1

Unicode Control Characters for Ancient Egyptian

Mark-Jan Nederhof

Stéphane Polis

Serge Rosmorduc

Draft chapter for handbook, formatting 7th January 2021.

1.1 The Unicode Standard

Unicode offers a universal solution to text encoding (Korpela, 2006). As of version 13 (2020), the Unicode standard comprises 143,859 characters in altogether 154 modern and ancient scripts. Different scripts can alternate within a single document by virtue of the assignment of 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 in the case of web pages or RTF for word processors.

Some Unicode characters do not result in a visual appearance on their own, but control formatting or modify the rendering of other characters. These are known as *control characters*. Common examples are the line feed and the horizontal tab. Control characters of a very different kind, which allow positioning of hieroglyphic signs with respect to one another, are 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. Search engines on the Web and search functionality of word processors and databases such as Word and Framemaker rely on Unicode.

1.2 Digital Encoding of Egyptian Texts

A popular format to encode Ancient Egyptian texts is the Manuel de Codage (MdC) (Buurman *et al.*, 1988), most notably in the implementation of JSesh.[1] Another such encoding format is PLOTTEXT (Stief, 1985b,a, 1988, 2001). The initial motivation for the MdC and PLOTTEXT was to prepare printed publications. It is now widely used for text corpora as well, such as *Ramses Online*[2] and the *Thesaurus Linguae Aegyptiae*[3]. However, electronic corpora and interchange of textual resources between different projects have requirements that diverge quite significantly from the ones of printed editions (Nederhof, 2013). Moreover, with the 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 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 transfer of textual resources between different individuals, projects, and tools.

[1] `http://jseshdoc.qenherkhopeshef.org`
[2] `http://ramses.ulg.ac.be/`
[3] `http://aaew.bbaw.de/tla/index.html`

1.3 Control Characters for Egyptian

Since Unicode 5.2 (2009) there are 1071 code points of Ancient Egyptian hieroglyphs. These code points by themselves have been of limited use, as until recently no mechanism was available to compose signs into actual hieroglyphic text as it appears in 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 (Richmond & Glass, 2016). 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 '&' it has been widely used since the release of WinGlyph (van den Berg, 1993, 1997).

The proposal attracted criticism for several reasons (Nederhof *et al.*, 2016b). 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 into a larger group using the horizontal joiner, which meant that many common groups could not be encoded, unless one resorted to use of the catch-all '+', 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 (Nederhof *et al.*, 2016a, 2017; Glass *et al.*, 2017). 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 'center insertion', inserts a group within another sign. This was specifically intended for *ḥw.t* enclosures.

Primitives akin to corner insertion exist in other types of encoding next to PLOTTEXT. E.g. MacScribe (Gozzoli, 2013) 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 the feet of a bird. A

limitation is that each sign can have at most two zones, and what these zones are needs to be specified for each sign individually.

A more general and more precisely defined solution is offered by RES.[4] 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.

The four corner insertions of PLOTTEXT were 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 one another. Lastly, a pair of control characters was introduced that act as parentheses. This allows horizontal and vertical groupings, as well as corner insertions, to be nested, in principle to arbitrary depth, although in practice the maximum depth of nesting 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 1.4.) are presented in Table 1.1: (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 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.

[4] `http://mjn.host.cs.st-andrews.ac.uk/egyptian/res/`

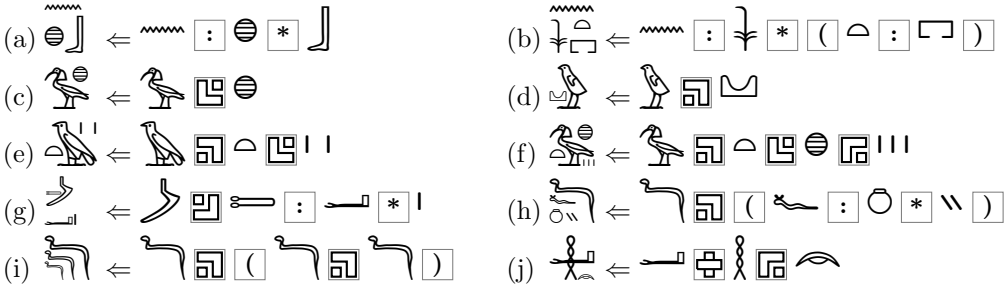

Table 1.1. Encodings of hieroglyphic text in Unicode.

The Unicode repertoire of nine control characters for Ancient Egyptian was informed by studies of composition of signs (Fischer, 1977; Meeks, 2017; Polis, 2018) and was chosen to specifically satisfy the following requirements:

(1) Each control character must have a meaning that is 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 for most users to agree that they are the same text. In Section 1.4 we assess to what extent 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 1.5.

(4) Encodings can be effectively searched for patterns of signs and specific spatial arrangements.

It should be noted that if one were to introduce *atomic* code points 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 encoding standard as a whole would then become unwieldy and unstable as more such code points are added for newly considered texts, and 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.

