Mark-Jan Nederhof*, Stéphane Polis**, Serge Rosmorduc***

Unicode Control Characters for Ancient Egyptian

1. THE UNICODE STANDARD

Unicode offers a universal solution to text encoding.[1] As of version 13 (2020), the Unicode standard comprises 143,859 characters in altogether 154 modern and ancient scripts. Different scripts can be alternated within a single document by assigning a unique number, or *code point*, to each character. Unicode characters are "abstract" in the sense that their exact appearance is not fixed: an italic "A" in Helvetica and a bold "A" in Times Roman correspond to the same code point. Font families, font styles and font weights are determined by higher-level protocols that are used in combination with Unicode, such as HTML and CSS for web pages or RTF for word processors.

Some Unicode characters do not result in a visual appearance on their own, but control the formatting or modify the rendering of other characters. These are known as *control characters*. The most common examples are the line feed and the horizontal tab. A very different kind of control character, which allows hieroglyphic signs to be positioned in relation to each other, is the topic of this article.

Thanks to Unicode, a large number of fonts can be used for a wide range of applications. Copy-and-paste functionality allows transfer of text between applications, including web pages and documents prepared using common word processors. Web search engines and search functionality in word processors and databases such as Word and Framemaker rely on Unicode.

* University of St Andrews.
** F.R.S.–FNRS/University of Liège.
*** CNAM.
The ongoing work reported here involves many other colleagues, most notably Andrew Glass, Jorke Grotenhuis, and Daniel Werning. Pivotal has been the assistance of Deborah Anderson.
1. Korpela 2006.

2. DIGITAL ENCODING OF EGYPTIAN TEXTS

A popular format to encode ancient Egyptian texts is the *Manuel de Codage* (MdC),[2] most notably in the JSesh implementation.[3] Another such encoding format is PLOTTEXT.[4] The initial motivation for the MdC and PLOTTEXT was to prepare printed publications. The former is now widely used for text corpora as well, such as *Ramses Online*[5] and the *Thesaurus Linguae Aegyptiae*.[6] However, electronic corpora and interchange of textual resources between different projects have requirements that diverge quite significantly from those of printed editions.[7] Moreover, with current technical solutions, common word processors need to be used in combination with tools such as JSesh in order to incorporate hieroglyphic texts as images within documents.

Conversely, Unicode has a number of inherent limitations that seem to preclude its use in some areas of Egyptology. For example, the notion of *abstract* character implies that details of appearance cannot be part of the encoding, which hinders applications in palaeography. Moreover, arbitrary fine-tuning in terms of scaling and positioning, as is often done in the publication of Egyptian texts, is beyond the capabilities of common font technologies such as OpenType. Nonetheless, there are many potential applications of Unicode in the fields of lexical and morpho-syntactic studies, for instance, as well as in the transfer of textual resources between different individuals, projects, and tools.

3. CONTROL CHARACTERS FOR EGYPTIAN

Since Unicode 5.2 (2009), there are 1071 code points of ancient Egyptian hieroglyphs. These code points in themselves have been of limited use, as until recently no mechanism was available to compose signs into the actual hieroglyphic text as it appears in the original inscriptions, with signs next to one another, above one another, or in other spatial arrangements.

An early Unicode proposal to add control characters for ancient Egyptian took three primitives from the MdC tradition.[8] These were the *horizontal joiner* "*", which arranges signs next to one another, the *vertical joiner* ":", which arranges signs (or horizontally joined groups) above one another, and a *ligature joiner* "+" for any other arrangements of signs. The latter was not in the original MdC, but in the form of "&" has been widely used since the release of WinGlyph.[9]

The proposal has been criticised for several reasons.[10] First, different users can very well attribute different meanings to "+". This violates the main aim of Unicode, which is the interchange of encodings without introducing ambiguity. Second, vertical groups of signs could not be combined

2. Buurman et al. 1988.
3. http://jseshdoc.qenherkhopeshef.org.
4. Stief 1985a; Stief 1985b; Stief 1988; Stief 2001.
5. http://ramses.ulg.ac.be/.
6. http://aaew.bbaw.de/tla/index.html.
7. Nederhof 2013.
8. Richmond, Glass 2016.
9. van den Berg 1993; van den Berg 1997.
10. Nederhof et al. 2016a.

into a larger group using the horizontal joiner, which meant that many common groups could not be encoded, unless the catch-all "+" was used, which is problematic as it is. Lastly, there was an implicit assumption that all valid/attested groups of signs could be enumerated and stored in precomposed form in a font.

