A Method to Accommodate Backward Compatibility on the Learning Application-based Transliteration to the Balinese Script

Gede Indrawan[1], I Gede Nurhayata[3], Sariyasa[4]
Department of Computer Science
Universitas Pendidikan Ganesha (Undiksha)
Singaraja, Indonesia

I Ketut Paramarta[2]
Department of Balinese Language Education
Universitas Pendidikan Ganesha (Undiksha)
Singaraja, Indonesia

***Abstract*—This research proposed a method to accommodate backward compatibility on the learning application-based transliteration to the Balinese Script. The objective is to accommodate the standard transliteration rules from the Balinese Language, Script, and Literature Advisory Agency. It is considered as the main contribution since there has not been a workaround in this research area. This multi-discipline collaboration work is one of the efforts to preserve digitally the endangered Balinese local language knowledge in Indonesia. The proposed method covered two aspects, i.e. (1) Its backward compatibility allows for interoperability at a certain level with the older transliteration rules; and (2) Breaking backward compatibility at a certain level is unavoidable since, for the same aspect, there is a contradictory treatment between the standard rule and the old one. This study was conducted on the developed web-based transliteration learning application, BaliScript, where its Latin text input will be converted into the Balinese Script output using the dedicated Balinese Unicode font. Through the experiment, the proposed method gave the expected transliteration results on the accommodation of backward compatibility.**

Keywords*—Backward compatibility; Balinese Script; learning application; transliteration

I. INTRODUCTION

As one of the diversity of local language knowledge in Indonesia, the endangered Balinese Script transliteration knowledge [1]–[3] raises concerns for the preservation. The Bali Government has already conducted the preservation efforts through the Bali Governor Regulation [4], [5] and strengthen them with the Bali Governor Circular Letter [6]. These efforts make the Balinese Language, including its Balinese Script transliteration knowledge, running as a mandatory local subject from elementary school to senior high school in Bali Province.

Multiple approaches other than the governmental approach should strengthen the preservation effort and should have a greater impact. This research joined the effort through the technological approach by multi-discipline collaboration between Computer Science and Language discipline. It proposed a method to accommodate backward compatibility on the learning application-based transliteration to the Balinese Script. This work has never been conducted yet and applied to the previous works that were still based on the older transliteration rules (for short, the older rules) from The Balinese Alphabet document [1]. It exposes the backward compatibility method to accommodate the standard transliteration rules (for short, the standard rules) from the Balinese Language, Script, and Literature Advisory Agency [7]. This Bali Province government agency [4] carries out guidance and formulates programs for the maintenance, study, development, and preservation of the Balinese Language, Script, and Literature.

This study was conducted on the developed web-based transliteration learning application, BaliScript, for further ubiquitous Balinese Language learning since the proposed method reusable for the mobile application [8], [9]. It also advances the previous work by (1) accommodating special words [1], [10] through a certain table structure in the database rather than hard-coding them in the application code; (2) making use of the more developed and the less bug of Noto Serif Balinese (NSB) font[2, 3] [11] to represent the Balinese Script rather than the Noto Sans Balinese font[4]. The NSB font is a dedicated Balinese Unicode font which makes it recognized on the computer system including mobile devices and makes the proposed method reusable on the mobile application; and (3) improving the learning experience on the application, that uses this method, through the addition of the Indonesian and English translation for the transliterated word (see the next Fig. 3). Overall, all of those advances are considered as the contribution of this work.

This paper is organized into several sections, i.e. Section I (Introduction) states the problem background related to the transliteration to the Balinese Script; Section II (Related Works) describes the related works in the area of the transliteration to the Balinese Script and its backward compatibility aspect; Section III (Research Method) exposes the supporting algorithm, the implementation, and the testing

This work was supported by the Indonesian Ministry of Education, Culture, Research, and Technology through the Applied Research Grant.

[1] The Balinese Alphabet, http://www.babadbali.com/aksarabali/alphabet.htm (Retrieved June 16, 2021)

[2] Balinese Unicode Table, http://unicode.org/charts/PDF/U1B00.pdf (Retrieved June 16, 2021)

[3] Google Noto Serif Balinese, https://github.com/googlefonts/noto-fonts/blob/master/unhinted/ttf/NotoSerifBalinese/NotoSerifBalinese-Regular.ttf (Retrieved June 16, 2021)

[4] Google Noto Fonts, https://www.google.com/get/noto/#sans-bali (Retrieved June 16, 2021)

of the proposed method; Section IV (Result and Analysis) covers the analysis of the testing result; and finally, Section V (Conclusion) consists of important conclusion and future work points.

