling alphabet that is used with SSL is given in Figure 5 which is a replica of BSL, and the Sinhala fingerspelling alphabet is shown in Figure 6. Unlike BSL or ASL, SSL has directional rotations, sliding of hands in different directions to denote fingerspelling signs. This makes modeling and animation of SSL signs complex in a graphical environment.

When two same characters occur at the same time, the most common technique used by the signers of SSL is to slide the hand by keeping the pose of the character to denote that there are two of them. However, when there are rotations in a character as shown in Figure 4(b), the hand must rotate to the initial position before sliding and showing the same character.

Another important issue in SSL fingerspelling is that two characters may have same initial posture position while one character is a single posture character. For example, both characters in Figures 4(a) and 4(b) have the same starting posture. When these two characters are signed together in න, ණ order, one cannot determine whether there is a character න, because ණ also start with the same posture coordinates. Practically, signers solve this problem by showing න first, then move the hand slightly away from their body and finally show ණ. This issue is common for most of the AP and MP characters occurring together in AP, MP order.

2.3. Number Signs in Sinhala Sign Language

The following number signs are available in SSL.

1. Sign gestures are present for numbers from 0 to 19.
2. All decade values have signs from 20 to 90.
3. All hundreds have signs from 100 to 900.
4. Signs exist for thousand, one hundred thousand, million, and billion.

Fig. 6. Sinhala fingerspelling alphabet.

When a number is interpreted in SSL, all the numbers with the specific signs given above can be animated properly. All other numbers are constructed with signs related to digits along with the necessary signs for hundred, thousand, million, and billion. For example, the number 4,756 is shown in the following five signs: "four" + "thousand" + "seven hundred" + "five" + "six," as shown in Figure 7.

It is important to note that we use the signs for the numbers five and six for fifty-six instead of the signs fifty and six. If the two adjoining numbers are the same, then the same sign is shown by changing the horizontal position of the number pose, as shown in Figure 8.

2.4. Computer-based Interpretation of Sign Gestures

Computer science researchers use three different techniques when interpreting a particular spoken/written language to its respective sign language.

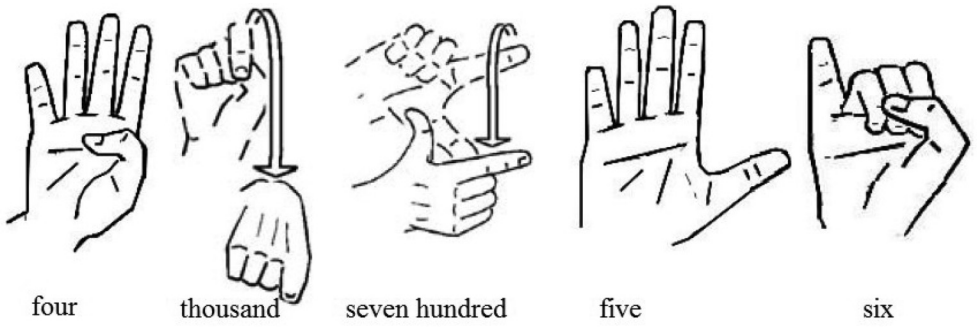

Fig. 7. Sign gestures showing the number 4,756.

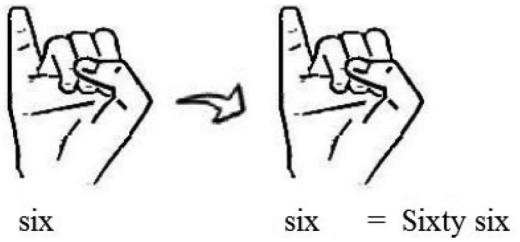

Fig. 8. Representing sixty-six in SSL sign gestures.

In the first technique, video of an actual person interpreting each sign is sequenced in the order of the words or characters. The missing frames between the different signs causes a problem, as the start position of the hands in the current sign does not start from the ending position of the previous sign in the video sequencing mechanism. To prevent misalignment of hands in joining videos, Solina et al. [8] proposed having the sign video clips recorded by the same interpreter in a single session. In this way, the interpreter's palms can be detected using computer vision techniques, and the start position of each sign can be adjusted by dropping frames until the palm position of the previous sign matches the starting palm position of the current sign. Chuang et al. [9] proposed a NURBS based special interpolation technique to identify best cut points of two sign video clips for joining and generate movement epenthesis between those cut points. Wang et al. [10] proposed a technique of adding hand transition videos in between two different sign video clips to prevent misalignment of hands for building Chinese weather reports in the Chinese Taipei sign language. They have collected the transition videos between signs from previous transition videos of the same signer or morphed the missing transitions. Efthimiou and Fotinea [11] have built an annotated video corpus to sequence videos of Greek sign language (GSL) taking grammar and other structural differences between Greek and GSL in to account.

The second technique involves using motion capturing to record the sign interpretation of a particular word/character by an actual interpreter and sequence signs according to the words or characters in a sentence. FAITH, a system developed for teaching Brazilian Sign Language (LIBRAS) fingerspelling, uses motion-captured fingerspelling signs to interpret words [12]. The hand movements of the transitions between two signs are constructed from the motion-captured data. Heloir et al. [13] used Cyber gloves and infrared cameras to capture finger and body movements and then used principal component analysis (PCA) to segment motion-captured data in order to construct new animations and transitions from them. Awad et al. [14] discussed a database driven

architecture for indexing and sequencing of motion-captured data to construct animations in the French sign language. Even though the animations are more realistic when the motion-captured sequences are played back using a 3D avatar, the addition of new signs takes a substantial amount of time, and the technique is very expensive due to the high cost of motion capture hardware. Post-processing of motion-captured data may be required to correct bone size differences between the signer and the avatar in order to use them with signing avatars [15].

The third technique involves mathematically calculating the hand positions of each gesture based on the posture positions and animating the human avatar accordingly. The technique proposed for animating American Sign Language (ASL) involves dividing the pose into different bone channels for an animation [16]. A 3D hand animation system is used to process English fingerspelling animations in real time using posture coordinates [17]. A signing tutor system named PAULA is used to train hearing adults in ASL fingerspelling [18] as well as to teach deaf children the ASL curriculum [19]. An avatar system (v-guido) used in the e-sign project utilizes a markup language named SiGML to define sign postures from a predefined set of poses (core signs) [20].

The first two techniques have their own pros and cons. Namely, video-based systems are easy to produce, but miss video frames of transitions when switching between signs while motion-capture-based systems produce more realistic signs, but the high cost of the motion-capture hardware makes it difficult for such systems to evolve with new signs. The dynamic calculation of the posture positions of a sign gesture employed in the third technique is a better option to take when designing a computer-based interpreter [21] capable of continuous sign synthesis [22, 23]. Hence, the third technique is used in this research with custom-developed variations to suit SSL. In addition, pauses between signs, proper reproduction of signing variations and a proper speed of sign synthesis are essential to ensure the avatar-based signing comprehended by a deaf person [24, 25].

3. METHODOLOGY

This system converts Sinhala text written in Unicode to phonetic English via a VB.NET application and feeds it to the animation system developed with Blender animation software [27]. It dynamically compiles a sequence of sign gesture animations related to the characters of a given word or a number. Moreover, it starts the animation of the current character right from the end posture of the previous character. Therefore, the system is modeled to mathematically calculate transitional posture positions of the avatar for a smooth animation.

3.1. Implementing Sinhala to Phonetic English Conversion

The animation system does not support non-English text in its interfaces; hence converting Sinhala to phonetic English is necessary. Each character in the Sinhala alphabet is given a tag from the 26-letter English alphabet. Tags are chosen by improving the tag set presented by Wasala and Gamage [5], and the following convention is used when deriving tags and converting Sinhala Unicode to phonetic English.

1. A Sinhala character is given an English tag that is pronounced similarly.
2. A larger tag is chosen for the MP counterpart of a given AP character.
3. Tag searching is performed in large tag to small tag order.
4. The “al” modifier is assigned the tag “w” and the non-printing zero width joiner (ZWJ) character is assigned the tag “qx”.