Subsequent proposals addressed these issues.[11] To find an alternative to "+", ideas were taken from PLOTTEXT, which has six primitives that allow a (group of) sign(s) to be "inserted" at empty spaces in or around a bigger sign. Four of these primitives, henceforth called "corner insertions", insert a group within one of four corners of the bigger sign. A fifth primitive inserts a group just above the feet of the bigger sign, assuming this is a bird. A sixth primitive, a "central insertion", inserts a group within another sign. This was specifically intended for *ḥw.t* enclosures.

Primitives akin to the corner insertion exist in other types of encoding next to PLOTTEXT. For example, MacScribe[12] added two binary operators to the MdC tradition, each of which inserts a group of signs into a particular *zone* immediately next to a base sign. Such a zone is typically an empty corner of the base sign, or the empty space above a bird's feet. A limitation is that each sign can have a maximum of two zones, and these zones must be specified individually for each sign.

A more general and precisely defined solution is offered by RES.[13] It allows insertion into one of the four corners, into one of the four sides, or in the middle of a bigger sign. It is also possible to insert a group within a group. Furthermore, one may manually adjust the (x, y) coordinates where the insertion is to take place. Another innovative feature is that the rendering of the insertion depends on the exact contours of the signs: the inserted group is gradually scaled up from 0 until a given minimal distance is reached between it and the larger sign. This distance is the default distance between signs and groups, but it can also be manually adjusted. If the distance is set to 0, then the inserted group is scaled up (and where applicable moved up/down or left/right) until it fits snugly between the larger sign and the bounding box around it.

PLOTTEXT's four corner insertions have been adopted in Unicode. If the base sign is a bird, then the lower left corner insertion places the inserted group above its feet, and thereby fulfils the role of the fifth insertion primitive of PLOTTEXT. An "overlay" control character was also adopted, which renders one sign, or one group of signs, on top of another. Lastly, a pair of control characters was introduced that act as parentheses. This allows nesting of horizontal and vertical groupings, as well as corner insertions, in principle to an arbitrary depth, although in practice the maximum nesting depth may be limited by the font. Efforts were initiated to design OpenType fonts that do not rely on enumerating groups of signs, but that interpret control characters dynamically. In this way, fonts will be able to render groups of signs that were not seen before.

Examples of groups in Unicode (all taken from Section 4) are presented in fig. 1: (a) the horizontal joiner binds more tightly than the vertical joiner; (b) a pair of parentheses is required for vertical groups that are combined with the horizontal joiner; (c-f) there can be insertions in

11. Nederhof et al. 2016b; Nederhof et al. 2017; Glass et al. 2017.
12. Gozzoli 2013.
13. http://mjn.host.cs.st-andrews.ac.uk/egyptian/res/.

multiple corners; (g) corner insertions bind more tightly than the horizontal and vertical joiners; (h-i) a pair of parentheses is required for vertical or horizontal groups or other corner insertions that are themselves inserted in a corner; (j) there can be corner insertions in an overlay.

The Unicode repertoire of nine control characters for Ancient Egyptian was informed by studies of signs composition[14] and was chosen specifically to satisfy the following requirements:

1. The meaning of each control character must be simple, well-defined and stable. This ensures that encodings preserve their validity over time and can be transferred between projects and tools without introducing ambiguity. However, there is some freedom in how a font could realize the scaling and positioning of signs. For example, the exact distance between signs in a horizontal or vertical group is not specified.
2. The repertoire of control characters must cover the main types of spatial arrangements observed in hieroglyphic and hieratic texts from different periods. To be more exact, the appearance of an original inscription and the appearance obtained by rendering its encoding must be similar enough that most users agree that it is the same text. In Section 4 we assess the extent to which this requirement is currently fulfilled.
3. The control characters can be implemented in modern font technology, in particular OpenType. This is discussed further in Section 5.
4. Encodings can be effectively searched for patterns of signs and specific spatial arrangements. It should be noted that if atomic code points were introduced for complex groups of signs, it would become easy to achieve an encoding for each known text with few or no control characters, as implementation of additional code points in itself is straightforward. However, the whole encoding standard would then become unwieldy and unstable as more such code points are added for newly considered texts, and the search functionality becomes close to impossible to implement. The best compromise therefore appears to be a small repertoire of powerful control characters, reducing the need for atomic encoding of composed and transformed signs to a minimum.