II. Related Works

Several related works on Latin-to-Balinese Script transliteration were conducted on the previous works [10], [12]–[20]. All of those were still based on the older rules from The Balinese Alphabet document, except [20]. Displaying Balinese Script output on those previous research was done by non-dedicated Balinese Unicode fonts (i.e. Bali Simbar[5] and Bali Simbar Dwijendra [21]) and dedicated Balinese Unicode font[2] [11] (i.e. Noto Sans Balinese and Noto Serif Balinese). The Bali Simbar (BS) font was utilized in [12] and gave a relatively good accuracy result on testing cases from The Balinese Alphabet document. It was also utilized in the developed robotic system that writes the Balinese Script from the Latin text input [13], and on the exploration of the line-break handling during the transliteration [14]. The Bali Simbar Dwijendra (BSD) font, as the improvement of the BS font, was utilized in [15] with additional testing cases from the Balinese Script dictionary [7] to the same testing cases on [12]. It was also utilized in the exploration of the mathematical expression transliteration [16]. Ten transliteration lessons were also learned by using this font on the other testing data [17]. The Noto Sans Balinese font was utilized in [10] with the same testing cases in [12] and gave a relatively good accuracy result. It was also utilized in the developed robotic system that writes Balinese Script from the Latin text input [18]. Extensive accuracy analysis on the developed algorithm [10] was done in [19] for future improvement. the Noto Serif Balinese font was utilized in [20] for the unavoidable affixed words that need to be transliterated.

The other side of transliteration related to the Balinese Script-to-Latin transliteration that utilized the GNU Optical Character Recognition (OCR), i.e. Ocrad[6] [22]. This research was limited only to the basic syllable recognition (see The Balinese Alphabet document) from the Balinese Script image that was based on the glyph shape of the Bali Simbar font. For advancing functionality and mobile adoption for ubiquitous learning, the utilization of the Tesseract[7] OCR was conducted that needs several future improvements [23].

III. Research Method

The proposed method to accommodate backward compatibility on the transliteration to the Balinese Script covers two aspects related to the older transliteration rules from The Balinese Alphabet document. Those two aspects, i.e. (1) Backward compatibility allows for interoperability at a certain level with the older rules; and (2) Breaking backward compatibility at a certain level is unavoidable since, for the same aspect, there is a contradictory treatment between the standard rule and the old one.

This section describes (1) the supporting algorithm of the proposed method; (2) the implementation on the BaliScript, which is the web-based transliteration learning application; and (3) the testing by using the updated testing cases of The Balinese Alphabet document to comply with the standard transliteration rules from the Balinese Language, Script, and Literature Advisory Agency [4], [7].

A. The Algorithm

The proposed method involves the NSB font with its dedicated Balinese Unicode Table [20]. The algorithm to accommodate backward compatibility on the transliteration to the Balinese Script covers two aspects, as described previously. The first aspect involves transliteration of the letter set MBC (Maintaining Backward Compatibility), i.e. the vowel “ĕ” (U+011B) with sound [ə] [24], “ö” (U+00F6) with the long sound of the vowel “ĕ”, the consonant na rambat “ņ” (U+0146) with sound [ŋa], sa sapa “ś” (U+015B) with sound [ʂa], sa saga “ş” (U+015F) with sound [ɕa], ta latik “ţ” (U+0163) with sound [ʈa], or its uppercase letter “Ĕ” (U+011A), “Ö” (U+00D6), the consonant “Ņ” (U+0145), “Ś” (U+015A), “Ş” (U+015E), “Ţ” (U+0162). The second aspect involves transliteration of the letter set BBC (Breaking Backward Compatibility), i.e. the vowel “e” (U+0065) or its uppercase letter “E” (U+0045) that has sound [e]. Noted that the uppercase letters were not the concern, since each of them has the same transliteration result as its counterpart lowercase letter.

Those two aspects should be handled by the proposed method. Fig. 1 shows the flowchart of the algorithm and uses regular expression [25], [26] on the implementation.

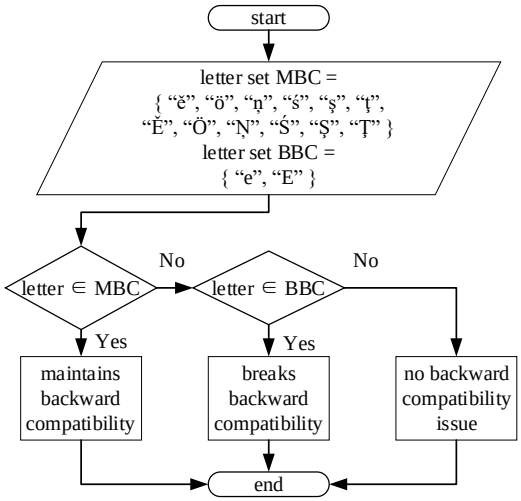


Fig. 1. The Flowchart of the Algorithm.