When a Sinhala word is converted to phonetic English, the characters in Sinhala Unicode are replaced with tags given in Table I. For example, the word ඵලිය (result)

Table I. Frequency of Special Characters

Ch	Tag	Ch	Tag	Ch	Tag	Ch	Tag	Ch	Tag	Ch	Tag	Ch	Tag
18 vowels and 2 half vowels													
අ	a	ආ	axa	ඇ	xcae	ඈ	aeae	ඉ	i	ඊ	ixi	උ	u
ඌ	uxu	එ	e	ඒ	exe	ඔ	o	ඕ	oxo	ඵ	zilu	ඵූ	ziluu
ඍ	zri	ඍා	zrii	ඓ	ai	ඖ	xau			ං	xonx	ඃ	zkf
41 consonants													
ක්	k	ඛ	zk	ග්	g	ඝ්	zg	න්	n	ණ්	zn	ල්	l
ව්	v	ච්	ch	ජ්	jh	ඣ්	s	ළ්	zl	ය්	y	ර්	r
ඨ්	zth	ඳ්	dh	ඩ	zdh	හ්	h	ශ්	sh	ෂ්	zsh	ත්	txh
ඨ්	zt	බ්	b	භ්	zb	ම්	m	ඩ්	d	ඪ්	zdx	ට්	t
ඛ	zon	ෆ්	f	ඡ්	ch	ඟ්	zng	ඹ්	xmb	ප්	p	ඵ්	zp
ඳ	qndh	ඡ	xjhx	ඬ	zndx	ඤ්	cn	ඣ	zjh	ඥ්	jhcn		
Extra tags for modifiers, al character and zero width joiner (zwj)													
ෙ	zaik	ා	zruu	zwj	qx	්	w	ෛ	zau				

is phonetically decoded as (ප් + (U+200D) + ර් + අ + ත් + ඉ + ළ් + අ + ල් + අ). When replaced with tags in Table I, the word ප්‍රතිඵල produces the output as “pwqxratxhizpala” (p w qx r a txh i zp a l a). The example also shows the placement of zero width joiner character (U+200D) with tag “qx” when writing an RK modifier and use of “w” tag for “al” modifier when there is no vowel following a consonant (Eg. ප් = p w).

Algorithm 1 is used to convert Unicode Sinhala text to phonetic English text in this research, which we have originally presented and validated in our work in [4]. Algorithm 1 is designed as follows. It replaces all vowels appearing in the input text with its corresponding tag and use “@” and “#” to distinctly separate vowel-type tag from the rest of the text. Then it replaces consonants with their respective tags and use “$” and “!” to demarcate consonant-type tags. In Sinhala, almost every modifier has its corresponding vowel sound. Therefore, we replace such modifiers with their respective vowel-tag and use “-” and “∼” to demarcate a vowel-tag used for a modifier. Then the ZWJ character is tagged with “qx” with “$” and “#” as the separator characters. Algorithm 1 detects a consonant-tag followed by a consonant-tag, a consonant-tag followed by direct vowel-tag and a consonant-tag followed by a space using the special characters used as prefixes and suffixes. Those are the places where there are no modifiers attached to a consonant. In such places, a hidden “a” sound representing vowel “අ” is present according to the Sinhala language phonology. Therefore, we inject vowel-tag “a” for hidden “අ” in such places. Finally, we remove all the special characters that are used as prefixes and suffixes of tags and return the phonetic English tag sequence as the output of Algorithm 1.

The output of Algorithm 1 uses the “space” character between tags only to separate words. Hence, the animation system of this research requires a mechanism to identify each Sinhala character from such sequence of tags in order to map it to the relevant fingerspelling sign. The following tag planning process made it possible to have

non-ambiguous identification of individual tags when searched from a large tag to a small tag according to convention(3).

ALGORITHM 1: Sinhala Unicode to Phonetic English Conversion Algorithm

// abbreviations: zero width joiner - ZWJ, carriage return - CR,
Input: Variables *TR* = input string in Unicode Sinhala without special characters;
Output: Variable *TR* = phonetic tag sequence after the processing;
// Sinhala character and phonetic tag placed in three categories (vowels, joiners(modifiers) and consonants)
vowels ← "අ-a,ආ-axa,..,ඔ ඕ-xonx"
joiners ← " ◌ා-axa,◌ි-i,◌ී-ixi,◌ු-u,..,◌aa-zruu,◌්-w"
conson ← "ඟ-zch,ඹ-zng,..,ෆ-f"
//processing section
Split *vowel* to two lists *vowel* & *phoneticvowel*
Split *conson* to two lists *consonant* & *phoneticconsonant*
Split *joiner* to two lists *modifier* and *phoneticmodifier*
Remove non-printable characters in *TR* **while keeping** ZWJ and CR
Repeat Until end of *vowellist*
    TR ← *TR*.replace(*vowel*, "@"+ *phoneticvowel* +"#")
End Repeat
Repeat Until end of *consonantlist*
    TR ← *TR*.replace(*consonant*,"$" & *phoneticconsonant* & "!")
End Repeat
Repeat Until end of *modifierlist*
    TR ← *TR*.replace(*modifier*,"-" & *phoneticmodifier* & "∼")
End Repeat
//zero width joiner replacement
TR ← *TR*.replace(ZWJ,"$qx#")
//introduce "a" sound for non *al* consonants
TR ← *TR*.replace("!$", "!-a∼")
TR ← *TR*.replace("!@", "!-a∼@")
TR ← *TR*.replace("! ", "!-a∼ ")
TR ← *TR*.replace("!" & CR, "!-a∼" & CR)
TR ← *TR*.replace("[\\!<>@#$`%^/&*."",()_+ = ∼ | {}?;:\['\-\]]", " ")
Return *TR*

In Sinhala, most vowels are paired in short and long pronunciation. For example, "අ", "ආ" is one such pair. Even though the ideal tags for ("අ", "ආ") pair are "a" and "aa" based on the pronunciation, large tag to small tag search may classify two "a" tags written together as one "aa" because "aa" is searched before "a". Therefore, we have designed the tags to be "a" and "axa" to prevent wrong classification in searching large tag to small tag. This technique is used for designing most of the other vowel pairs.

No individual character in Sinhala has "z", "c", and "x" as phonetic tags. Therefore, we have used "z" character in front of an AP tag to derive its corresponding MP tag. For example, tag "k" for character "ක්" and tag "zk" for its MP character "ඏ". When large tags are searched before small tags, all MP characters are identified before AP characters, thereby eliminating the possibility of wrong identification in extremely common occurrences of AP and MP together in Sinhala. For example, take the segment "dhzdh" in the word "bxaudhzdha" meaning Buddhist. If the tags are searched from large to small tag "zdh" and "dh" are separated properly. Otherwise, erroneous identification would occur as "dh" + z + "dh" if the small tag "dh" is searched first.

ALGORITHM 2: NtoS (number to sign) Recursive Algorithm

```
//This recursive algorithm NtoS converts numbers in digits to SSL number sign gestures.
// Shortened form of functions; LN = length, LT = left and RT = right functions respectively
Input: Variables NV = number as value; N = number as string;
Output: Variable OT output the number gesture sequence;
OT ← "";
P1 ← {Array of sign gesture names from 0 to 9}
P2 ← {Array of sign gesture names; P2(0) is an empty string and from P2(1) onward
      list of sign gesture names from 10–19, 20–90 in decades, 100–900 in hundreds}
If LN(N) = = 1 Then OT ← P1(NV)
Else If LN(N) > 1 Then
    If NV = = 0 Then OT ← ""
    Else If NV < 20 and NV >= 10 Then OT ← P2(RT(N,1)+1)
    Else if NV < 99 and Val(RT(N,1)) = = 0 Then OT ← P2(LT(N,1)+9)
    Else if NV < 999 and Val(RT(N,2)) = = 0 Then OT ← P2(LT(N,1)+18)
    Else
      If LN(N) = = 2 Then OT ← NtoS(LT(N,1)) + " " + NtoS(RT(N,1))
      If LN(N) = = 3 Then OT ← NtoS(LT(N, 1)+ "00") + " " + NtoS(RT(N,2))
      If LN(N) > 3 and LN(N) <= 6 Then
          OT ← NtoS(LT(N, LN(N)-3)) + "dhahasa" + NtoS(RT(N,3))
      Else If LN(N) > 6 and LN(N) <= 9 Then
          If Val(LT(RT(N,6),3)) > 0 Then
            OT ← NtoS(LT(N, LN(N) - 6)) + "miliyana" + NtoS(RT(N,6))
          Else
            OT ← NtoS(LT(N, LN(N) - 6)) + "miliyana" + NtoS(RT(N,3))
      Else If LN(N) > 9 and LN(N) <= 12 Then
          If Val(LT(RT(N, 9), 3)) > 0 Then
            OT ← NtoS(LT(N, LN(N) - 9)) + "biliyana " + NtoS(RT(N, 9))
          Else If Val(LT(RT(N, 6), 3)) > 0 Then
            OT ← NtoS(LT(N, LN(N) - 9)) + "biliyana" + NtoS(RT(N, 6))
          Else
            OT ← NtoS(LT(N, LN(N) - 9)) + "biliyana" + NtoS(RT(N, 3))
If InStr(OT, "binwdhuva") > 1 Then
    If Trim(OT) <> "binwdhuva" Then OT = OT.replace("binwdhuva", " ")
OT ← OT.replace(" ", " ")
Return OT
```

3.2. Implementing Digit-to-Number-Sign Conversion

The number in digit form is converted to a string of number gestures using the following recursive algorithm implemented in VB.NET.

This algorithm works by dividing a number string into a set of three digits and processing recursively by introducing the connecting words thousand, million, and billion. Numbers from 0 to 999,999,999,999 can be converted to SSL number sign sequence using Algorithm 2.