4. ADEQUACY

To critically assess the coverage of the control characters currently available in Unicode, monumental inscriptions from the Old Kingdom,[15] the New Kingdom,[16] and the Ptolemaic era[17] were investigated, with texts written in lines and in columns, and left-to-right or right-to-left.

Fig. 2 exemplifies all the types of encodings discussed in Section 3. Groups such as (q) and (z) abound and illustrate that vertical groups can occur inside horizontal groups, which can themselves be part of vertical groups, etc. Examples such as (v) further show that there is no obvious limitation to the number of signs that can be grouped using the horizontal and vertical joiners.

14. Fischer 1977; Meeks 2017; Polis 2018.
15. Lloyd et al. 2008; James 1961.
16. Iskander, Goelet 2015.
17. Biston-Moulin, Thiers 2016.

Examples of corner insertion are widespread and include those discussed before, viz. (g), (m-p) and (r-s), as well as (h) and (w). In (t), a group with bottom-right corner insertion is inserted in the bottom-left corner of another sign. Groups such as (j) and (k) can also be analysed as corner insertions.

Unicode currently lacks a central insertion, which would be required for groups such as (e) and (y). It is further common for low/wide signs and high/narrow signs to be rotated by a quarter turn if that makes them fit better in the composition of a group. Some signs may also be rotated by half a turn without this changing their meaning, as for example the crescent moon in (n). In addition, signs may be mirrored horizontally, sometimes in order to create meaningful visual interactions with other signs, and sometimes because they have no clear front or back, as in (l). Hence, both rotation and mirroring would be desiderata among further Unicode controls.

Finally, a frequent phenomenon in ancient Egyptian inscriptions is that neighbouring groups are squeezed together to reduce the amount of empty space, as for example in (a), (b), (d), (f), (i), (u) and (x). This may be called *kerning*, by analogy with the concept of this name in modern typography. What is different here is that this squeezing together generally involves a pair of neighbouring groups, rather than a pair of neighbouring characters. As a consequence, this form of kerning is beyond the capabilities of modern font technology, and a solution is not likely to be found within the framework of Unicode. If kerning encoding is required, then a more specialised technology such as RES needs to be used, which realizes kerning dynamically by analysing the contours of signs.

5. IMPLEMENTATION

The signs within a group are scaled and positioned depending on the sizes of the other signs. Ideally, this involves arithmetic and dynamic scaling, which are outside the capabilities of OpenType. However, arithmetic can to some extent be simulated by OpenType's features, and a font may contain several precompiled scaled copies of each sign.[18] Recently, much progress along these lines has been made by Andrew Glass.[19]

In addition, open-source code is available to automatically construct a font that can handle all the groups within a given corpus.[20] Fig. 3 demonstrates such a font in a web page. The disadvantage of this approach is that the font needs to be regenerated each time the corpus is extended.

Lastly, there is open-source code to render encodings in webpages in terms of HTML canvas, implemented using an existing framework for RES, whose functionality subsumes that of the Unicode control characters.[21] This also includes an online graphical editor.[22]

18. Nederhof et al. 2017.
19. Glass 2020.
20. https://github.com/nederhof/opentypehiero.
21. https://github.com/nederhof/resjs.
22. https://mjn.host.cs.st-andrews.ac.uk/egyptian/res/js/edit_uni.html.

6. OUTLOOK

As stated above, it is our hope that future versions of Unicode will include central insertion, rotation, and mirroring. Furthermore, encoding of cartouches and other enclosures still awaits a permanent solution. As complete and undamaged texts are the exception in ancient inscription corpora, further desiderata are primitives for lacunas and shading.