[5] Bali Simbar, http://www.babadbali.com/aksarabali/balisimbar.htm (Retrieved June 16, 2021)

[6] The GNU Ocrad OCR, https://www.gnu.org/software/ocrad/ (Retrieved June 16, 2021)

[7] Tesseract OCR, https://github.com/tesseract-ocr/ (Retrieved June 16, 2021)

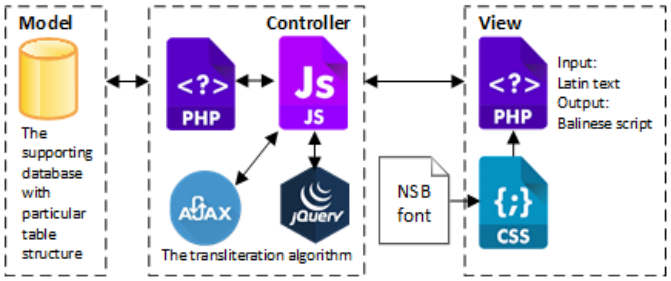
B. The Implementation

Fig. 2 (a) shows the Model-View-Controller (MVC) architecture [27]–[29] of the web-based transliteration learning application, BaliScript, that was used by the proposed method. The supporting database's table (Fig. 2 b) consists of records from the Balinese Script dictionary [7]. Fig. 3 shows the Indonesian and English translation of the example transliterated word for improving the learning experience on the application. As described previously, this feature is one of several advances as the contribution of this work. The BaliScript was constructed by Apache web server, MySQL database server, and PHP code combined with JavaScript code. This application was also used for the exploration of scriptio continua management in the previous work [30].

Fig. 3 (a) shows the View of the MVC, i.e. (1) the input view that uses the Select box[8]; and (2) output view that displays the transliteration result and other results from the closest similar words in the database where the similarity calculation is based on the Levenshtein distance [31], [32]. Fig. 3 (b) shows the transliteration output from the example homonym word [33], [34] at the similarity list by using AJAX-based switching (clicking on the word "USE" related to the certain word).

C. The Testing

The testing of the proposed method was conducted on the BaliScript, which was run on the Intel Core i7-4600U CPU @2.09GHz platform with 8 GB RAM and Windows 8 64-bit Operating System.



(a)

id	word	sword	special	idn	eng	…
…	…	…	…	…	…	…
1952	basa	bhāßa	1	bahasa	language	…
1953	basa	basa	0	bumbu	seasoning	…
…	…	…	…	…	…	…

(b)

Fig. 2. The BaliScript Implementation: (a) MVC Architecture; (b) Supporting Table in the Database

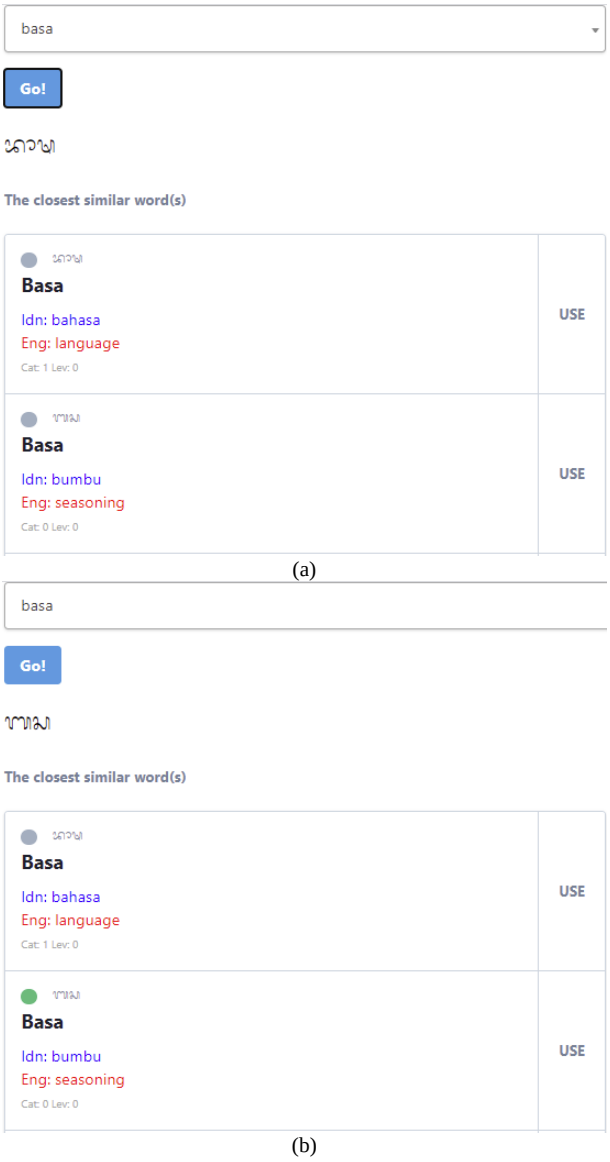


Fig. 3. The View of the BaliScript with Transliteration and Translation Result at the Same Time: (a) Output with the Closest Similar Words; (b) Output from the Example Homonym Word at the Similarity List.

IV. Result and Analysis

Table I shows the testing cases consist of sections of interest (the marked sections) related to the result of backward compatibility (see Fig. 4). Noted that the testing used the updated testing cases that comply with the standard transliteration rules from the Balinese Language, Script, and Literature Advisory Agency [7] rather than the original testing cases [10] that refer to The Balinese Alphabet document.