The numbers written in the textual form of Sinhala also differ from the corresponding number-sign gesture sequences. Instead of writing a string replacement function, this research uses an alternative technique to animate such numbers. For example, in Sinhala, for the number "25," we write විසි පහ (visi paha) in textual form, but the animation sequence of "25" has to be දෙක පහ (dheka paha). Therefore, the sign used to show the number "2" is duplicated and assigned a new sign named "visi." Then, it is stored in the sign database so that every time the word "visi" occurs in the word sequence, the sign for "2" is animated. There are eight such variations in decade values from 20 to 90 and nine such variations in hundreds from 100 to 900. The variations

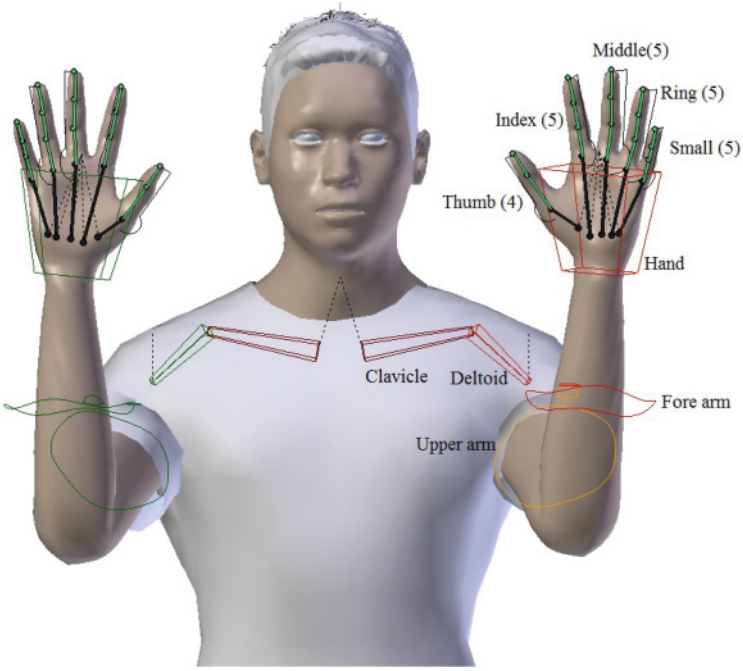

Fig. 9. Upper body of avatar with bone names. Number of bones in fingers are given in brackets.

in the Sinhala words relating to thousand, million, and billion are also handled in the same manner.

3.3. Modeling the 3D Human

Unlike a computer game where a 3D model is controlled by a human player, a 3D human that can be controlled via a series of commands is required. Most of such models are commercially available for purchase. In this research, we have opted to design a human model using a free 3D human avatar-building tool named MakeHuman [26], which has many features to change the model, such as gender, age, clothing, and the dimensions of different body parts.

The human model is then imported to Blender animation software [27], and a skeleton (rig) is attached for animation. Figure 9 depicts the upper body segment of the rigged human model used in this research.

The bone set of the skeleton is comprised of over 70 bones, including 25 different bones/bone groups in the area of the palm and fingers of each hand. Each bone of the finger has X,Y,Z coordinates (Eular) and is only allowed to rotate in the X axis in order to prevent abnormal poses. In order to prevent gimbal lock [28], W,X,Y,Z coordinates (Quaternion) are used for the other bone groups. A higher concentration of bones is used in the palm and finger area to allow the precise hand/finger movement required to display the exact form of the fingerspelled signs. Automatic adjustments through inverse kinematic (I-K) influences are disabled to allow all the bones to move according to the exact coordinates assigned by the animation script.

The rendering system is changed to a Blender game engine (BGE), and the animation system is developed using Python scripting in Blender. The animation script is controlled by the “always sensor” of BGE that runs at the idle frame rate frequency of 60Hz, but the system slows down when there is a graphical or logic-based process

Table II. Bone Coordinates of the Different Lines of a Sign Definition

Line	Bone
1	Coordinates of both left- and right-hand clavicle, deltoid, upper arm, forearm, and hand bones
2	Coordinates of left-hand bone groups of thumb, index, middle, ring, and small fingers and bone parts of thumb, index, middle, ring, and small fingers of left hand
3	Coordinates of right-hand bone groups of thumb, index, middle, ring, and small fingers and bone parts of thumb, index, middle, ring, and small fingers of right hand

being executed in a script. Therefore, the actual frame rate of animations did not go beyond 15 frames per second in the present research.

3.4. Sign Database

A text file is used as the sign database with the intention of parsing line by line. A sign may have one or more hand poses, and each hand pose is defined in the sign database in three different lines of numerical data, which are the W,X,Y,Z coordinates of the different bones. Table II illustrates the bones whose coordinates are defined in a specific line.

3.5. Fingerspelling Animation System

The animation system of this research displays the 3D avatar frame by frame after changing its bone positions in small increments to obtain a smooth hand motion from one location to another. The final position of the hand defines a single pose sign, while multiple poses and movement in between poses define a multi-pose sign. A user can define the number of frames per animation of a given sign or a given pose. The deviation (increment) from the current bone position to the next frame's position is calculated by taking the displacement of each bone between the start position and the end position of the pose and dividing it by the number of frames. The transition between two signs is handled by taking the difference between the end position of the previous sign and the start position of the current sign.

When an unknown word is detected, the word is decoded into a sequence of phonetic tags related to characters. The pose positions of the tags are looked up from the sign database and sequenced in the play list. The user is allowed to set the wait time between poses in number frames so that the animation will play at an acceptable speed, while the pose is kept on the screen during the wait time for the viewer to identify the sign gesture.

3.6. Variations Provided for Realistic Signing of the 3D Avatar

Word separation is achieved by introducing an idle gesture to the playlist at the end of each fingerspelled word where the avatar's hands move back to an idle position, enabling the viewer to identify the end of the word.

Only one instance of posture positions of each digit is defined in the sign database and decade values of a number is decoded as single digits. For example, "thirty-three" is animated as "three" and "three". When two adjoining digits of decade values of a number are same, the viewer does not have any visual indicator that two numbers are being signed because of the same posture positions (e.g., 22, 55, and 77 in the number 22,255,677). Human interpreters handle such situations by displaying the second number slightly away from the body than the first number. By introducing the extra pose position to move the hand slightly toward the body and adjusting the upper hand and forearm of the avatar slightly away from the body for the second number, a virtual V-shaped movement similar to the actual human interpretation can be achieved. Hence, the system is implemented to introduce an extra pose and modify

the upper hand and the forearm posture coordinates of the second number dynamically when two similar adjoining numbers are detected in decade values.

ALGORITHM 3: Algorithm for Building Playlist of Sign Gestures in the System

Input: Sentence *S*
Output: Play list of sign gestures
// J,P are signs constructed from previous sign coordinates, D is the sign for idle hand coordinates
//A1 = number sign sequence of first instance of multiple or single-digit number in S.
// Mk is a number sign corresponding to a single digit of a multi-digit number.
Search *S* for number sign sequences and add them to array $A \leftarrow (A1..Ae)$
For each number sequence from *A1* to *Ae*, **do**
    Split number sequence into number signs $M \leftarrow M1..Mp$
    Search for two identical number signs in *M* where $Mk == Mk+1$, $1<k<p$
        If found
            Read pose coordinates of Mk and Mk+1 from the sign database
            Construct a sign J by altering forearm and upper arm positions of the pose Mk
            Insert sign *J* in *M* between Mk and Mk+1
            Temporarily shift upper arm and forearm positions of all the poses of Mk+1
            Update *A* with added *J* and changed Mk+1
        end
End For
Update *S* with updated signs in *A*
Look up known words in sign database from *S* and mark the rest as unknown words
For each unknown word remaining in *S* ***do***
    Search the word from large tag to small tag to obtain phonetic tag sequence C,
    $C \leftarrow (C1 .. Cn)$
    Search in C for two identical characters where $Cx == Cx+1$ and $1 < x < n-1$
        If found
            Read pose coordinates of Cx+1 from sign database
            Temporarily shift upper arm positions of all the poses of sign Cx+1 to place it right of the Cx
            Update C with changed Cx+1
        end
    Update S with updated C
    Append idle sign gesture D at the end of each word
End For
Remove tags for ZWJ and "al" modifier from *S*
Populate *Playlist* with signs in S in the order of words and fingerspelling signs of unknown words in *S*
Return PlayList

An approach slightly different to the animation of two similar adjoining numbers is used in displaying two similar adjoining characters. For example, let us take the word වත්ත (vaththa, meaning garden) that has two adjoining ත් (th) characters. The animation script changes the upper hand coordinates of the second "th" character to move it to the right of the first "th" character without adding an extra pose or change of first "th" character pose.

