Sign Writing Unicode Support: Using an Assisted Entry Process to Neutralize Sign and Symbol Variability

Support de SignWriting en Unicode: utiliser un assistant d'entrée pour neutraliser la variabilité de signe et symbole

Guylhem Aznar, Patrice Dalle
{aznar, dalle}@irit.fr

Équipe TCI, IRIT
UPS - 118 route de Narbonne
31062 Toulouse Cedex 9

Résumé
SignWriting (SW) est l'un des formalismes d'écriture des langues des signes les plus populaires. Il consiste en une représentation planaire des mouvements corporels réalisés. Les segments corporels ainsi que les mouvements sont abrégés en symboles, qui au total avec leur positions respectives réalisent un signe SW. La position précise des symboles ainsi que ces symboles sont sauvegardés dans le fichier.

Le choix d'une série de symboles par l'utilisateur qui compose un signe SW est responsable d'une « variabilité interpersonnelle », qui peut être comparée aux variantes orthographiques (ex: clé/clef) dans les autres langues. Le positionnement fin par l'utilisateur de chaque symbole sur le canevas est responsable de la « variabilité intra-personnelle », qui n'a pas d'équivalent dans les autres langues écrites.

Ces deux variabilités sont une source de difficultés dans le support informatique de SW. Par exemple, les fonctions d'édition de texte, comme rechercher et remplacer, nécessitent des calculs compliqués afin de vérifier si un signe remplit le critère de recherche donné.

Nous présentons ici une solution utilisant le processus d'entrée pour résoudre ces variabilités. La solution proposée fait appel à une phase d'interaction avec l'utilisateur afin de remplacer le signe tel qu'il est entré par un signe standard, issu d'un dictionnaire.

Lors de ce processus, le choix des symboles par l'utilisateur est guidé à travers un menu proposant les symboles les plus fréquents. Leur positionnement est facilité par un positionnement automatique du symbole choisi. Un bénéfice secondaire est donc un gain de temps potentiel pour les utilisateurs.

Les études statistiques et les algorithmes nécessaires pour ces études statistiques, à la base des arbres de probabilités, feront l'objet d'études séparées.

Mots clef: *SignWriting, entrée, prédictive, interaction, variabilité*

Abstract
SignWriting (SW) is one of the most popular writing formalisms of sign language. SW consists of a planar representation of performed body gestures. Body segments and movements are abbreviated into symbols, whose sum and planar position give a SW sign. The precise position of the symbols and the symbols chosen are stored in the file.

The user's choice of a given set of symbols to compose a SW sign is responsible of a variability called “inter-personal variability”, which can be compared to the various acceptable spellings (ex: color/colour) in other languages. The user’s fine positioning of each symbol on the canvas, called “intra-personal variability”, has no equivalent in other written languages.

Both types of variability are responsible of difficulties in SW computer support. For example, search and replace text-editing functions require complicated calculations to compute whether a sign can be considered as matching the entered search pattern.

Using an entry process, we present here a solution to resolve these variabilities. The proposed solution takes advantage of user interaction, and replaces the user-entered sign by a standard sign, which comes from a dictionary.

Through this process, the user’s choice of symbols is guided by a menu which first suggests the most frequently encountered symbols. Their positioning is facilitated by an automatic positioning of the chosen symbol. Thus, a secondary advantage of this is that it could potentially be a timesaver for users.

Statistical studies and their necessary algorithms, upon which probability is based, will be studied separately.

Keywords: *SignWriting, entry, predictive, interaction, variability*

1. SignWriting structure, SWML and Unicode

1.1. Signs and symbols

A sign language *sign*, corresponding to a meaning, is transcribed in a SW *sign*, composed of symbols, positioned on a 2d canvas called a *signbox* (Sutton, 1995).

Figure 1: a SW sign

1.2. SWML

SWML, the most used SW format, offers interesting features (Da Rocha, 2001), such as a XML-based text format, replacement of individual symbols by their SSS entry, and fine positioning of the symbols on the signbox.

However, it suffers from many problems, making it difficult to implement: large file size, a single non-vectorial font, and a lack of support by the operating system core.

1.2.1. File size

The strong point of SWML, i.e. being in an XML-based format, is also a weak point due to file size constraints: in SWML, a single sign requires a full page of XML to portray it, since the bare minimum of one line is necessary to position each symbol.

Along with XML headers overhead, this means a single phrase in sign language requires many pages of SWML.