[8] Select2 box, https://select2.org (Retrieved June 16, 2021)

TABLE I. TESTING TRANSLITERATION CASES

No.	Case[a]	Case[b]	Remarks[c]
1	ha na ca ra ka da ta sa wa la ma ga ba nga pa ja ya nya Bakta K**a**la Pak**s**a Raka Cakra W**a**laka Krama	ha na ca ra ka da ta sa wa la ma ga ba nga pa ja ya nya Bakta K**ạ**la Pak**ṣ**a Raka Cakra W**ạ**laka Krama	Basic syllables and examples. Bring Time Force “Bigger brother” Disc “A non-priest” Member
2	Kād**e**p J**ĕ**ro Siya Kayu S**e**la Angklung Daitya Patūt Dw**ī**	Kād**e**p J**ĕ**ro Siya Kayu S**ē**la Angklung Daitya Patūt Dw**ī**	Vowel signs examples. Sold House Nine Wood Yam “Musical instrument” Giant “Should be” Two
3	a ā i ī u ū **e** ai o au Ak**ś**ara **Iś**wara Upacāra Eka Airlangga Ong OM	a ā i ī u ū **ē** ai o au Ak**ś**ara **Iś**wara Upacāra Eka Airlangga Ong OM	Independent vowels and examples. Alphabet “God’s name” Ceremony One “A Javanese King” “One holy letter” “Symbol of God”
4	rĕ rö lĕ lö Talĕr Kĕrĕng	rĕ rö lĕ lö Talĕr Kĕrĕng	Illegal combination of syllable - vowel signs and examples. Also “Eat a lot”
5	Pak Raman Pakraman Baglug Rubag lugu Briag	Pak Raman Pakraman Baglug Rubag lugu Briag	Semi vowels examples. Mr. Raman Membership Stupid “Naive Rubag” Laughter
6	**n**a dha tha **ṭ**a **ṣ**a **ś**a gha bha pha Ga**n**itri Garudha P**a**rtha Ja**ṭ**ayu Bhi**ś**ama **Ś**iwa Laghu	**n**a dha tha **ṭ**a **ṣ**a **ś**a gha bha pha Ga**n**itri Garudha P**ạ**rtha Ja**ṭ**ayu Bh**i****ś**ama **Ś**iwa Laghu	Akśara swalalita and examples. Chain “Big eagle” “Arjuna's alias” “A bird in Ramayana” Decree “God’s name” “Low tone in singing”
7	C**e**ngc**e**ng Bangkung Manah Kar**n**a Ka**p**al	C**ē**ngc**ē**ng Bangkung Manah Kar**n**a Ka**p**al	Sound killers examples. “Musical instrument” Pig Mind Ear Ship
8	Mang Siddham	Mang Siddham	Miscellaneous signs examples. “Holy letter” Perfect
9	Om Swastiastu Om **S**anti, **S**anti, **S**anti, Om	Om Swastiastu Om **S**anti, **S**anti, **S**anti, Om	Holy symbol Ongkara examples. “May God blesses you” “May peace be everywhere”
10	cha kha	cha kha	Miscellaneous syllables.
11	0 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5 6 7 8 9	The digits.
12	, . < .0. > >> : "	, . < .0. > >> : "	Punctuations.
13	i u **e** o ĕ ö pu phu Sekala sekal**ĕ** Samping Suk**ś**ma K**ś**atria Strī Smerti U**t**ama Dharma Tamblang	i u **ē** o ĕ ö pu phu Sekala sekal**ĕ** Samping Suk**ś**ma K**ś**atria Strī Smerti U**ṭ**ama Dharma Tamblang	Some variation of usages. Combination of independence vowel a kara with vowel signs Pairing of pa kapal with suku or suku ilut Romanization of the inherent sound: Real real Usage of pangangge akśara: Side “Thank you” Warrior Wife “Books of Vedha” Primary Religion “A village’s name”
14	hā nā cā rā kā dā tā sā wā lā mā gā pā yā **n**ā dhā thā **ṭ**ā **ṣ**ā **ś**ā ghā bhā	hā nā cā rā kā dā tā sā wā lā mā gā pā yā **n**ā dhā thā **ṭ**ā **ṣ**ā **ś**ā ghā bhā	Ligatures.
15	Bank P**e**mbangunan Da**e**rah Bali B**e** P**e** D**e** Bali Ba P**e** Da Bali Ba Pa Da Bali	Bank P**e**mbangunan Da**e**rah Bali B**ē**, P**ē**, D**ē**, Bali Ba, P**e**, Da, Bali Ba, Pa, Da, Bali	Abbreviations examples. Regional Development Bank Bali Be Pe De Bali Ba Pe Da Bali Ba Pa Da Bali
16	Ak**e**h ak**ś**aran**e**, 47, luir ipun: ak**ś**ara suara, 14, ak**ś**ara wianjana, 33, ak**ś**ara suara punika tal**ĕ**r dados pangangg**e** suara, tur madr**ĕ**w**e** suara kakalih, kawā**śt**anin: suara hr**ĕ**swa miwah dīrgha.	Ak**e**h ak**ś**aran**ē**, 47, luir ipun: ak**ś**ara suara, 14, ak**ś**ara wianjana, 33, ak**ś**ara suara punika tal**ĕ**r dados pangangg**e** suara, tur madr**ĕ**w**e** suara kakalih, kawā**śt**anin: suara hr**ĕ**swa miwah dīrgha.	Word boundaries and line break rules. Many of those letters, 47, i.e.: vowels, 14, consonants, 33, those vowels also become vowel signs, and have two sounds, each is called: sound hrĕswa and dīrgha.