The variation of consonants with RK modifiers and SK characters is achieved by introducing new signs for them and introducing an additional type of pose delay to the animation system. For example, the derived consonant ක්‍ර (kra), which is made by combining ක් + ර් + අ (k + r + a), requires moving the pose of letter ක් (k) in a curve, as depicted in Figure 3(a). In order to define such a curve in the sign database, there

has to be at least eight to ten postures to simulate that movement where the pause between those different poses is around two frames to have a smooth motion. The regular pose delay of 12 frames is intended to provide a pause in animation to identify the posture position, but for the poses of curves in RK and SK, a short delay was introduced. The change was done globally to use it with any required sign, and such curves have been tagged with a special tag named "ShortPose" in the sign definition, and user is allowed to adjust the "ShortPose" delay. In addition to the regular set of 61 character fingerspelling definitions, which includes ශ, ද, and ඩ SK characters, signs for ක්‍ර, ග්‍ර, ත්‍ර, ජ්‍ර, ද්‍ර, ප්‍ර, and ව්‍ර (kra, gra, thra, jra, dhra, pra, and vra) of derived consonants with RK are defined as additional fingerspelling signs with "ShortPose" tags in the sign database.

ALGORITHM 4: Algorithm for Animating Playlist of Gestures in the System

```
Input: Playlist of gestures
Output: 3D avatar is animated on screen.
for each sign in playlist do
    if the sign is a multi-posture sign then
        for each pose of the sign
            do
                Calculate posture coordinates of the next frame
                Update bone locations and display the frame
                if all the bones moved to final location of pose then
                    Signal animation complete
                End
            until animation is complete
            if current pose is a regular pose, then
                Wait for multi-posture wait time
            else if current pose is a ShortPose then
                Wait for ShortPose wait time
            end
        end
    else
        do
            Calculate posture of the next frame
            Update bone locations and display the frame
            if all the bones moved to final location of the pose then
                Signal animation complete
            end
        until animation complete
        Wait for single posture wait time
    end
end
```

4. FINDINGS, RESULTS, AND DISCUSSION

Before initiating the testing of fingerspelling Sinhala words, all the signs related to the SSL fingerspelling alphabet are modeled in the Blender animation software [27]. The actual orientation and presentation of the sign gestures published in the printed literature [2] are different from the orientation and placement of hands with the body when an actual interpreter fingerspells that character. For example, the difference in the orientation of the character "g" in the printed alphabet and actual signing is depicted in Figure 10. Therefore, videos of sign interpreters fingerspelling the characters are recorded, and adjustments are made to the fingerspelling signs of each character in this research.

Fig. 10. Orientations of actual interpreter's hand vs. printed literature.

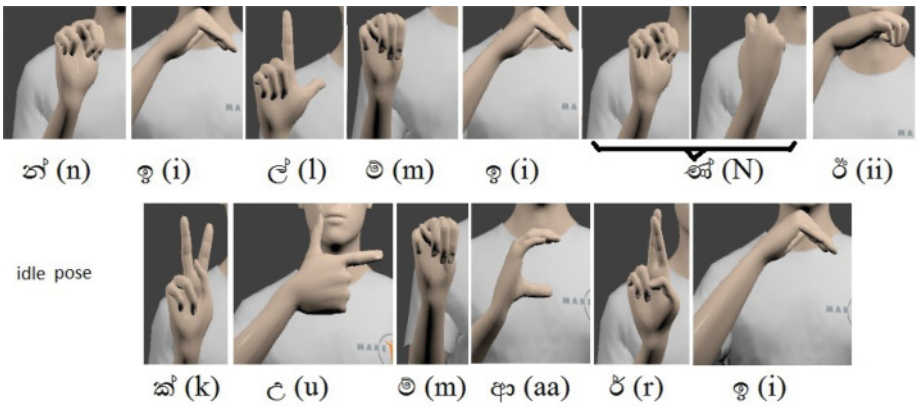

Fig. 11. Animation of the name Nilminii (row1), idle pose (row2), Kumaari (row2).

4.1. Animating Regular Words

The system is tested with regular words that do not have conversational signs and special variations for proper phonetic sequencing functionality. The inclusion of the idle sign to the sequence to demarcate words is also tested.

The animation sequence of the name නිල්මිණි කුමාරි (pronounced "Nilminii Kumaari") is animated by the system, as shown in Figure 11. The idle pose is inserted by the system to denote the space between words.

The last consonant ණ (n) of "Nilminii" is animated with two poses. Therefore, both single-pose and multi-pose characters of the above name are animated properly.

4.2. Animating Words with the Same Adjoining Letters

The adjustment of the pose position of the second letter of the two adjoining similar letters is tested for proper functionality. The following word ගත්තා (took), which is animated in the sequence of ග් + අ + ත් + ත් + ා (g + a + th + th + aa) in Figure 12, shows the displacement in the posture of the second ත් (th) character where the same character appears twice in succession.

4.3. Animating words with RK and SK Modifiers

Animating characters with RK and SK modifiers requires the utilization of different pose definitions. Figure 13 depicts how the word ක්‍රම (krama) with the meaning "methods," with the first character having an RK modifier, is animated.

Figure 14 depicts the use of ග (SK ga) as the second character of the word ගඟ (ganga) with the meaning "river."

In order to animate the horizontal curve shape of ක්‍ර (kra) in Figure 13, maintaining the pose of ක් (k) requires utilization of the wait time "ShortPose." The "ShortPose" tag is designed to wait for one or two frames (user-defined value) before moving to the

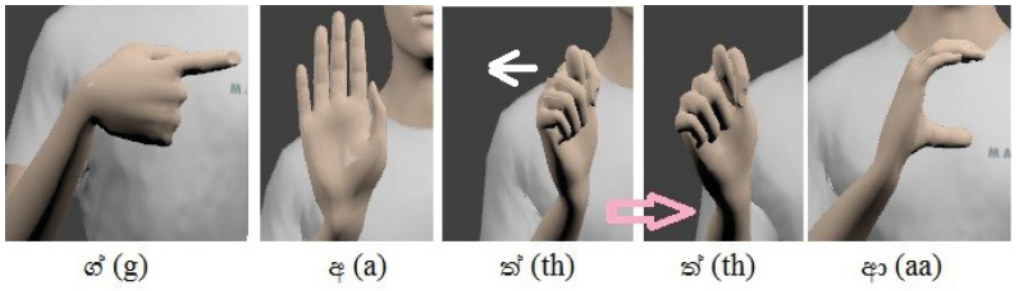

Fig. 12. Animation of word with two similar letters ත් (th) occurring together.

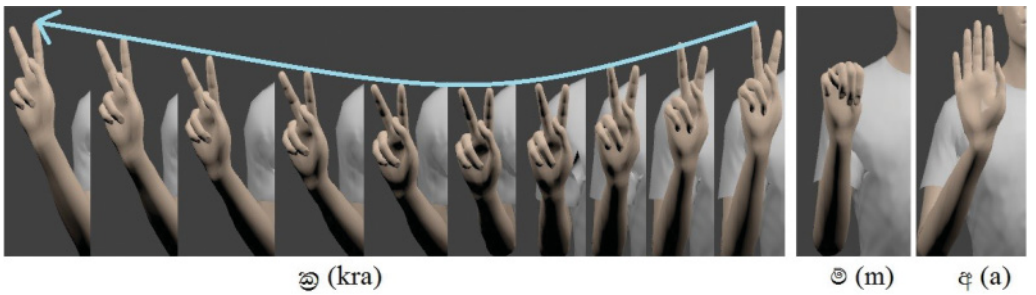

Fig. 13. Animation of the word ක්‍රම (krama), first letter is with RK modifier.

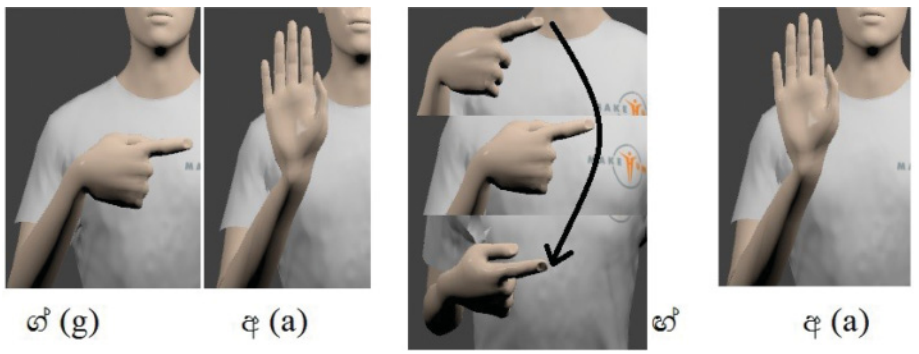

Fig. 14. Animation of the word ගඟ (ganga), last consonant with SK modifier.

next pose so that the curve shape is smoothly animated. A similar technique is used in Figure 14 to animate the vertical curve shape of ඟ් (SK g) while maintaining the pose of ග් (g). Both horizontal and vertical curves are animated properly.

4.4. Sequencing and Animating Numbers

Digit to number gesture conversion and Sinhala Unicode text to phonetic English text conversion are done in the VB.NET application. The generated sign gesture sequence is sent to the animation system as a normal sentence of words. For example, 25,000 is converted to “dheka paha dhahasa” (two five thousand) and animated as shown in Figure 15.

