Handbook of Digital Egyptology: Texts

UAH MOA 01

Handbook of Digital Egyptology: Texts

Edited by Carlos Gracia Zamacona & Jónatan Ortiz-García

Universidad de Alcalá
EDITORIAL UNIVERSIDAD DE ALCALÁ

Con el patrocinio de la Comunidad de Madrid, Proyecto *The Earlier Ancient Egyptian Mortuary Texts Variability* (www.mortexvar.com), Programa Atracción de Talento 1 (2018-T1/HUM-10215).

Plaza de San Diego, s/n
28801 Alcalá de Henares
www.uah.es

I.S.B.N.: 978-84-18979-09-5
Depósito legal: M-34533-2021

Composición: Solana e Hijos, A. G., S.A.U.
Impresión y encuadernación: Solana e Hijos, A.G., S.A.U.
Impreso en España

Table of contents

2. Formatting of ancient Egyptian hieroglyphic text

Mark-Jan Nederhof
University of St Andrews

Abstract
For encoding of ancient Egyptian hieroglyphic text, a central issue is the choice of primitives to express how hieroglyphs are positioned relative to one another. Different such primitives have been used for preparing printed documents and for compiling electronic corpora.
Recently, nine control characters were introduced in Unicode for formatting of hieroglyphic text. The adequacy of such control characters can be determined by systematically encoding texts from different periods.

Keywords
writing systems – corpora – document preparation – font technology – Unicode.

1. Theory

1.1. Relative positioning of signs

Ancient Egyptian writing holds an exceptional position among the writing systems of the world.[1] It is 'complete' in that it allows any word in the language to be expressed;[2] see also Klinkenberg.[3] Among all complete writing systems, it is among

[1] Daniels – Bright 1996.
[2] Sampson 1985.
[3] Klinkenberg 2018.

the oldest, next to Sumerian cuneiform, and the period of its continuous use is among the longest, next to Chinese.

The main form of ancient Egyptian writing was the hieroglyphic script. From this, two cursive scripts were derived, known as hieratic and demotic. The hieroglyphic script continued to be used next to the cursive scripts throughout much of the history of ancient Egypt, albeit generally for different kinds of documents.[4]

Most writing systems can be traced back to pictographic origins. Later stages of development tended to simplify the pictograms to more abstract forms. Ancient Egyptian hieroglyphs however remained recognizably figurative. In addition, the repertoire of hieroglyphs was to some extent open-ended, and the appearance of common hieroglyphs could be varied, motivated by extralinguistic considerations, some of which could be related to the decorative or religious functions of inscriptions. This poses severe challenges to the encoding of hieroglyphs, henceforth signs for short; see further Chapter 3.

Another uncommon feature is that the arrangement of signs involves two dimensions, horizontal and vertical. Text can be written in columns, top-to-bottom, or in rows (i.e. lines), either left-to-right or right-to-left. The choice between left-to-right and right-to-left is apparent from the orientation of the individual signs; ancient Egyptian has this in common with Anatolian hieroglyphs.[5] If text is written in columns, then narrow signs may also be arranged next to one another. If text is in rows, then flat signs may also be arranged below one another. Furthermore, one may find horizontal combinations of signs that appear within vertical arrangements, which are themselves combined horizontally with other vertically arranged groups of signs, etc. We will use the term tabular arrangement to refer to signs being arranged horizontally and/or vertically.

If we look at original ancient Egyptian hieroglyphic texts, we witness one overarching guiding principle, namely aesthetics. The available surface of a wall, a stela, or a sheet of papyrus is filled with signs in a way that is pleasing to the eye, in particular, that avoids overly large empty spaces between signs. Where signs were arranged horizontally or vertically, these arrangements ultimately served the purposes of aesthetics, and were likely not a goal in themselves. In fact, much of the layout of signs in typical monumental inscriptions cannot be described solely in terms of tabular arrangement. Non-tabular arrangements of signs include various types of composition.[6]

It should be borne in mind that typical arrangements of signs are highly productive. That is, if a sequence of signs can be arranged in a certain way, then that arrangement could in principle also exist for another sequence of the same length in

---

[4] Davies – Laboury 2020.

[5] Werner 1991.

[6] Fischer 1977; Meeks 2017; Polis 2018.

which the signs, pair-wise, have similar dimensions and outlines, provided of course that the sequence of signs makes linguistic sense. As a consequence, it is not feasible to exhaustively list all potential groups of signs, and each newly investigated text may reveal groups of signs that were not attested before. This distinguishes ancient Egyptian from complex writing systems in which a bounded number of basic signs can be combined with a bounded number of diacritic signs in a relatively small number of ways, as for example in various South Asian scripts.