The work reported here on the control characters should be envisioned as complementary to the extension of the hieroglyphic signs repertoire in Unicode. We advocate a drastic change of direction here: the actual attestations of signs (with an analysis of their iconic features and functions in context) should be at the centre of the definition of any new code points. Such an endeavour can be supported by the *Thot Sign List*,[23] an open digital repertoire of hieroglyphic signs.[24]

BIBLIOGRAPHY

van den Berg 1993
van den Berg, H., "GLYPH for Windows—Hieroglyphic Text Processing on IBM-Compatible Computers", *InfEg* 8, 1993, pp. 113–121.

van den Berg 1997
van den Berg, H., *Manuel de Codage: A Standard System for the Computer-Encoding of Egyptian Transliteration and Hieroglyphic Texts*, 1997, <http://www.catchpenny.org/codage/>, accessed 22 December 2020.

Biston-Moulin, Thiers 2016
Biston-Moulin, S., Thiers, C., *Le temple de Ptah à Karnak*, BiGen 49, Cairo, 2016.

Buurman et al. 1988
Buurman, J., Grimal, N., Hainsworth, M., Hallof, J., van der Plas, D., "Inventaire des signes hiéroglyphiques en vue de leur saisie informatique", *InfEg* 2, 1988 (3rd ed.).

Fischer 1977
Fischer, H.G., "The Evolution of Composite Hieroglyphs in Ancient Egypt", *MMJ* 12, 1977, pp. 5–19.

Glass 2020
Glass, A., *Cluster Model for Egyptian Hieroglyphic Quadrats*, 2020, <http://www.unicode.org/L2/L2020/20176-hierogyph-cluster.pdf>, accessed 22 December 2020.

Glass et al. 2017
Glass, A., Hafemann, I., Nederhof, M.-J., Polis, S., Richmond, B., Rosmorduc, S., Schweitzer, S., *A Method for Encoding Egyptian Quadrats in Unicode*, 2017, <http://www.unicode.org/L2/L2017/17112r-quadrat-encoding.pdf>, accessed 22 December 2020.

Gozzoli 2013
Gozzoli, R.B., "Hieroglyphic Text Processors, Manuel de Codage, Unicode, and Lexicography", in S. Polis, J. Winand (eds.), *Texts, Languages & Information Technology in Egyptology*, AegLeod 9, Liège, 2013, pp. 89–101.

Iskander, Goelet 2015
Iskander, S., Goelet, O., *The Temple of Ramesses II in Abydos*, vol.1: *Wall Scenes*, Atlanta (GA), 2015.

James 1961
James, T.G.H., *Hieroglyphic Texts from Egyptian stelae, etc.*, Part I, London, 1961 (2nd ed.).

23. http://thotsignlist.org/.
24. Polis et al. 2021.

Korpela 2006
Korpela, J.K., *Unicode Explained*, Sebastopol (CA), 2006.

Lloyd et al. 2008
Lloyd, A.B., Spencer, A.J., El-Khouli, A., *Saqqâra Tombs*, vol. III: *The Mastaba of Neferseshemptah*, ASEg 41, London, 2008.

Meeks 2017
Meeks, D., "Ancient Egyptian Composite Hieroglyphs: a Typology", in *Eikones Workshop, Egyptian and Maya writing: Comparing Hieroglyphic Domains, Basel, June 9–11, 2017*, Basel, 2017 (unpublished, draft available online: https://www.academia.edu/33538298/Ancient_Egyptian_%20composite_hieroglyphs_a_typology).

Nederhof 2013
Nederhof, M.-J., "The Manuel de Codage Encoding of Hieroglyphs Impedes Development of Corpora", in S. Polis, J. Winand (eds.), *Texts, Languages & Information Technology in Egyptology*, AegLeod 9, Liège, 2013, pp. 103–110.

Nederhof et al. 2016a
Nederhof, M.-J., Rajan, V., Richter, T.S., Hafemann, I., Schweitzer, S., Polis, S., Rosmorduc, S., *Comments on Three Control Characters for Egyptian Hieroglyphs*, 2016, <http://www.unicode.org/L2/L2016/16090-comment-ctl-char-egyptian.pdf>, accessed 22 December 2020.

Nederhof et al. 2016b
Nederhof, M.-J., Rajan, V., Lang, J., Polis, S., Rosmorduc, S., Richter, T.S., Hafemann, I., Schweitzer, S., *A Comprehensive System of Control Characters for Ancient Egyptian Hieroglyphic Text (Preliminary Version)*, 2016, <http://www.unicode.org/L2/L2016/16177-egyptian.pdf>, accessed 22 December 2020.