[a] The original testing cases

[b] The updated testing cases that comply with the standard transliteration rules

[c] The Balinese Alphabet document

For example in case 2 of Table I, since the vowel "e" of the Balinese word "S**ē**la" (Yam) has sound [e] [24] for a certain meaning (the other "e", U+0065, with sound [ə] has a different meaning), to comply with the standard rule, the writing of that vowel should be changed to "ē". This condition breaks backward compatibility of the transliteration since the vowel "e" is a member of the letter set BBC (see The Algorithm section).

There are several sections of interest in Table I related to the testing result, i.e.:

- The bold underlined section on the original testing case shows a section of interest related to the backward compatibility of the transliteration.
- The bold underlined section on the updated testing case shows a section of interest that has backward compatibility where its transliteration result adheres to [7] and is the same as the transliteration result of the original testing case. This backward compatibility was achieved due to the process related to the algorithm.
- The bold dotted-underlined section on the updated testing case shows a section of interest that has a transliteration result that adheres to [7] but different from the transliteration result of the original testing case using The Balinese Alphabet document.
- The bold gray section on the updated testing case shows a section of interest that has broken backward compatibility by using different writing where its transliteration result adheres to [7] and the same to the transliteration result of the original testing case.
- The underline-across-space section on the updated testing case shows a section of interest that has a transliteration result that adheres to [7] and different from the transliteration result of the original testing case. This is because continuous (phrase or sentence) transliteration was used rather than word-by-word transliteration. If both updated and original testing cases use the same kind of transliteration then the result should be the same. It needs to be mentioned as a perspective that relatively was not related to backward compatibility. For example in case 2 of Table 2, the Balinese phrase "Kād**e**p J**ĕ**ro" (Sold House) has continuous transliteration result "ᬓᬵᬤᭂᬧᬚᭂᬭᭀ" adheres to [7] and different from word-by-word transliteration result "ᬓᬵᬤᭂᬧ ᬚᭂᬭᭀ". In continuous transliteration, the second word of "J**ĕ**ro" (House) has its consonant "J" was transliterated in appended form as "ᬳ᭄ᬚᭂ" (hanging below the regular form of consonant "p" of word "Kād**e**p") while its vowel "ĕ" was transliterated as a vowel sign (upper form). In word-by-word transliteration, the second word of "J**ĕ**ro" has its consonant "J" was transliterated in regular form as "ᬚᭂ" (positioned on the side after the sound killer *adeg-adeg* "᭄" that kill the inherent sound of consonant "p" of word "Kād**e**p") while its vowel "ĕ" was transliterated as a vowel sign (upper form).

Even though Balinese Script employs scriptio continua style [35], Fig. 4 shows its transliteration result in non-scriptio continua style (including preserved line breaks) which is possible to be generated for ease of visual analysis by the BaliScript learning application. This style was supported by the white-space[9] property of Cascading Style Sheets (CSS) that was set as *pre-line*. This kind of non-scriptio continua style has the same space and line break format as its Latin text input from the testing transliteration cases of Table I. It has a clear mapping between the input section of the Latin text (i.e. alphabet, syllable, word, or punctuation) and its related output section of the Balinese Script. That clear mapping was caused by the spaces and line breaks between those sections that were preserved by the transliteration algorithm [30].