1.2. Encoding of relative positioning

Encoding of the relative positioning of signs in hieroglyphic texts may be primarily seen as a technological problem. The central challenge is to design a repertoire of machine-interpretable primitives that is powerful enough to express the relative positioning seen in most typical texts, to a reasonable degree of faithfulness. These primitives need not correspond however to any codified rules applied by the ancient artists in order to compose aesthetically pleasing inscriptions.

The earliest kinds of formatting of hieroglyphic text relied on substantial manual effort to determine appropriate scaling and positioning, which had to be done for each group of signs individually.[7] This could also be done manually for a given pattern of tabular arrangements of signs, assuming a bounded number of such patterns.[8] A competing approach was to automate some of the effort by introducing higher-level primitives for horizontal and vertical arrangement. Two of the earliest encodings that included such primitives were tied to two different and independently developed software packages, called GLYPH and PLOTTEXT.

GLYPH allows encoding of a hieroglyphic text consisting of a sequence of rows and/or columns.[9] Encodings of groups of signs within a row or column are separated by the '-' operator. Within the encoding of a group, there can be zero or more occurrences of the two binary operators '*' and ':'. The first arranges its arguments horizontally, and the second arranges its arguments vertically. The '*' has higher operator precedence than ':' and this can be overridden by pairs of parentheses. There are further primitives for cartouches and for shading (also known as hatching) of signs or of groups of signs, to indicate that these are damaged in the original inscription.

PLOTTEXT uses a notably different encoding for hieroglyphic text.[10] It allows underspecification of tabular arrangements, in that a simple sequence of signs

---

[7] James 1985; Diop 1992.

[8] Lesko 1974.

[9] Buurman – Schimmelpenninck van der Oije 1976; Buurman 1985; Buurman – de Moel 1987; Buurman 1988a; van den Berg 1993; van der Plas 1985.

[10] Stief 1985a; 1985b; 1988a; 1988b; 2001.

separated by space symbols is formatted through heuristics determined by the software, which considers the sizes of the signs and in particular the ratios of their widths and heights. For example, two neighbouring signs that are low and wide may possibly be placed one above the other, and two neighbouring signs that are high and narrow may be placed one next to the other.

TABLE 1. ENCODINGS OF HIEROGLYPHIC TEXT

	PLOTTEXT	MdC	RES	Unicode
𓊪𓐱𓏏𓐰𓇯	`p t pt` *or* `"p/t,pt"`	`p*t:pt`	`p*t:pt`	𓊪𓐱𓏏𓐰𓇯
𓋹𓈖𓐰𓐍	`S34 "N35,Aa1"`	`S34-N35:Aa1`	`S34-N35:Aa1`	𓋹𓈖𓐰𓐍
𓆓𓐶𓎛	`D=H`	`D#H`	`stack(D,H)`	𓆓𓐶𓎛
𓅭𓐴𓇳	`zA;/ra;`	`zA&ra`	`insert[te](zA,ra)`	𓅭𓐴𓇳
𓄂𓐲𓏏	`F4;t//;`	`F4&t`	`insert[ts](F4,t)`	𓄂𓐲𓏏
𓅱𓐳𓏏	`w;t/;`	`t&w`	`insert[s](w,t)`	𓅱𓐳𓏏
𓆓𓐳𓐷𓈖𓐰𓂧𓐸	`D;n,d;`	`D&n&d`	`insert[bs](D,n:d)`	𓆓𓐳𓐷𓈖𓐰𓂧𓐸
𓀔𓐵𓏪	`A17;//Z2b;`	`A17&Z2`	`insert[be](A17,Z2a)`	𓀔𓐵𓏪
𓅱𓐳𓏏𓐴𓏏	`w;t/;;/t;`	`t&w&t`	`insert[te](insert[s](w,t),t)`	𓅱𓐳𓏏𓐴𓏏

Tabular arrangement can also be specified explicitly, by enclosing a group within a pair of double-quotes, and then using the binary operators ‘,’ for vertical arrangement and ‘/’ for horizontal arrangement. (This is in a row of text, while the meanings of ‘,’ and ‘/’ are swapped in a column of text.) A space symbol within such a group leaves the tabular arrangement underspecified, much as before. Outside a pair of double-quotes, ‘/’ acts as an explicit separator of groups. The space symbol has higher precedence than ‘/’, which has higher precedence than ‘,’. An omitted argument of ‘,’ creates padding with empty space within a group, and there are further primitives for empty spaces of particular sizes, as well as ‘negative space’.