This causes storage problems, not so important nowadays given the gigabyte-range storage space offered by modern computers. However, it imposes strain on text-handling routines, which have to manipulate such huge files. It also causes delays in electronic transfers (uploading, downloading) of SWML files, especially when compared to other languages who can take advantage of a dedicated encoding system (ex: ISO-8859-1 “latin-1” for Western Europe).

This problem is common in every XML-based format. Many solutions have been proposed (Bayardo, 2004), most of which are based on binary formats (Martin, 1999) or compression (Liefke, 1999), of which both go against the basic principles of XML (Bray, 1996).

1.2.2. One non-vectorial font

This problem is not inherent to the SWML format, but is linked to its implementation. Since the only font currently available is a bitmap font, SW documents cannot be resized, and suffer from artifacts such as pixelation when they are printed.

SVG-based fonts (Ferraiolo, 1999) are under work, which could overcome this problem.

1.2.3. No support at the operating system core

XML-based encodings are made for easy data manipulation, not for text-encoding. While the latter is still technically possible, this is not how it is supposed to be done encoding formats such as the ISO-8859 series or the Unicode (ISO, 2003) standards are the industry-standard solutions to text-encoding problems.
This is why none of the currently available operating-systems support XML-based text-encodings, while most of them support Unicode based encodings. While it may still be technically possible, it would require an enormous amount of work on every target

operating system.

Current SWML applications, then, have to duplicate the existing operating-system's text-handling functions, which causes software overhead and technical difficulty in unifying the applications: all SWML software has to reinvent a symbol positioning system and interface, along with sign handling, before starting to manage its dedicated functions such as sending a SW email.

Therefore, each application (word processor, e-mail client, instant messenger) has to be recreated in SW. Copy and paste options between SW-compatible applications and other applications are not even possible.

Moreover, communication with people who do not have SW-compatible applications is also impossible, which is the reason why many people on SW mailing lists resort to posting pictures or screenshots of their SW documents: to ensure everyone can read them.

Another problem is the lack of integration between different SW-compatible applications: each modification in the symbol system and interface must be ported to other SW-compatible applications, which risk presenting an incoherent interface to the user.

1.3. Unicode

Due to the aforementioned difficulties, a new SW encoding format based on Unicode has been proposed (Aznar, 2004. pp.109-110). It promises easy integration at the operating-system level, thanks to a design which uses the principles of other Unicode supports.

Integration, duplication of function and communication with people who do not use SW are made simple by direct operating system support at the Unicode engine level.

However, while this new encoding may solve many problems issuing from the XML nature of the SWML format, it also has problems which resemble those of the SWML format.

2. Variability

The biggest problem is the non-bijective relationship between a *sign* in sign language and a *sign* in SW.

First, sign languages usually offer different signs for a given meaning - signs that are more or less complicated, or constricted by the limitations of the signer, such as when the signer is holding a drink with one hand. Yet in such situations, the 3d movement is globally the same.

A similar variability exists in the transcription, due to the permissive nature of SW, which

offers many solutions to represent a sign. As long as a reader can mentally reconstruct the sequence of movements for a given sign, the SW transcription can be considered valid.

In addition to this, SW offers different point of views (XY, XZ and YZ 3d axes) and many families of symbols with more or less precision in what they stand for. Moreover, it lets the user manually position each symbol on the canvas.

All this results in many possible transcriptions of a single concept.

The choice of a method to perform a sign belongs to the signer and will not be considered below: we simply consider the case where two or more persons are writing down in SW the signs they see a single signer perform. This simplification will then be reconsidered.

2.1. Inter-personal variation

While writers see the same sign, they can write it in different ways: the most obvious being the set of symbols they pick up to write the sign, which depends on the movements they perceived or their method of deconstructing / reconstructing each movement:

Figure 2: Inter-personal variation in the SW sign for “deaf”

Here we see the writer of the left SW sign represented both the eye and the mouth movements, while the writer of the right SW sign considered the near-closure of the eyelids was the principal feature.

Moreover, the first writer indicated two consequent movements of the index with a single “contact” symbol (the star) in the first position, and a static position of the index in the second position. The second writer used two contact symbols instead, with the index in between.

Experienced SW users and sign language scholars know even more variations, which can be translated into different choices of a symbol set.

2.2. Intra-personal variation

Even within one chosen symbol set, the SW signs may vary. This can be detected though multiple occurrences of the same sign by the same user: the manual positioning of the symbols is responsible for near-imperceptible variations in the fine positioning of every symbol.