James (1961), *Hieroglyphic texts from Egyptian stelae, etc., Part I*

Pl. 16 Pl. 13 Pl. 14 Pl. 21 Pl. 34 Pl. 36

← (a)  → (b)  (c)  (d)  (e)  (f) 

Lloyd *et al.* (2008), *Saqqâra tombs III: The mastaba of Neferseshemptah*

Pl. 5 Pl. 5 Pl. 5 Pl. 13 Pl. 6 Pl. 18

↤ (g)  (h)  (i)  → (j)  ↦ (k)  (l) 

Ishkander & Goelet (2015), *The Temple of Ramesses II in Abydos*, Volume 1

Pl. 3.1.59 Pl. 3.1.59 Pl. 3.1.61 Pl. 2.2.28 Pl. 2.2.30 Pl. 2.2.32

← (m)  (n)  (o)  ↦ (p)  (q)  (r) 

Biston-Moulin & Thiers (2016), *Le temple de Ptah à Karnak*

P. 66 P. 66 P. 6 P. 32 P. 32 P. 32 P. 41 P. 48

← (s)  (t)  → (u)  (v)  (w)  (x)  (y)  (z) 

Table 1.2. Groups of signs from a diachronic sample of texts.

1.4 Adequacy

To critically assess the coverage of the control characters currently available in Unicode, monumental inscriptions from the Old Kingdom (Lloyd *et al.*, 2008; James, 1961), the New Kingdom (Ishkander & Goelet, 2015), and the Ptolemaic era (Biston-Moulin & Thiers, 2016) were investigated, with texts written in lines and in columns, and left-to-right or right-to-left.

Table 1.2 exemplifies all the types of encodings discussed in Section 1.3. Groups such as (q) and (z) abound that 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.

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 analyzed as corner insertions.

Currently missing from Unicode is a center 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 encoding of kerning is required, one must currently resort to more specialized technology such as RES, which realizes kerning dynamically by analyzing the contours of signs.

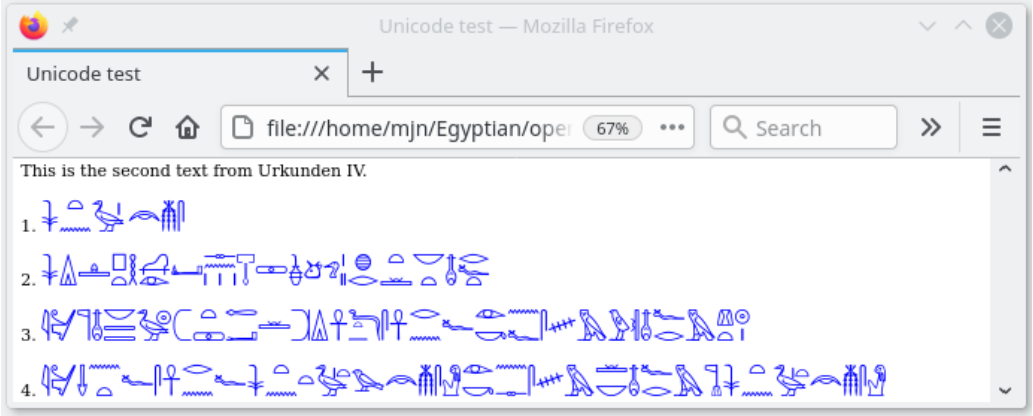

Table 1.3. OpenType font for hieroglyphic text rendered in Firefox.

1.5 Implementation

Signs within a group are scaled and positioned depending on sizes of 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 scalings of each sign (Nederhof *et al.*, 2017). Recently, much progress along these lines has been made by Andrew Glass (Glass, 2020).

In addition, open-source code is available to automatically construct a font that can handle all groups within a given corpus.[5] Table 1.3 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.[6] This also includes an online graphical editor.[7]

[5] `https://github.com/nederhof/opentypehiero`
[6] `https://github.com/nederhof/resjs`
[7] `https://mjn.host.cs.st-andrews.ac.uk/egyptian/res/js/edit_uni.html`

1.6 Outlook

As stated above, it is our hope that future versions of Unicode will include center insertion, rotation, and mirroring. Furthermore, encoding of cartouches and other enclosures still awaits a permanent solution. As complete and undamaged texts are the exception within corpora of ancient inscriptions, 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 repertoire of hieroglyphic signs in Unicode. We advocate a drastic change of direction here: actual attestations of signs (with an analysis of their iconic features and functions in context) should be at the center of the definition of any new code points. Such an endeavour can be supported by the *Thot Sign List*[8], an open digital repertoire of hieroglyphic signs (Polis *et al.*, 2020).

[8] `http://thotsignlist.org/`

Acknowledgements

The ongoing work reported here involves many other colleagues, most notably Andrew Glass, Simon Schweitzer, and Daniel Werning. Pivotal has been the assistance of Deborah Anderson.