Similar numbers that appear in succession in decade values of a number are decoded by the animation system to change the pose coordinates of the second number. Those

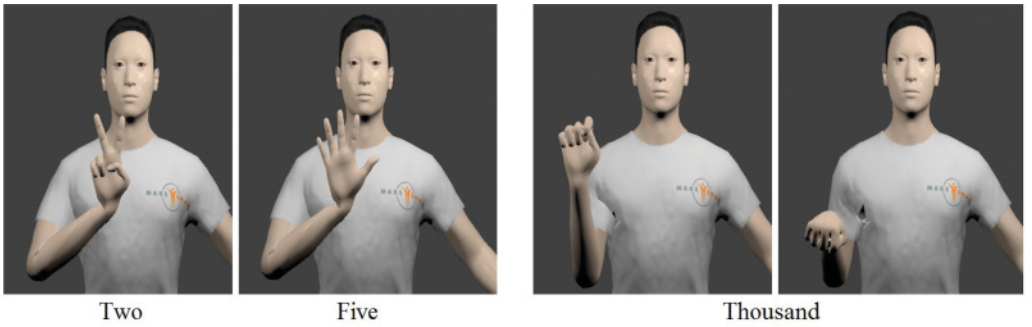

Fig. 15. Animation of the number 25,000.

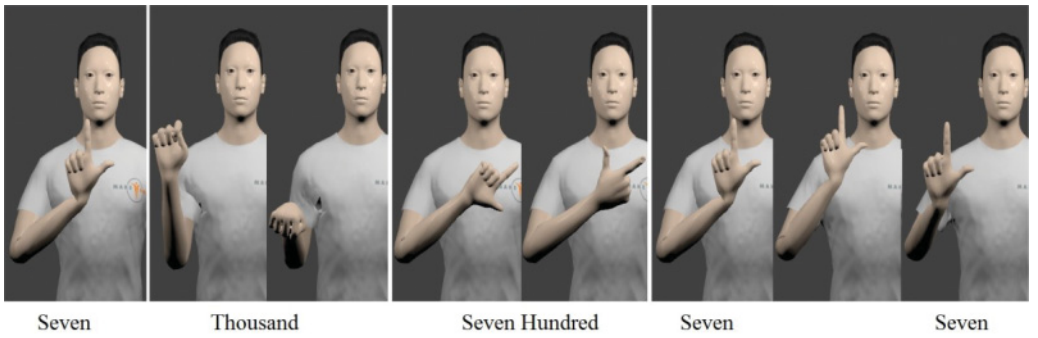

Fig. 16. Animation of 7,777 (seven thousand seven hundred seven seven).

numbers displayed in two places for the viewer to identify the two numbers occur in succession.The hand movement resembles a V shape when animated. For example, the number 7,777 is animated as හත, දහස, හත්සීය, හත, හත (seven, thousand, seven hundred, seven, seven) where the last digit 7 is animated with an extra pose and shifted pose placement away from the standard pose of 7, as shown in Figure 16.

The textual form of the numbers differs from the number sign sequence in terms of synonyms for numbers and the way in which the number is written in text. For example, we write the number "2" as "dheka," and we write the number "5" as "paha," but we write "25,000" as "visi pan dhahasa." The word "visi" must map to "dheka," and "pan" must map to "paha." Therefore, the sign gestures for number "2" and number "5" are duplicated as the "visi" and "pan" signs, respectively. Hence, when "visi pan dhahasa" comes in for animation, it will be properly animated as "dheka paha dhahasa." There is another variation when we write "visi pas dhenek" (i.e., "25 people"). Then, the number "5" is represented as "pas"; hence, the number sign "5" is also duplicated as the "pas" sign. Such textual variations for writing numbers in Sinhala are handled by introducing duplicate signs to the sign database.

4.5. Evaluation of the System

The multi-line sign definition was changed to accommodate the variation for RK and SK characters and led us to maintain two types of pose delays (regular, ShortPose) instead of one pose delay used in our previously published work in [3]. That change was done globally, making it easier to define any sign gesture requiring movements in curves. The fingerspelling and number signs of the system was initially modeled by

Table III. Fingerspelling Identification Results of a Sample of 10 Users

Word	Meaning	Phonetic English decoded character tags of the word separated by spaces	First attempt identification count	Mistaken characters	Avg. human-like rating
අධ්‍යාපනය	Education	a zdh w qx y axa p a n a y a	9	ධ්‍යා	7.1
බහුශ්‍රුත	Well-heard	b a h u zri txh a	9	ශ්‍ර	7.9
අර්ථසාධක	Provident / wellbeing	a r w zth a s axa zdh a k a	10	-	8.1
බෙහෙවර්ණ	Multi-color	v zaik v a r w zn a	8	බෙ	7.1
මනඃකල්පිත	Fictional	m a n a zkf k a l w p i txh a	8	නඃ	7.6
භාෂාව	Language	zb axa zsh axa v a	10	-	8.3
විශාරද	Expert	v i sh axa r a dh a	10	-	7.4
ගජකේසරී	Elephant hair	g a jh a k eze s a r ixi	4	ජ	6.3
උණ්ඩුකපුච්ඡය	Appendix (human)	u zn w d u k a p u ch w zch a y a	7	ච්ඡ	6.2
ප්‍රතිඵල	Results	p w qx r a txh i zp a l a	10	-	7.4
කෲර	Cruel	k zruu r a	10	-	8.1
ගෘහෝපකරණ	Household items	g zru h oxo p a k a r a zn a	8	ගෘ, හෝ	6.6
පොළඟා	Python	p o zl a zng axa	9	ළ	6.7
දෛවඥ	Astrologer	dh zaik v a jhcn a	8	ඥ	6.9
බණ්ඩාංක	Coordinates	zk a zn w d axa xonx k a	9	බ	8.2
හඬ	Sound	h a zndx a	10	-	7.9
ඇට්ටර	Stubborn	xcae t w t a r a	9	-	5.8
ගෑරුප්පුව	Fork	g aeae r u p w p u v a	10	-	6.5
සඤ්ඤක	A type of Sinhala characters	s a cn w cn a k a	2	ඤ	5.6
මුදලිදු	Mudliar (honorary))	m u dh a l i qndh u	9	ළු	6.6
කිඹුලා	Crocodile	k i xmb u l axa	8	ඹ	7.4
සජ්ඣායනා	Chant	s a jh w zjh axa y a n axa	6	ජ්ඣ	7.2
සාංඝික	Buddhist clergy's	s axa xonx zg i k a	10	-	8.2
ආරූඪ	Assume	axa r uxu zdx a	9	ඪ	6.8
ධර්මිෂ්ඨ	Devoted	zdh a r w m i zsh w zt a	9	ෂ්	7.4
නෞකාව	Ship	n zau k axa v a	1	නෞ	5.1
ෆැන්ටා	Fanta (brand name)	f xcae n w t axa	10	-	8.1
ඉඵු	Calling a dog	i xjhx u	8	ඵ	7.8
එසැනින්	In an instant	e s xcae n i n w	10	-	8.0
ඖෂධ	Medicine	xau zsh a zdh a	4	ඖ	6.8
ඓතිහාසික	Ancient	ai txh i h axa s i k a	8	ඓ	8.0
ව්‍යංජන	Curry	v aeae xonx xjhx a n a	9	ජ	7.0
Total			261		230.1
Percentage ((Total/320)*100)			81.56%	-	71.90%

referring to the SSL sign alphabet given in the literature and adjusted them for actual orientation by watching fingerspelling videos of actual SSL interpreters.

Then we have prepared screen cast videos 32 fingerspelled words generated from the system, which randomly contain all the characters and modifiers of the modern Sinhala alphabet except the characters ඩ, ළ and ෲ, which are not used in modern Sinhala words. We have shown those videos to 10 users without disclosing the words and asked them to identify each word and the results are given in Table III. Seven users of the sample are the teachers of deaf schools of Balangoda and Rathmalana in Sri Lanka, where they teach the regular school curriculum for Sinhala deaf children using SSL. The remaining three people are the leading SSL interpreters in Sri Lanka who validated fingerspelling and number signs modeled in this study. The user sample consisted of 3 females and 7 males whose ages are between 28–56 years.