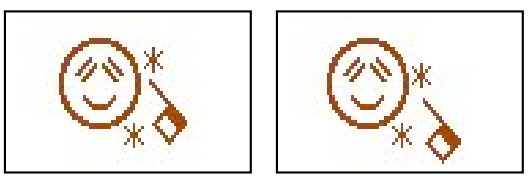

Figure 3: The same sign is performed twice

For example, we can see contact symbols have slight variations, with the upper contact symbol being slightly above the eye in the transcription on the left while being slightly below the eye in the transcription on the right.

Moreover, the index finger has a different position: it is slightly above the mouth in the transcription on the left while being at the mouth level in transcription on the right.

Such near-imperceptible variations in the fine positioning are due to the human manual positioning of the symbols. It is nearly impossible for two SW signs, even while using the same set of symbols, to share the exact position for every symbol on the sign canvas: the relative position of the symbols to the center of the sign changes.

Another variation, which can hardly be represented outside a SW-compatible application, is the global positioning of the sign in the signbox: even if the relative position of the symbols to the center of the sign did not change, their global position could.

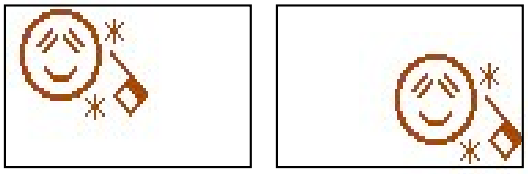

Figure 4: the same set of symbols at the very same relative position, with a difference

All these variations are responsible for divergences in the encoding of one and the same sign. They result from the sub-constrained characteristic of existing SW editors which allow the user to manually position the symbol.

2.3. Neutralizing such differences

Inter and intra-personal variabilities can be either considered as a feature of the precision and flexibility given by SW, or as an encoding problem which should be overcome.

A linguistic evaluation would be required to make sure such variations do not carry any special meaning (as underscript or overscript texts carry in English) . This evaluation was not performed.

Yet given our approach which focuses on computer support, we could tend to favor the latter.

There are also many logical arguments supporting our postulate:

- computer treatment of SW documents suffers from such differences, and statistical studies of how SW is written shall untangle both aforementioned variabilities
- “find and replace” functions, while an evident and useful feature of other language

word-processors, are still being researched for SW (Aerts, 2004. pp. 79-81; Da Rocha, 2004. pp. 32-34)

- documents written by one person may or may not be understandable by another, depending on the amplitude of the inter-personal variations
- a writer may have difficulty maintaining coherence within his or her own documents

Such arguments mean that even if such variations did indeed carry special meanings, they could sometimes be worth neutralizing. And, we studied various solutions that could neutralize such differences. First, using the "FREU" model (Aznar, 2004. pp. 109-110) on how SW can be supported in existing operating systems, there are 3 potential levels at which this neutralization can be performed:

2.3.1. Encoding level

The variabilities can be neutralized at the encoding level, if the encoding can manage some reunification.

It was initially considered an interesting feature of the first proposed Unicode encoding, in which concentric layers neutralized the fine positioning problem, thanks to the use of angular positions. However, it only partially neutralized intra-personal variation: the angular values or the order of the layers could still differ if the changes were significant. Moreover, it did nothing for inter-personal variation.

In a bitmapped approach like SWML, an equivalent solution was the discretization of the Cartesian coordinates. As in the case of Unicode, it did nothing for inter-personal variation, and it only neutralized intra-personal variations which were equivalent to the amount of discretization performed.

2.3.2. Rendering level

An algorithmic and dictionary-based approach can be used to find identical signs. This approach can use proposed pattern matching algorithms for find and replace functions. It could also neutralize both the intra-personal variation, depending on the algorithms' capabilities, and the inter-personal variation, depending on the dictionary size.

However, it will do this at high computational costs, since the comparison would have to be performed every time a sign is displayed.

2.3.3. Entry level

A logical solution for the high computational costs would be saving the result as a metadata.

However, in such a case, it would make more sense to perform this task at the entry level, in order to take advantage of the interaction, and to allow the user to specify one's intentions.

A secondary advantage is that algorithm and dictionary approaches, when completed

through a statistical approach, could save time in the user-entry tasks (Horstmann, 1996. pp. 155-168).

This would happen especially if the most frequently used symbols and signs are presented to the user from the beginning, or if similar positioning and completion helping features were being offered (Roald, 2004. pp. 75-78).

2.4. The signer's choice of a signing method

While not studied above, this can be linked to the variations already described: small, unique variations in the way people sign result in either a different movement (different symbol: inter-personal variation) or a different position (different fine position: intra-personal variation).

While such differences can be interesting from a linguistic perspective, they do not fall within the scope of this paper.