Nederhof et al. 2017
Nederhof, M.-J., Rajan, V., Lang, J., Polis, S., Rosmorduc, S., Richter, T.S., Hafemann, I., Schweitzer, S., *A System of Control Characters for Ancient Egyptian Hieroglyphic Text (Updated Version)*, 2017, <http://www.unicode.org/L2/L2016/16210r-egyptian-control.pdf>, accessed 22 December 2020.

Polis 2018
Polis, S., "The Functions and Toposyntax of Ancient Egyptian hieroglyphs", *SIGNATA* 9, 2018, pp. 291–363.

Polis et al. 2021
Polis, S., Desert, L., Dils, P., Grotenhuis, J., Razanajao, V., Richter, T.S., Rosmorduc, S., Schweitzer, S.D., Werning, D.A., Winand, J., "The Thot Sign List (TSL)—an Open Digital Repertoire of Hieroglyphic Signs", *ENiM* 14, 2021, pp. 55–74.

Richmond, Glass 2016
Richmond, B., Glass, A., *Proposal to Encode Three Control Characters for Egyptian Hieroglyphs*, 2016, <https://www.unicode.org/L2/L2016/16018r-three-for-egyptian.pdf>, accessed 22 December 2020.

Stief 1985a
Stief, N., "Hieroglyphen, Koptisch, Umschrift, u.a.—ein Textausgabesystem", *GöttMisz* 86, 1985, pp. 37–44.

Stief 1985b
Stief, N., "A Programm System for the Edition of Text", *InfEg* 1, 1985, pp. 197–208.

Stief 1988
Stief, N., "Weitere Möglichkeiten bei der Hieroglyphenausgabe via Computer", *InfEg* 5, 1988, pp. 46–51.

Stief 2001
Stief, N., *PLOTTEXT—ein Programmsystem zur Ausgabe von Texten*, Version 4.09, Regionales Hochschulrechenzentrum (RHRZ) der Universität Bonn, 2001.

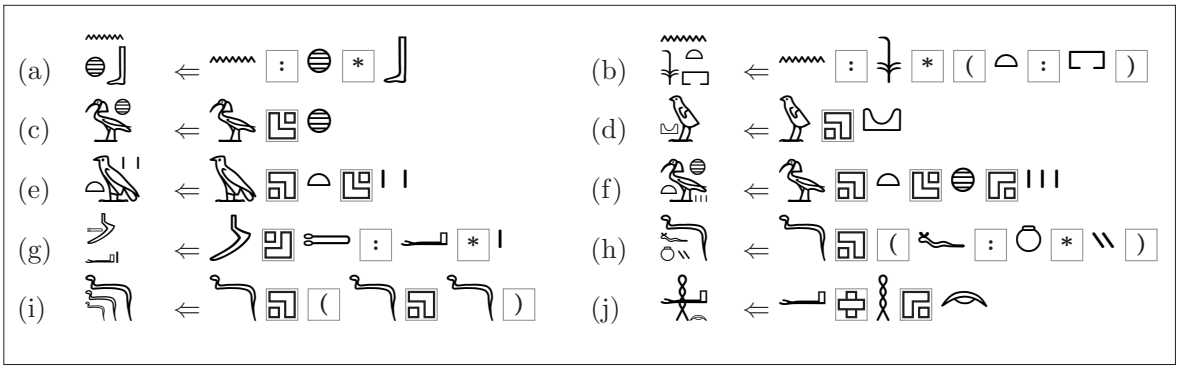

Fig. 1. Encodings of hieroglyphic text in Unicode.