Table IV. Individual Fingerspelling Character Identification Results of a Sample of 10 Users

A	B	C	D	E	F(%)	A	B	C	D	E	F(%)
අ	a	59	590	590	100.00	ඊ	zt	1	10	9	90.00
ආ	axa	15	150	150	100.00	ඤ	cn	2	20	3	15.00
ඈ	xcae	3	30	28	93.33	ජ	jh	2	20	2	10.00
ඇ	aeae	2	20	18	90.00	ශ	sh	1	10	8	80.00
ඉ	i	11	110	110	100.00	ච	ch	1	10	6	60.00
ඊ	ixi	1	10	10	100.00	ස	zg	1	10	10	100.00
උ	u	10	100	100	100.00	ඛ	zk	1	10	8	80.00
ඌ	uxu	1	10	7	70.00	භ	zb	1	10	7	70.00
එ	e	1	10	10	100.00	ද	dh	3	30	30	100.00
ඒ	eze	1	10	8	80.00	ඵ	zp	1	10	8	80.00
ඔ	o	1	10	10	100.00	ක	k	12	120	120	100.00
ඕ	oxo	1	10	5	50.00	ග	g	3	30	25	83.33
ඓ	ai	1	10	10	100.00	ට	t	3	30	28	93.33
ඖ	xau	1	10	4	40.00	ඩ	d	2	20	18	90.00
ඃ	zkf	1	10	10	100.00	න	n	8	80	80	100.00
ං	xonx	3	30	25	83.33	ප	p	8	80	80	100.00
ඤ	jhcn	1	10	10	100.00	බ	b	1	10	10	100.00
ඥ	xjhx	2	20	3	15.00	ම	m	3	30	30	100.00
ඬ	zndx	1	10	7	70.00	ය	y	4	40	40	100.00
ඳ	qndh	1	10	8	80.00	ර	r	11	110	107	97.27
ඡ	zch	1	10	5	50.00	ල	l	4	40	35	87.50
ඟ	zng	1	10	10	100.00	ව	v	8	80	78	97.50
ඣ	zjh	1	10	6	60.00	ස	s	8	80	70	87.50
ත	txh	4	40	40	100.00	හ	h	4	40	38	95.00
ඨ	zth	1	10	8	80.00	ෆ	f	1	10	10	100.00
ධ	zdh	4	40	33	82.50	ඍ	zri	1	10	8	80.00
ඹ	xmb	1	10	2	20.00	ු	zru	1	10	8	80.00
ෂ	zsh	3	30	28	93.33	ෛ	zaik	2	20	15	75.00
ඪ	zdx	1	10	8	80.00	ෞ	zau	1	10	0	0.00
ණ	zn	4	40	36	90.00	ූ	zruu	1	10	10	100.00
ළ	zl	1	10	8	80.00	-	-	-	-	-	-

A - character, B- phonetic tag, C- count in sample, D- total count for 10 users, E – correctly identified count, F – percentage of correct identification.

We have observed that there is no need to identify all the characters in a word in order to identify the word. Yet, mis-identification of rarely used characters and modifiers by SSL users is a common issue, which has led to the highest number of word identification errors on the first attempt for this 10-user sample. When the first and the last characters of a word are missed, it is difficult to predict the word. The above experiment ran with a frame rate of 9–12 frames per second, 12 frames per multi-posture sign, 5-frame standard pause between intermediate poses and 2-frame pause for short poses. With those settings, users have identified a word with an accuracy of 81.56% and they have given a rating of 71.9% for human-like animation of the system.

We have manually counted the user-indentified characters of the entire sample to calculate individual character identification rates, which are presented in Table IV.

The characters ඊ, ඤ, ඣ, ජ, ඥ, ඖ and the modifier ෞ had identification rates below 50%. SSL users have explained to us that we have modeled ඤ, ඣ, ජ to look similar but the character ඣ have right-to-left rotation, ජ have left-to-right rotation and ඤ have right-to-left rotation with a sliding to right to differentiate them while showing a same posture of small finger open and the rest of the fingers closed. Furthermore, they have told us that the poses of the character ඊ are derived from the poses of ම and බ characters, but we have modeled ඊ with the poses of ම and හ, which led to identification mistakes. The character ඥ had low identification rates because it had a SK modifier with the pose

of ජ character, which had a modeling error. The postures of ොෟ and ඖෟ are mis-identified by the users due to the missing posture of fully closed hand for ෙ. We have remodeled the erroneous fingerspelling signs identified by the SSL users after the testing.

Furthermore, users have pointed out that when a word has an AP character followed by an MP character of the same AP character, for example ච and ඡ in the word උණ්ඩුකපුවිඡය, animation does not properly demarcate the end of AP character ච and the beginning of the MP character ඡ because ච has a single posture and ඡ also start with the same posture. To solve this common issue, we have horizontally shifted the starting position of MP character to the right of AP character while maintaining the same posture in the fingerspelling database.

In the same experimental setup, we have animated 20 numbers that contain 40 number signs in SSL and produced screen cast videos of those numbers. We have shown those videos to the same set of users and their identification results are better for numbers as indicated in Table V.

The result set also indicates that the proper decoding and sequencing of numbers into respective number signs are taking place by the Algorithm 2, ranging from 1 digit to 12 digits including thousand, million, and billion signs.

To validate first part of Algorithm 3, we have sent the multiples of 11 (22–99) that are having the same digits from 2 to 9 through the algorithm. It correctly identified two adjoining numbers and introduced a dummy sign by changing the upper arm coordinates of the digit to move the arm backwards. While maintaining the pose of the first digit, right-hand upper-arm coordinates of the second digit are altered to show the pose of the second digit slightly right to the first digit. The calculated upper-arm coordinates of dummy pose and second digit and the original upper-arm coordinates of 2 to 9 are shown in Table VI.

The values added to the respective upper-arm X, Y, Z coordinates of the right hand to generate dummy pose and the second digit's starting pose are chosen to prevent the right hand crossing the face and the body of the avatar.

To validate the second part of Algorithm 3, one sample each from the types of different postures that can occur in SSL fingerspelling signs are processed. The original and shifted pose coordinates of those characters are given in Table VII.

The first type is the AP characters that have a single-posture fingerspelling sign. The second type is MP characters that are having a rotational second posture. The third type is also MP characters, which have a sliding movement as the second posture. The fourth type is the character ඤ්, which has both rotation and sliding movements. For all four types of these characters, the required shifted posture coordinates with shifts of $(-0.069, 0.169, -0.022)$ in X,Y,Z directions for the upper arm of the right hand are properly generated by the Algorithm 3. In Sinhala, there are 18 consonants having type 1 postures, 12 constants having type 2 postures, 2 constants having type 3 postures and 1 constant with type 4 postures.

The output row in Table VIII shows how Algorithm 4 processes the playlist of the sentence නීලා ගඟට මල් පහක් දැම්මේ ප්‍රාර්ථනා ඉටුවූ නිසාය meaning "Neela dropped five flowers to the river because she got her wishes fulfilled", which is phonetically decoded by Algorithm 1 as "nixilaxa gazngata malw pahakw dhxcaemwmeze pwqxraxarwzthanaxa ituvuxu nisaxaya" before passing to BGE.

Algorithm 4 treats every sign as a collection of postures and it reads the playlist and processes the postures of each sign sequentially. Moreover, it updates the positions of each bone related to the next frame in each iteration. Subsequently, the BGE displays the frame when the bone positions are updated. If all the bone positions match the coordinates of a posture, then Algorithm 4 signals as animation complete and proceed to the next posture. Wait time is the number of frames that Algorithm 4 keeps the current bone positions unchanged in order to ensure that the users recognize a posture. Hence,

Table V. Number Identification Results of a Sample of 10 Users with the Sequencing of Algorithm 2

Number	Output of the algorithm 2	Sequence of signs	A	B
0	binwdhuva	0	10	8.1
25	dheka paha	2, 5	10	8.2
116	sixiya dhahasaya	100, 16	10	8.2
314	txhunwsixiya dhaha hatxhara	300, 14	10	7.9
644	hayasixiya hatxhara hatxhara	600, 4, 4	10	7.6
718	hatxhwsixiya dhaha ata	700, 18	10	7.9
1055	eka dhahasa paha paha	1, thousand, 5, 5	10	7.8
2469	dheka dhahasa haxarasixiya haya navaya	2, thousand, 400, 6, 9	9	8.3
7019	hatxha dhahasa dhaha navaya	7, thousand, 19	10	8.0
10011	dhahaya dhahasa ekolaha	10, thousand, 11	10	7.9
30015	txhiha dhahasa pahalava	30, thousand, 15	9	8.4
654897	hayasixiya paha hatxhara dhahasa atasixiya navaya hatxha	600, 5, 4, thousand, 800, 9, 7	9	7.7
4510263	hatxhara miliyana panwsixiya dhahaya dhahasa dhesixiya haya txhuna	4, million, 500, 10, thousand, 200, 6, 3	9	7.8
8000017	ata miliyana dhaha hatxha	8, million, 17	9	8.2
50040020	panaha miliyana hatxhaliha dhahasa viswsa	50, million,40, thousand, 20	9	8.0
78210012	hatxha ata miliyana dhesixiya dhahaya dhahasa dholaha	7, 8, million, 200, 10, thousand, 12	9	8.0
356784471	txhunwsixiya paha haya miliyana hatxhwsixiya ata hatxhara dhahasa haxarasixiya hatxha eka	300, 5, 6, million, 700, 8, 4, thousand, 400, 7, 1	8	7.8
9970680560	navaya biliyana navasixiya hxcaetxhwtxhaeaeva miliyana hayasixiya asuxuva dhahasa panwsixiya hxcaeta	9, billion, 900,70, million, 600, 80, thousand, 500, 60	7	7.6
40000090013	hatxhaliha biliyana anuxuva dhahasa dhahatxhuna	40, billion, 90, thousand, 13	8	7.7
999999999999	navasixiya navaya navaya biliyana navasixiya navaya navaya miliyana navasixiya navaya navaya dhahasa navasixiya navaya navaya	900, 9, 9, billion, 900, 9, 9, million, 900, 9, 9, thousand, 900, 9, 9	8	7.9
Percentage (%)			92.0	79.5

A - First attempt full number identification count out of 10 users, B- Average human like rating by 10 users.