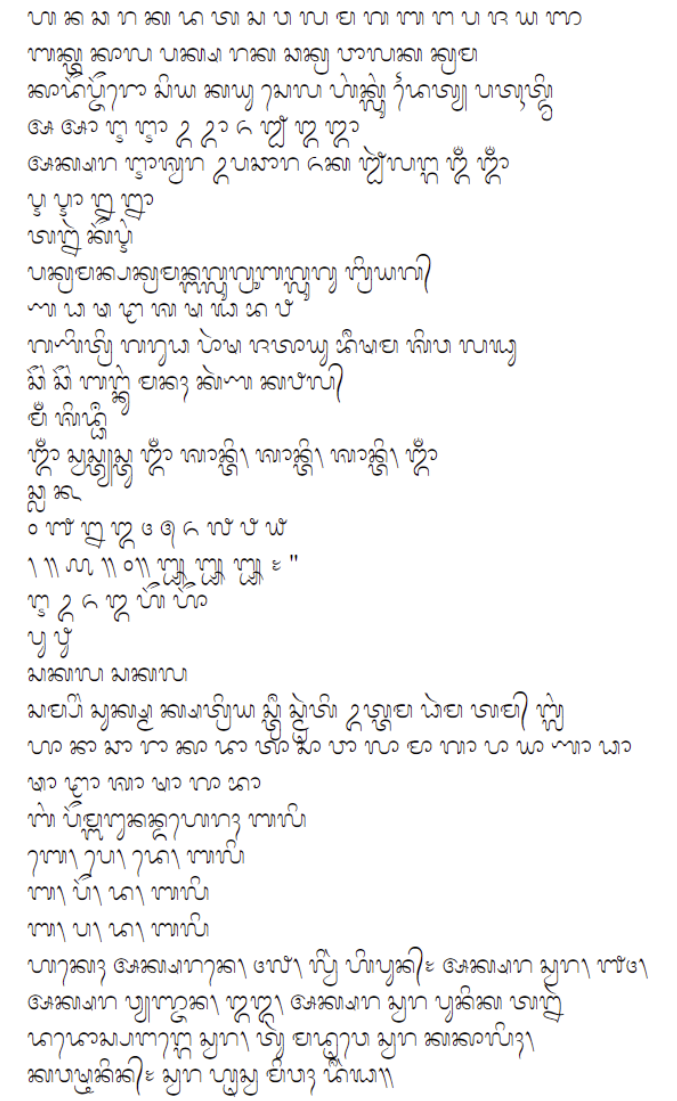


Fig. 4. The Balinese Script Transliteration Result with non-scriptio Continua Style, Including Preserved Line Breaks, from the BaliScript Learning Application.

[9] CSS white-space, https://www.w3schools.com/cssref/pr_text_white-space.asp (Retrieved June 16, 2021)

The backward compatibility analysis of the transliteration results in Fig. 4 was based on the marked sections in Table I. The algorithm maintains backward compatibility and on the other side unavoidably breaks backward compatibility to comply with the standard transliteration rules [7].

Related to maintaining backward compatibility, the bold underlined section on the updated testing case shows a section of interest where its transliteration result adheres to [7] and is the same as the transliteration result of the original testing case. For example, in case 2 of Table I, the Balinese phrase "Kādep Jĕro" (Sold House) has its continuous transliteration result "ᬓᬵᬤᭂᬧ᭄ᬚᭂᬭᭀ" that adheres to [7] and is the same as the continuous transliteration result from The Balinese Alphabet document (see the previous underline-across-space section).

The bold underlined sections, i.e. the vowel "e" (U+0065) of the word "Kād**e**p" and "ĕ" (U+011B) of the word "J**ĕ**ro" has the same sound [ə] [24] and should be transliterated the same by using "ᭂ" (Balinese vowel sign *pepet*, U+1B42). Their variant words from [7], i.e. "ad**e**p" (only available in its root word) and "j**e**ro" should be transliterated the same. These variances of words "Kād**e**p", "kad**e**p", "J**ĕ**ro", "j**e**ro", and others should be registered with their related same value in column "sword" of database's table (see Fig. 2 b) for the same transliteration result. This is the effort for maintaining transliteration backward compatibility along with other cases in Table I, i.e. word "Ak**ś**ara", "**Iş**wara", "Ga**ṇ**itri", "Bh**iś**ama", "**Ş**iwa", "Kar**ṇ**a", "**Ş**anti", "sekal**ĕ**", "Suk**ś**ma", "K**ś**atria", "P**e**mbangunan", "Da**e**rah", "Ak**e**h", "tal**ĕ**r", "pangangg**e**", "madr**ĕ**w**e**", "kaw**āśţ**anin", and "hr**ĕ**swa".

From those cases with bold underline marks, certain of those were also marked with bold dotted-underline since each of them has a transliteration result that adheres to [7] but different from the transliteration result of the original testing case. For example, the Balinese word "**Iş**wara" (God's name) and "Bh**iś**ama" (Decree), each in case 3 and case 6 of Table I, have their variant words from [7], i.e. "**Is**wara" and "b**is**ama" should be transliterated the same "ᬈᬱ᭄ᬯᬭ" and "ᬪᬷᬰᬫ" [7] but different from the transliteration result "ᬇᬱ᭄ᬯᬭ" (without vowel sign *tedung* "ᬵ") and "ᬪᬶᬰᬫ" (without vowel sign *ulu sari* "ᬷ"). This is a condition that should be taken care of by the effort for maintaining backward compatibility transliteration. Above that condition, these variances of word "**Iş**wara", "**Is**wara", "Bh**iś**ama", "b**is**ama", and others should be registered with their related same value in column "sword" of database's table (see Fig. 2 (b)) for the same transliteration result that adheres to [7].

From those cases with bold underline marks, certain of those were safe to associate its vowel "e" (U+0065) to the vowel "ē" (U+0113) through the database registration because of its sound [e] [24]. This condition was possible since no counterpart word has the vowel "e" (U+0065) with sound [ə]. This condition is related to the next testing cases with the bold gray section. For example, the Balinese word "Ak**e**h" (Many), in case 16 of Table I, has its variant words from [7], i.e. "ak**ē**h" should be transliterated the same "ᬳᬓᬾᬄ" [7]. As the exception to the standard rule [7] where the vowel "e" (U+0065) should be transliterated by using "ᭂ" (Balinese vowel sign *pepet*, U+1B42) while the vowel "ē" (U+0113) should be transliterated by using "᭄" (Balinese *adeg-adeg*, U+1B44), these variances of the word "Ak**e**h", and "ak**ē**h" should be registered with their related same value in column "sword" of database's table (see Fig. 2(b)) for the same transliteration result that adheres to [7]. This is a condition that should be taken care of by the effort for maintaining backward compatibility transliteration since by nature people write the word in the easiest way (write "e" rather than "ē"), including inputting text to the transliteration application.