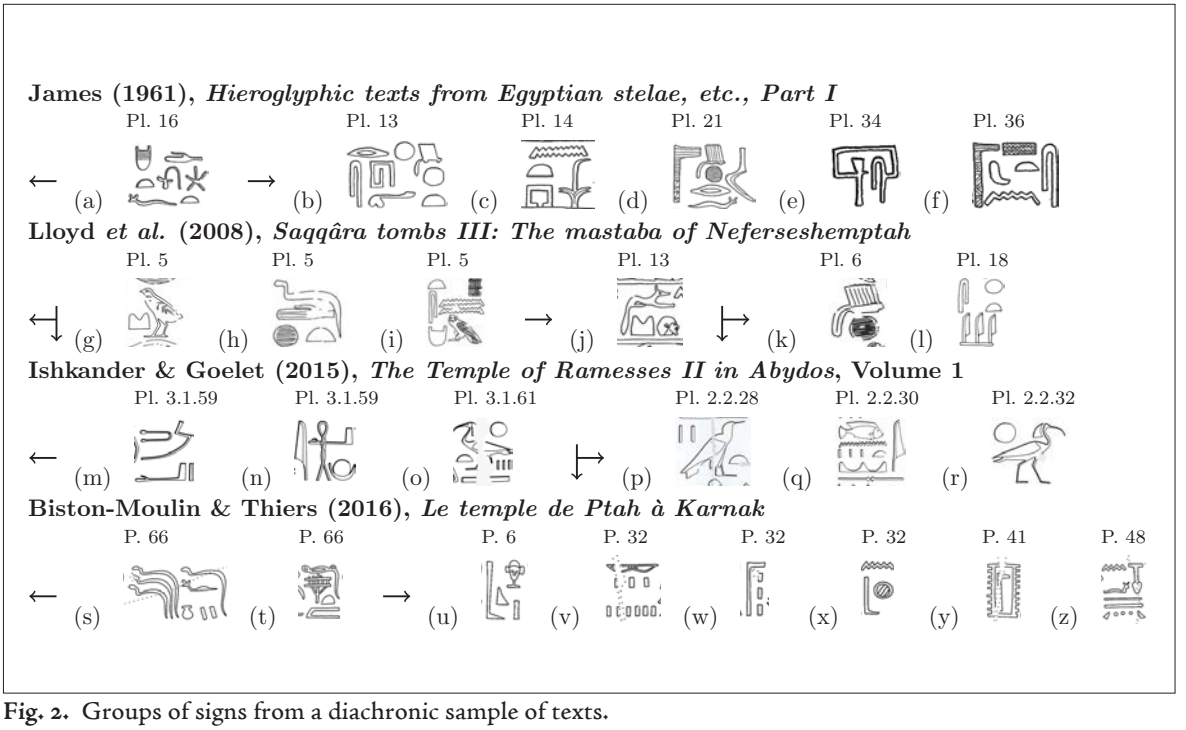

Fig. 2. Groups of signs from a diachronic sample of texts.

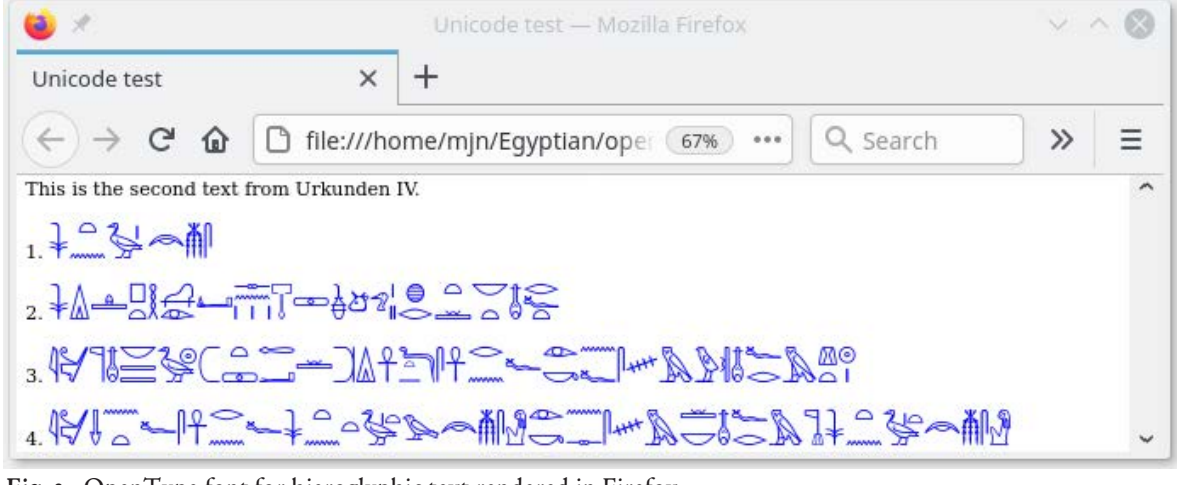

Fig. 3. OpenType font for hieroglyphic text rendered in Firefox.