References

van den Berg, H. (1993), GLYPH for Windows – hieroglyphic text processing on IBM-compatible computers, in *Informatique et Égyptologie 8*, (113–121).

van den Berg, H. (1997), Manuel de Codage: A standard system for the computer-encoding of Egyptian transliteration and hieroglyphic texts, `http://www.catchpenny.org/codage/`, accessed 2020-12-11.

Biston-Moulin, S. & C. Thiers (2016), *Le temple de Ptah à Karnak*, volume 49 of *Bibliothèque générale*, Institut français d'archéologie orientale, Le Caire.

Buurman, J., N. Grimal, M. Hainsworth, J. Hallof, & D. van der Plas (1988), *Inventaire des signes hiéroglyphiques en vue de leur saisie informatique – Informatique et Égyptologie 2*, Institut de France, Paris, 3rd edition.

Fischer, H.G. (1977), The evolution of composite hieroglyphs in Ancient Egypt, *Metropolitan Museum Journal* 12:5–19.

Glass, A. (2020), Cluster model for Egyptian hieroglyphic quadrats, `http://www.unicode.org/L2/L2020/20176-hierogyph-cluster.pdf`.

Glass, A., I. Hafemann, M.-J. Nederhof, S. Polis, B. Richmond, S. Rosmorduc, & S. Schweitzer (2017), A method for encoding Egyptian quadrats in Unicode, `http://www.unicode.org/L2/L2017/17112r-quadrat-encoding.pdf`.

Gozzoli, R.B. (2013), Hieroglyphic text processors, Manuel de Codage, Unicode, and lexicography, in S. Polis & J. Winand (eds.), *Texts, Languages & Information Technology in Egyptology*, Presses Universitaires de Liège, (89–101).

Ishkander, S. & O. Goelet (2015), *The Temple of Ramesses II in Abydos: Volume 1, Wall Scenes*, Lockwood Press, Atlanta.

James, T.G.H. (1961), *Hieroglyphic texts from Egyptian stelae, etc., Part I*, British Museum, 2nd edition.

Korpela, J.K. (2006), *Unicode Explained*, O'Reilly.

Lloyd, A.B., A.J. Spencer, & A. El-Khouli (2008), *Saqqâra tombs III: The mastaba of Neferseshemptah*, volume 41 of *Archaeological Survey of Egypt*, Egypt Exploration Society, London.

Meeks, D. (2017), Ancient Egyptian composite hieroglyphs: a typology, in *Eikones workshop*, Basel.

Nederhof, M.-J. (2013), The Manuel de Codage encoding of hieroglyphs impedes development of corpora, in S. Polis & J. Winand (eds.), *Texts, Languages & Information Technology in Egyptology*, Presses Universitaires de Liège, (103–110).

Nederhof, M.-J., V. Rajan, J. Lang, S. Polis, S. Rosmorduc, T.S. Richter, I. Hafemann, & S. Schweitzer (2016a), A comprehensive system of control characters for Ancient Egyptian hieroglyphic text (preliminary version), `http://www.unicode.org/L2/L2016/16177-egyptian.pdf`.

Nederhof, M.-J., V. Rajan, J. Lang, S. Polis, S. Rosmorduc, T.S. Richter, I. Hafemann, & S. Schweitzer (2017), A system of control characters for Ancient Egyptian hieroglyphic text (updated version), `http://www.unicode.org/L2/L2016/16210r-egyptian-control.pdf`.

Nederhof, M.-J., V. Rajan, T.S. Richter, I. Hafemann, S. Schweitzer, S. Polis, & S. Rosmorduc (2016b), Comments on three control characters for Egyptian hieroglyphs, `http://www.unicode.org/L2/L2016/16090-comment-ctl-char-egyptian.pdf`.

Polis, S. (2018), The functions and toposyntax of Ancient Egyptian hieroglyphs, *SIGNATA* 9:291–363.

Polis, S., L. Desert, P. Dils, J. Grotenhuis, V. Razanajao, T.S. Richter, S. Rosmorduc, S.D. Schweitzer, D.A. Werning, & J. Winand (2020), The Thot Sign List (TSL) – an open digital repertoire of hieroglyphic signs, *Égypte Nilotique et Méditerranénne* 13, to appear.

Richmond, B. & A. Glass (2016), Proposal to encode three control characters for Egyptian hieroglyphs, `https://www.unicode.org/L2/L2016/16018r-three-for-egyptian.pdf`.

Stief, N. (1985a), Hieroglyphen, Koptisch, Umschrift, u.a. – ein Textausgabesystem, *Göttinger Miszellen* 86:37–44.

Stief, N. (1985b), A programm system for the edition of text, in *Informatique et Égyptologie 1*, Paris, (197–208).

Stief, N. (1988), Weitere Möglichkeiten bei der Hieroglyphenausgabe via Computer, in *Informatique et Égyptologie 5*, (46–51).

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