They are addressed in the proposed Unicode format, through the conservation of the original SW transcription as metadata of the sign proposed by the entry process as we will now present.

3. Entry process

While neutralization can occur at any of the 3 levels, we opted for an entry-level process which can interact with the user.

Similar approaches have already been implemented, mostly in mobile devices, which offer a faster text entry process, by means of statistical analysis of the word possibly being entered, followed by the proposal of the most probable word, which if chosen will terminate the remainder of the text-entry process.

The entry process detailed below will concentrate on mouse and keyboard entries, in which SignWriter and SWML Edit rely on the same elements in the user interface: a menu, where symbols are organized in families, and a canvas, where the symbols can be positioned.

3.1. The user chooses

Instead of letting the user choose the symbol from a mouse or keyboard based complex ordering system, the most frequently used symbols, which take into consideration symbols previously positioned by the user, are proposed in the menu next to the existing ordered menu.

The consideration of already positioned symbols means this menu becomes interactive: these “most frequently used symbols” are context sensitive, and depending on what a user is entering, different symbols are presented to the user.

Regardless of the symbol being chosen by the user, the symbol choice probability tree is updated to add this occurrence. Therefore, such a system can take advantage of the difference between the predicted symbols and the symbols being picked up, to "learn" from the user.

While not as dynamic as the Dasher approach for text based entry (Ward, 2000. pp. 129–137), it offers interesting shortcuts to the current deep-menu entry method.

3.2. Automatic initial positioning of the symbol

As soon as a symbol is selected, it is automatically positioned on the 2d canvas – this depends on the most probable position it will take, and is influenced by the already entered symbols. Existing software, such as SWML Edit, by default places the symbol in a corner or at the center of the canvas.

The symbol can still be moved by the user from this position to any other position, as only the initial position of the symbol is changed. If the user decides to change the default initial position, the new updated position is added to the symbols positioning probability tree. The system also takes advantage of the difference between the predicted position and the position decided by the user and "learns" from it.

3.3. Iterative propositions

While more symbols are added and questioned onto the canvas, a representation of the recognized sign is presented to the user next to the active signbox. Different signs can be proposed to the user, in the order of input probability - the user can browse that list thought shortcuts. If a sign is chosen by the user, it terminates the entry process for the current signbox and moves to the next signbox, also updating the probability tree : the choice, the order and the position of the symbols used to enter that sign are added to the dictionary.

Since the sign is replaced by a dictionary sign, coherence can be achieved (ex: tv -> television, color -> colour).

This possibility could be used to initiate a spelling normalization process.

If the user does not pick any of the proposed symbols after a given time, or if a special action is performed, indicating the end of the entry process, the currently represented sign is taken as such: it is then offered to the user to add this sign to its private dictionary.

4. Remaining issues

4.1. Required tools

To have the proper tools, this approach requires multiples subsequent studies.

The required tools are:

- a symbol and sign comparison algorithm, to perform the statistical studies on existing SW documents, which is needed for creating both the symbol choice and the sign dictionary probability trees.
- a symbol choice probability tree, to match a sequence of n symbols with the frequency of the possible following n+1 symbols and their respective initial positions
- a sign dictionary probability tree, to match a potential final sign with the set of chosen symbols, the position of the symbols and the order of symbols being drawn on the canvas.

4.2. No comparative evaluation

The probability tree is one possible approach – other statistical approaches could be used. However, they have not been evaluated in comparative studies.

Likewise, the respective importance of the set of symbols, the symbols' position, and the order of the set of symbols in the sign dictionary, has not been evaluated in comparative studies.

4.3. No tools for comparison

Finally, the acceleration of the entry process is only theoretical and has not been evaluated yet.

It could be evaluated by implementing the process and test it with end users . However, comparison criteria can be used to predict the speed a new approach may offer. KSPC is such a criterion for text-based input (MacKenzie, 2002, pp. 195-210).

New criteria must be developed or former criteria must be adapted to the unique nature of SW.

4.4. Pen-based entries

Pen-based entry approaches have not been studied - they will at the least require a symbol recognition algorithm, since they could benefit from an automatic recognition of the symbol being drawn, thus avoiding the user having to pick up the symbol from the menu.

Similar statistical studies should be performed, since pen-based entries could also benefit from automatic updates on the represented symbol, depending on the continuous pen entry. Finally, along with automatic positioning, drag and drop, or point to position could relieve the user of hassle of fine positioning.

However, so many complex computer-human-interaction aspects require an analysis by CHI oriented teams.