Related to unavoidable breaking backward compatibility to comply with the standard transliteration rules [7], the bold gray section on the updated testing cases shows a section of interest that has broken backward compatibility by using different writing where its transliteration result adheres to [7] and the same to the transliteration result of the original testing case. For example, in case 2 of Table I, the Balinese word "S**ē**la" (Yam) with its vowel "ē" (U+0113) has its transliteration result "ᬲᬾᬮ" that adheres to [7] and is the same as the transliteration result from The Balinese Alphabet document (see the previous standard rules [7] where the vowel "e" and "ē", each should be transliterated by using vowel sign *pepet* and sound killer *adeg-adeg*). If using the Balinese word "S**e**la" with its vowel "e" (U+0065) from the original testing case, its transliteration result "ᬲᭂᬮ" does not adhere to [7] even though is the same as the transliteration result from The Balinese Alphabet document.

V. Conclusion and Future Work

A method to accommodate backward compatibility was proposed on the learning application-based transliteration to the Balinese Script. It covered two aspects related to considered sets of letters. The first aspect concerns the transliteration of a certain set of letters that causes backward compatibility to be maintained. The second aspect concerns the transliteration of a certain set of letters that causes backward compatibility to be broken unavoidably to comply with the standard rules from the Balinese Language, Script, and Literature Advisory Agency.

Acknowledgment

This work was supported by the Indonesian Ministry of Education, Culture, Research, and Technology through the multi-years Applied Research Grant.

References

[1] G. Indrawan, I. K. Paramarta, K. Agustini, and Sariyasa, "Latin-to-Balinese script transliteration method on mobile application: a comparison," Indones. J. Electr. Eng. Comput. Sci., vol. 10, no. 3, pp. 1331-1342, Jun. 2018.

[2] S. Karimi, F. Scholer, and A. Turpin, "Machine transliteration survey," ACM Comput. Surv., vol. 43, no. 3, pp. 1–46, Apr. 2011.

[3] K. Kaur and P. Singh, "Review of machine transliteration techniques," Int. J. Comput. Appl., vol. 107, no. 20, pp. 13–16, 2014.

[4] Bali Government, "Bali Government Regulation No. 3 on Balinese Language, Script, and Literature [Peraturan Pemerintah Bali No. 3 tentang Bahasa, Aksara, dan Sastra Bali]," 1992. https://bphn.go.id/data/documents/perda_3_1992.pdf (Retrieved June 16, 2021).

[5] Bali Government, "Bali Governor Regulation No. 80 on Protection and Usage of Balinese Language, Script, and Literature, also Organizing Balinese Language Month [Peraturan Gubernur Bali No. 80 tentang Pelindungan dan Penggunaan Bahasa, Aksara, dan Sastra Bali serta Penyelenggaraan Bulan Bahasa Bali]," 2018. https://jdih.baliprov.go.id/produk-hukum/peraturan/abstrak/24665 (Retrieved June 16, 2021).

[6] Bali Government, "Bali Governor Circular Letter No. 3172 Year 2019 about The Usage of Balinese Traditional Clothing and Balinese Script [Surat Edaran Gubernur Bali No. 3172 Tahun 2019 tentang Penggunaan Busana Adat Bali dan Aksara Bali]," 2019. https://jdih.baliprov.go.id/produk-hukum/peraturan/abstrak/24741 (Retrieved June 16, 2021).

[7] I. G. K. Anom et al., Balinese - Indonesian Dictionary with its Latin and Balinese Script [Kamus Bali - Indonesia Beraksara Latin dan Bali]. Denpasar: Bali Province, 2009.

[8] G. J. Hwang, C. C. Tsai, and S. J. H. Yang, "Criteria, strategies and research issues of context-aware ubiquitous learning," J. Educ. Technol. Soc., vol. 11, no. 2, 2008.

[9] H. Ogata, Y. Matsuka, M. M. El Bishouty, and Y. Yano, "LORAMS: linking physical objects and videos for capturing and sharing learning experiences towards ubiquitous learning," Int. J. Mob. Learn. Organ., vol. 3, no. 4, pp. 337, 2009.

[10] G. Indrawan, I. K. Paramarta, and K. Agustini, "A new method of Latin-to-Balinese script transliteration based on noto sans balinese font and dictionary data structure," in The 2nd International Conference on Software Engineering and Information Management (ICSIM), 2019.

[11] The Unicode Consortium, The Unicode Standard Version 13.0 – Core Specification. The Unicode Consortium, 2020.

[12] G. Indrawan, Sariyasa, and I. K. Paramarta, "A new method of Latin-to-Balinese script transliteration based on bali simbar font," in The 4th International Conference on Informatics and Computing (ICIC), 2019.