Algorithm 4 waits for a number of frames based on the pose type (single, multi-posture, or shortpose) defined in the each posture definition.

The texture/polygon detail of the 3D avatar, complexity of logical processing, and the complexity of postures of fingerspelling/numbers have led to computational challenges in 3D avatar-based animation systems developed for most sign languages. We have also observed a slowdown of animation speed of our prototype on computers that produce an average frame-rate of less than 7 frames per second when using the preferred settings of 12 frames per multi-posture sign, 5-frame standard pause between intermediate poses and 2-frame pause for short poses. Lowering the number of frames required to complete

Table VI. Right-Hand Upper-Arm Coordinates of Poses when Same Number Occurring Twice in the Decades

Digit	Coordinates of right-hand upper arm in db for the first digit*			Right-hand upper-arm coordinates of the dummy pose*			Right-hand upper-arm coordinates of the second digit*		
	X	Y	Z	X	Y	Z	X	Y	Z
2	0.274	−0.402	0.048	0.324	−0.352	0.010	0.205	−0.233	0.026
3	0.261	−0.400	0.050	0.311	−0.350	0.012	0.192	−0.231	0.028
4	0.263	−0.394	0.044	0.313	−0.344	0.006	0.194	−0.225	0.022
5	0.266	−0.405	0.051	0.316	−0.355	0.013	0.197	−0.236	0.029
6	0.268	−0.407	0.050	0.318	−0.357	0.012	0.199	−0.238	0.028
7	0.259	−0.398	0.052	0.309	−0.348	0.014	0.190	−0.229	0.030
8	0.276	−0.404	0.047	0.326	−0.354	0.009	0.207	−0.235	0.025
9	0.266	−0.396	0.048	0.316	−0.346	0.010	0.197	−0.227	0.026
C	–	–	–	0.050	0.050	−0.038	−0.069	0.169	−0.022

*W value is unchanged, Row C – difference from the first digit coordinates.

Table VII. Types of Fingerspelled Signs and Their Shifted Movements for Two Consecutive Occurrences

Poses	Ch	Pose	Coordinates of right hand upper arm in db for the first character*			Shifted right-hand upper-arm coordinates of the second character*		
			X	Y	Z	X	Y	Z
One	ඩ්	1	0.158	−0.129	0.298	0.089	0.040	0.276
Two (movement to second pose is a rotation)	ණ්	1	0.150	−0.124	0.292	0.081	0.045	0.270
		2	0.150	−0.124	0.292	0.081	0.045	0.270
Two (movement to second pose is a slide)	ශ්	1	0.154	−0.120	0.295	0.085	0.049	0.273
		2	0.337	−0.038	0.280	0.268	0.131	0.258
Multiple (with Rotation + Slide)	ඤ්	1	0.164	−0.145	0.277	0.095	0.024	0.255
		2	0.164	−0.145	0.277	0.095	0.024	0.255
		3	0.164	−0.145	0.277	0.095	0.024	0.255
		4	0.065	0.042	0.255	−0.004	0.211	0.233
C			–	–	–	−0.069	0.169	−0.022

*W value is unchanged, Row C – difference from the first character coordinates.

a multi-posture sign effectively increased the animation speed of our prototype, but it reduces the smoothness of animations because of the large bone position displacements per frame.

The prototype of this study allows a potential user to adjust the number of frames to complete animation of a single posture, multi-posture and fingerspelling sign. Moreover, it also allows changing the number of frames required for different types of pauses. This flexibility provides users to achieve sufficient signing speed for the 3D avatar depending on the processor speed and graphics capabilities of their computers.

4.6. Summary of Findings and the Limitations of the System

The aim of this research is to document, model and sequence entire finger spelling alphabet of SSL and the complete set of number signs that are required to animate full numbers up to billions. It is achieved and the modeled signs of the system are tested with the real users for the ability to recognize signs that are shown in a sequence.

The second objective is to support all the variations required for a continuously animated SSL fingerspelling by this system as if a human interpreter performs them. Algorithm 2 does the necessary on-the-fly adjustments to postures of same adjoining characters and numbers. The introduction of the “ShortPose” delay helped to model the signs for SK and RK characters properly to sequence them with Algorithm 4. Our testing indicated that the users gave ratings of 71.9% for fingerspelled words and

Table VIII. Animation Sequence of the Sentence "නීලා ගහට මල් පහක් දැම්මේ ප්‍රාර්ථනා ඉටුවූ නිසාය"

	Ch	Ch	Ch	Ch	Ch	Ch	Ch	Ch
Seq. of chars	න්	ී	ල්	ා	sp			
Play list	n	ixi	l	axa	id			
No. of poses	1	1	1	1	1			
Output order	n,S	ixi,S	l,S	axa,S	id,S			
Seq. of chars	ග්	අ	හ්	අ	ට	අ	sp	
play list	g	a	zng	a	t	a	id	
No. of poses	1	1	8	1	1	1	1	
Output order	g,S	a,S	zng1,SH,.., zng8,SH	a,S	t,S	a,S	id,S	
Seq. of chars	ම	අ	ල්	sp	පහක්	sp		
play list	m	a	l	id	paha	id		
No. of poses	1	1	1	1	1	1		
Output order	m,S	a,S	l,S	id,S	5,S	id,S		
Seq. of chars	ද්	ැ	ම	ම	ේ	sp		
play list	dh	xcae	m	sf-m	eze	id		
No. of poses	1	1	1	1	1	1		
Output order	dh,S	xcae,S	m,S	sf-m,S	eze,S	id,S		
Seq. of chars	ප්‍ර	ා	ර්	ථ්	අ	න්	ා	sp
play list	pwqxr	axa	r	zth	a	n	axa	id
No. of poses	10	1	1	2	1	1	1	1
Output order	pwqxr1,SH,.., pwqxr10,SH	axa,S	r,S	zth1,M, zth2,M	a,S	n,S	axa,S	id,S
Seq. of chars	ඉ	ට	උ	ව්	ූ	sp		
play list	i	t	u	v	uxu			
No. of poses	1	1	1	1	1	1		
Output order	i,S	t,S	u,S	v,S	uxu,S	id,S		
Seq. of chars	න්	ඉ	ස්	ා	ය්	අ		
play list	n	i	s	axa	y	a		
No. of poses	1	1	1	1	1	1		
Output order	n,S	i,S	s,S	axa,S	y,S	a,S		

sp- space, id - idle pose, sf-m - shifted posture of m, S – wait time for single posture, M – wait time for multi posture, SH – wait time for short posture, Ch – character.

79.5% for numbers for human-like animation capability of the system. The first-time recognition rate for fingerspelled words stands at 81.5% and 92% for numbers.

Furthermore, the design of SSL fingerspelling signs composed of a single posture for vowels and AP characters while rotation or vertical slide denotes MP characters with an exception of sign for the character ඤ්, which has both rotation and a slide. To indicate the same character appearing twice in succession, sliding movement of the second character to the right is the commonly adopted technique.

Videos of the human-centered experiments of this research are captured on a laptop with a core i5 processor at the speed of 2.3Ghz, 4GB RAM and achieved frame rate was in a range of 9–12 frames per second. The users are allowed to reduce the number of frames per sign, frames per pause to shorten the time to complete an animation of a gesture. This will speed up the animation on slow computers, but the animation will not be smooth due to the large displacement of each bone required per frame.

As limitations, this prototype does not have single fingerspelling signs to animate conjuncts such as ෂ or ඩ except the conjuncts that are directly listed in the Sinhala

alphabet. Another limitation is that the non-inclusion of the "repaya" modifier, which is used to shorten the words such as වර්ණ to වණ් by removing the character ර්. However, we have non-exhaustively observed that our prototype animates such occurrences as their longer versions by skipping ZWJ because the convention of writing ක්‍ෂ is ක් + ZWJ + ෂ, ඬ as න් + ZWJ + ධ and ණ් as ර් + ZWJ + ණ. The prototype also does not cover sequencing and animation of fractions, decimals, equations, arithmetic symbols related to numbers, which needs further work.

5. CONCLUSIONS

This article presents a successful implementation of an SSL fingerspelling and number signing system that interprets Sinhala text in SSL using a 3D avatar. The undocumented domain knowledge of different signing variations of SSL fingerspelling was gathered and implemented. The variations implemented in the system make the signing process of the 3D avatar synonymous to the performance of an actual human interpreter who does fingerspelling. Hence, this system can be used as an educational tool to learn/teach number signing and fingerspelling or to perform live interpretation tasks using the Sinhala fingerspelling alphabet.