5. Conclusion

The variability problem has been identified, and the two variabilities have been separated. No encoding can fix both variabilities by itself, which means that a dedicated step is required

The position of this neutralization process has been evaluated: it appears it should belong to the entry level, which is consistent with similar approaches for existing text-completion methods used on handheld computers.

To materialize this neutralization process, a user interaction scenario has also been proposed – however it can not be implemented immediately.

Additional studies are required to implement such an approach: first, comparison algorithms are required to build the probability trees. Then, a set of documents is needed for the algorithms to run on and populate the trees.

The pen-based entry could benefit from these studies, but will likely require studies of its own.

Finally, evaluation of the proposed work must be performed on existing SW documents, and be submitted to user testing to validate the propositions. In order to avoid a complex full reimplementation, an experimental implementation of the neutralization process could be added to existing SWML based tools, which suffer from the same variability problems.

This would validate the approach and evaluate the potential acceleration of the entry process.

Bibliography

Aerts, S., Braem, B., Van Mulders, K., & De Weerdt, L. (2004). Searching SignWriting signs. Proceedings from LREC 2004: *4th International Conference on Language Resources and Evaluation (RPSL Workshop)*, pp. 79-81. Lisbon, Portugal.

Aznar, G., & Dalle, P. (2004). Computer support for SignWriting written form of sign language. Proceedings from LREC 2004: *4th International Conference on Language Resources and Evaluation (RPSL Workshop)*, pp. 109-110, Lisbon, Portugal.

Bayardo, R., Gruhl, D., Josifovski, V., & Myllymaki, J. (2004). An evaluation of binary XML encoding optimizations for fast stream based XML processing. Proceedings from WWW 2004: *The Thirteenth International World Wide Web Conference*, May 17-22, 2004. New York, New York.

Bray, T., & Sperberg-McQueen, C. M. (1996, November). Extensible Markup Language. W3C working draft: Work in progress. Retrieved 2006-12-06 from http://www.w3.org/TR/WD-xml-961114.html

Da Rocha Costa, A., Dimuro, G., & De Freitas, J. (2004). A sign matching technique to support searches in sign language texts. Proceedings from LREC 2004: 4th International Conference on Language Resources and Evaluation (RPSL Workshop), pp. 32-34, Lisbon, Portugal.

Da Rocha Costa, A., & Dimuro, G. (2001). A SignWriting-based approach to sign language processing. Proceedings from GW 2001: Gesture Workshop. pp. 202-212 London, UK.

Ferraiolo, J. (1999, December). Scalable vector graphics (SVG) specification. W3C working draft: Work in progress. Retrieved 2006-12-06 from http://www.w3.org/TR/1999/WD-SVG-19990211/

Horstmann-Koester, H., & Levine, S. (1996). Effect of a word prediction feature on user performance. Augmentative and Alternative Communication, 12, 155-168.

ISO/IEC 10646-1:2003 (1993). Information technology - universal multiple-octet coded character set (UCS). Retrieved 2006-12-06 from http://www.iso.ch/iso/fr/CatalogueDetailPage.CatalogueDetail?CSNUMBER=39921

Liefke H., & Suciu, D. (1999). XMill: an efficient compressor for XML data. *Technical Report MSCIS*, pp. 153-164. University of Pennsylvania.

MacKenzie, I. S. (2002). KSPC (keystrokes per character) as a characteristic of text entry techniques. Proceedings of the *Fourth International Symposium on Human-Computer Interaction with Mobile Devices*, pp. 195-210. Heidelberg, Germany: Springer-Verlag

Martin, B., & Jano, B. (1999). Wap binary xml content format. *W3C recommendation wbxml*. Retrieved 2006-12-06 on http://www.w3.org/1999/06/NOTE-wbxml-

19990624/

Sutton, V., & Gleaves, R. (1995). SignWriter - the world's first sign language processor. *Ed. Center for Sutton Movement Writing*. La Jolla, CA.

Roald, I. (2004). Making dictionaries of technical signs: from paper and glue through SW-DOS to SignBank. Proceedings from LREC 2004: *4th International Conference on Language Resources and Evaluation (RPSL Workshop)*, pp. 75-78, Lisbon, Portugal.

The Unicode Consortium. The Unicode Standard: Version 2.0. Reading, Mass.: Addison-Wesley Developers Press, 1996.

Ward, D. J., Blackwell, A. F., & MacKay, D. J. C. (2000). Dasher - a data entry interface using continuous gestures and language models. Proceedings of the ACM Symposium on User Interface Software and Technology: *UIST'00*, pp. 129–137, New York: ACM.