[13] P. N. Crisnapati et al., "Pasang aksara bot: a Balinese script writing robot using finite state automata transliteration method," J. Phys. Conf. Ser., vol. 1175, no. 1, pp. 012108, 2019.

[14] G. Indrawan, K. Setemen, W. Sutaya, and I. K. Paramarta, "Handling of line breaking on Latin-to-Balinese script transliteration web application as part of Balinese language ubiquitous learning," in The 6th International Conference on Science in Information Technology (ICSITech), 2020.

[15] G. Indrawan, I. P. E. Swastika, Sariyasa, and I. K. Paramarta, "An improved algorithm and accuracy analysis testing cases of Latin-to-Balinese script transliteration method based on bali simbar dwijendra font," Test Eng. Manag., vol. 83, pp. 7676–7683, 2020.

[16] G. Indrawan, G. R. Dantes, K. Y. E. Aryanto, and I. K. Paramarta, "Handling of mathematical expression on Latin-to-Balinese script transliteration method on mobile computing," in The 5th International Conference on Informatics and Computing (ICIC), 2020.

[17] G. Indrawan, I. G. A. Gunadi, M. S. Gitakarma, and I. K. Paramarta, "Latin to Balinese script transliteration: lessons learned from computer-based implementation," in The 4th International Conference on Software Engineering and Information Management (ICSIM), in press.

[18] G. Indrawan, N. N. H. Puspita, I. K. Paramarta, and Sariyasa, "LBtrans-Bot: a Latin-to-Balinese script transliteration robotic system based on noto sans balinese font," Indones. J. Electr. Eng. Comput. Sci., vol. 12, no. 3, pp. 1247-1256, Dec. 2018.

[19] L. H. Loekito, G. Indrawan, Sariyasa, and I. K. Paramarta, "Error analysis of Latin-to-Balinese script transliteration method based on noto sans balinese font," in Proceedings of the 3rd International Conference on Innovative Research Across Disciplines (ICIRAD), 2020.

[20] G. Indrawan, I. K. Paramarta, D. P. Ramendra, I. G. A. Gunadi, and Sariyasa, "A method for the affixed word transliteration to the Balinese script on the learning web application," Turkish J. Comput. Math. Educ., vol. 12, no. 6, pp. 2849–2857, Apr. 2021.

[21] I. M. Suatjana, Bali Simbar Dwijendra. Denpasar: Yayasan Dwijendra, 2009.

[22] G. Indrawan, I. G. A. Gunadi, and I. K. Paramarta, "Towards ubiquitous learning of Balinese-to-Latin script transliteration as part of Balinese language education," in Proceedings of the 4th Asian Education Symposium (AES), 2020.

[23] G. Indrawan, K. U. Ariawan, K. Agustini, and I. K. Paramarta, "Finite-state machine for post-processing method of Balinese script to Latin transliteration," in The 6th International Conference on Science, Technology, and Interdisciplinary Research (IC-STAR), in press.

[24] J. Esling, Handbook of the International Phonetic Association: A Guide to the Use of the International Phonetic Alphabet. Cambridge University Press, 1999.

[25] S. Hollos and J. R. Hollos, Finite Automata and Regular Expressions: Problems and Solutions. Longmont, CO: Exstrom Laboratories LLC, 2013.

[26] L. Groner and G. Manricks, JavaScript Regular Expressions. Birmingham: Packt Publishing, 2015.

[27] A. Sunardi and Suharjito, "MVC architecture: a comparative study between laravel framework and slim framework in freelancer project monitoring system web based," Procedia Computer Science, 2019.

[28] A. Moutaouakkil and S. Mbarki, "MVC frameworks modernization approach: adding MVC concepts to KDM metamodel," Int. J. Adv. Comput. Sci. Appl., 2019.

[29] M. S. Singh, "MVC framework: a modern web application development approach and working," Int. Res. J. Eng. Technol., 2020.

[30] G. Indrawan, I. W. Sutaya, K. U. Ariawan, M. S. Gitakarma, I. G. Nurhayata, and I. K. Paramarta, "A method for scriptio continua management on the transliteration to the Balinese script," Turkish J. Comput. Math. Educ., vol. 12, no. 6, pp. 4182–4191, 2021.

[31] K. M. M. Aung and A. N. Htwe, "Comparison of levenshtein distance algorithm and needleman-wunsch distance algorithm for string matching," Natl. J. Parallel Soft Comput., 2019.

[32] C. Zhao and S. Sahni, "String correction using the damerau-levenshtein distance," BMC Bioinformatics, 2019.

[33] U. Roll, R. A. Correia, and O. Berger-Tal, "Using machine learning to disentangle homonyms in large text corpora," Conserv. Biol., 2018.

[34] J. R. Sánchez-González, "Hemi-and homonyms in the big data era," Diversity, 2020.

[35] A. R. Widiarti and R. Pulungan, "A method for solving scriptio continua in Javanese manuscript transliteration," Heliyon, vol. 6, no. 4, p. e03827, Apr. 2020