ACKNOWLEDGMENTS

We acknowledge the feedback given by Mr. M. R. Premasiri, principal of the deaf school, Balangoda, Mr. J.C. Delmar, senior teacher of the deaf school Rathmalana and Mrs. K.L.C.N. Dias, sign interpreter of the ministry of social empowerment and welfare, Sri Lanka for this research. We wish to thank the teachers of the deaf school, Balangoda and the deaf school, Rathmalana for participating in the fingerspelled word and number identification experiments of this research. We also acknowledge the support of Dr. J. Wattewidanage, Mr. J.P.P. Tharanga, and the technical crew of the video production studio at the Center for Education Technology and Media (CETMe) of the Open University of Sri Lanka.

REFERENCES

[1] M. Ranaweera Premasiri, H. A. D. K. N. Hettiarachchi, J. C. Delmar, P. H. S. Dias, G. S. Mallika, W. G. S. A. K. Rajapaksha, A. N. Jayasekara, S. M. W. Samarakoon et al. (Eds.). 2015. *Sri Lankan Sign Dictionary: Sinhala Language.* National Institute of Education, Maharagama, Sri Lanka.

[2] Adam Stone (Ed.). 2007. *An Introduction to Sri Lankan Sign Language*. Karunaratne and Sons, Mathara, Sri Lanka. Retrieved August 22, 2014 from http://www.rohanaspecialschool.org/wp-content/uploads/2010/08/AnIntroductionToSriLankanSignLanguage.pdf.

[3] M. Punchimudiyanse and R. G. N. Meegama. 2015. 3D Signing avatar for sinhala sign language. In *2015 IEEE 10th International Conference on Industrial and Information Systems (ICIIS)*, Peradeniya, Sri Lanka. 290–295. DOI:10.1109/ICIINFS.2015.7399026

[4] M. Punchimudiyanse and R. G. N. Meegama. 2015. Unicode Sinhala and phonetic English bi-directional conversion for Sinhala speech recognizer. In *2015 IEEE 10th International Conference on Industrial and Information Systems (ICIIS), Peradeniya, Sri Lanka*. 296–301. DOI:10.1109/ICIINFS.2015.7399027

[5] Asanka Wasala and Kumudu Gamage. 2005. Research report on phonetics and phonology of Sinhala. LTRL, University of Colombo School of Computing. Retrieved May 15, 2014 from http://www.columbia.edu/∼kf2119/SPLTE1014/Day%203%20slides%20and%20readings/SinhalaPhoneticsand Phonology.pdf.

[6] Adam Stone. 2007. Fingerspelling in Sinhala. (April 2007). Retrieved June 05, 2014 from http://www.foundinceylon.com/2007/04/16/fingerspelling-in-sinhala.

[7] Unicode Inc. 2015. Sinhala Range: 0D80-0DFF, The Unicode Standard, version 8.0 (2015). Retrieved January 15, 2016 from http://unicode.org/charts/PDF/U0D80.pdf.

[8] F. Solina, S. Krapez, A. Jakic, and V. Komac. 2000. Multimedia dictionary and synthesis of sign language. In *Design Management of Multimedia Information Systems: Opportunities and Challenges*, Syed Mahbubur Rahman (Ed.). IDEA Group Publishing, United States, 268–281.

[9] Ze-Jing Chuang, Chung-Hsien Wu, and Wei-Sheng Chen. 2006. Movement epenthesis generation using NURBS-Based spatial interpolation. *IEEE Transactions on Circuits and Systems for Video Technology*. 16, 11, (Nov. 2006), 1313–1323. DOI:10.1109/TCSVT.2006.883509

[10] Ru Wang, Dehui Kong, Lichun Wang, and Baocai Yin. 2009. Gestures' smooth motion for chinese sign language synthesis based on video joining. In *International Conference on Computational Intelligence and Software Engineering*. Wuhan, China, 1–5.

[11] Eleni Efthimiou and Stavroula-Evita Fotinea. 2007. GSLC: Creation and annotation of a greek sign language corpus for HCI. In *International Conference on Universal Access in Human-Computer Interaction*. Springer, Berlin, 657–666.

[12] Selan R. dos Santos, Leonardo Bezerra, Antonino A. Feitosa Neto, and Silvano Maneck Malfatti. 2009. FAITH: A desktop virtual reality system for fingerspelling. In *Proceedings of the XI Symposium on Virtual and Augmented Reality (SVR 2009)*. Porto Alegre, Brazil, 189–198.

[13] Alexis Heloir, Sylvie Gibet, Franck Multon, and Nicolas Courty. 2005. Captured motion data processing for real time synthesis of sign language. In Lecture Notes in Computer Science (vol. 3881). Springer, Berlin, 168–171.

[14] Charly Awad, Nicolas Courty, Kyle Duarte, Thibaut Le Naour, and Sylvie Gibet. 2009. A combined semantic and motion capture database for real-time sign language synthesis. In *International Workshop on Intelligent Virtual Agents*. Springer, Berlin, 432–438.

[15] Nicoletta Adamo-Villani. 2008. 3D Rendering of american sign language finger-spelling: A comparative study of two animation techniques. *International Journal. of Human and Social Sciences* 3, 4, 314–319.

[16] Matt Huenerfauth. 2006. *Generating American Sign Language Classifier Predicates for English-to-ASL Machine Translation*. Ph.D. Dissertation. University of Pennsylvania, USA.

[17] Nicoletta Adamo-Villani and Gerardo Beni. 2004. Automated fingerspelling by highly realistic 3D animation. *British Journal of Educational Technology* 35, 3, (May 2004), 345–362. DOI: http://dx.doi.org/10.1111/j.0007-1013.2004.00393.x

[18] Mary Jo Davidson. 2006. PAULA: A computer-based sign language tutor for hearing adults. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems (ITS 2006): Teaching with Agents, Robots, and NLP*. Jhongli, Taiwan.

[19] Rosalee Wolfe, Mary Jo Davidson, John McDonald, and Carrie Frank. 2007. Using an animation-based technology to support reading curricula for deaf elementary school children. In *Proceedings of the 22nd Annual International Technology & Persons with Disabilities Conference*. Los Angeles, CA, USA.

[20] Uwe Ehrhardt, B. Davies, N. Thomas, M. Sheard, J. Glauert, R. Elliott, J. Tryggvason, T. Hanke et al. 2004. *The eSIGN Approach*. Technical Report on Deliverable D7-2 for eSiGN: Essential sign language information on government networks project. Retrieved July 10, 2014 from http://www.visicast.cmp.uea.ac.uk/Papers/eSIGNApproach.pdf.

[21] Desmond Eustin van Wyk. 2008. *Virtual Human Modelling and Animation for Real-time Sign Language Visualisation*. Master's thesis. University of the Western Cape, Cape Town, South Africa.

[22] Sarah Ebling, R. Wolfe, J. Schnepp, S. Baowidan, J. McDonald, R. Moncrief, S. Sidler-Miserez, K. Tissi et al. 2015. Synthesizing the finger alphabet of swiss german sign language and evaluating the comprehensibility of the resulting animations. In *Proceedings of the 6th Workshop on Speech and Language Processing for Assistive Technologies (SLPAT)*. Dresden, Germany, 10–15.

[23] Inês Almeida, Luísa Coheur, and Sara Candeias. 2015. From european portuguese to portuguese sign language. In *Proceedings of the 6th Workshop on Speech and Language Processing for Assistive Technologies (SLPAT)*. Dresden, Germany, 140–143.

[24] Matt Huenerfauth. 2008. Evaluation of a psycholinguistically motivated timing model for animations of american sign language. In *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility (Assets'08)*. ACM, New York, 129–136. DOI: http://dx.doi.org/10.1145/1414471.1414496

[25] Michael Kipp, Alexis Heloir, and Quan Nguyen. 2011. Sign language avatars: Animation and comprehensibility. In *Intelligent Virtual Agents*. Springer, Berlin, 113–126.

[26] Manuel Bastioni, J. Hauquier, J. Palmius, T. Larsson, A. Pinto, R. Baer, F. Grobbelaar, A. Mignon et al. 2014. MAKEHUMAN: Open source tool for making 3d characters, version 1.02. (2014). Retrieved September 11, 2014 from http://download.tuxfamily.org/makehuman/releases/1.0.2/makehuman-1.0.2-win32.zip.

[27] Blender Foundation. 2014. BLENDER: Free and Open Source 3D Creation Suite, version 2.72b. (2014). Retrieved October 30, 2014 from http://download.blender.org/release/Blender2.72/blender-2.72b-windows64.zip.

[28] Vincent Lepetit and Pascal Fua. 2005. *Monocular Model-based 3D Tracking of Rigid Objects*. K Now Publishers Inc., Netherlands, 11–14.

Received April 2016; revised February 2017; accepted May 2017