PLOTTEXT also has primitives for cartouches and similar enclosing signs, as well as for shading. The ‘=’ operator creates an overlay (also known as stacking) of two signs. Mirroring of a sign (horizontal inversion) is possible by inverting the font within an encoding.

In addition, there are six ‘insertion’ primitives. Such a primitive takes a ‘base’ sign and a group of signs that is inserted somewhere within the bounding box of the base sign. One such primitive is the ‘center insertion’, which inserts the group in the center of the base sign. Then there are four ‘corner insertions’, which insert the group in the top-left, top-right, bottom-left, or bottom-right corner, respectively. One more insertion is applicable if the base sign is a bird, and it inserts the group in front, just above the bird's feet.

Another innovative feature of PLOTTEXT is the encoding of a composite sign in terms of its parts. For example, a king wearing a white crown holding a flail is represented by codes for a king, a white crown, and a flail, respectively, separated by the '+' operator.

A tool developed at the University of Oxford uses an encoding that has at least one feature in common with PLOTTEXT, namely operators for corner insertion, or to be precise, one for the bottom-left corner and one for the top-right corner of a base sign.[11] Groups are enclosed in braces, binary operator '/' achieves vertical arrangement, whitespace (or juxtaposition) achieves horizontal arrangement, and a pair of ';' delimits a nested group. Whitespace has higher precedence than the '/'. The encoding also has primitives for cartouches, empty spaces, and 'negative space'.

It appears that a majority of the user community at the time favoured further development in the direction of GLYPH.[12] PLOTTEXT continued to be used however, for example by Beinlich & Saleh[13] and Graefe.[14] Some other encodings, such as those proposed by Marti & Piolle,[15] do not seem to have been pursued further.

There were subsequent efforts to document the encoding used by GLYPH, resulting in a publication commonly referred to as the Manuel de Codage, or MdC for short.[16] The MdC also includes the '\' operator for mirroring signs, the '#' operator for overlaying two signs, text-critical markers, and primitives for leaving empty space within groups, and for a number of enclosing signs akin to cartouches such as serekhs and walled enclosures. The tradition of the MdC lived on in successors of GLYPH, such as WinGlyph and MacScribe.[17]

Many typical cases of relative positioning cannot be faithfully described in terms of tabular arrangement alone. Corner insertions as they exist in PLOTTEXT effectively cover a fair portion of those remaining cases. The MdC as last published does not include any primitives for corner insertion however.[18]

One solution is to introduce a new code for each combination of signs that may be arranged by corner insertion. Such an atomic encoding is for example 'O38a' on p. 9 of Buurman & Schimmelpenninck van der Oije for .[19] Another solution is to

[11] Baines – Griffin 1988.
[12] Buurman et al. 1985; Buurman 1988b; Grimal 1990.
[13] Beinlich – Saleh 1989.
[14] Graefe 1994.
[15] Marti – Piolle 1985.
[16] Buurman et al. 1988.
[17] Gozzoli 2013.
[18] Stief 1988a.
[19] Buurman – Schimmelpenninck van der Oije 1976.

reinterpret the operators of tabular arrangement if applied on certain combinations of signs. It appears that this was attempted in GLYPH for at least a period of time. For example, 'D:Y1' stood for  rather than the expected .[20]

Another solution is found in WinGlyph.[21] A combination of signs separated by the new binary operator '&' is arranged in some non-tabular fashion, which could be corner insertion. Examples are 'wr&t' and 'G14&X1', which are both rendered as an insertion in the top-right corner:  and , respectively. The general meaning of '&' is underspecified however. In other cases it may correspond to, for example, insertion into one of the other three corners. Users can moreover define new sequences of signs combined with '&', to mean an arbitrary relative positioning. WinGlyph further includes primitives for rotation and scaling of signs.

Because corner insertions are highly productive, none of the three above-mentioned solutions is able to handle this form of relative positioning in its full generality. In particular, if a new example of corner insertion of two signs is found, then a new atomic encoding would have to be introduced, or ':' or '*' would have to be reinterpreted for a new combination of signs, or the meaning of '&' would have to be defined for a new combination of signs, respectively. In each case, the encoding is unstable, and exchange of texts between different tools and different users is problematic.

Another solution to corner insertion appears to have originated in MacScribe.[22] It involves the binary operators '^^^' and '&&&'. Each takes as arguments a base sign and a group that is inserted in the base sign. Any sign that can act as a base sign has up to two zones, zone 1 and zone 2, which can be corners or other areas in or around the base sign. In the case of '^^^', the base sign is the second argument and the group is the first argument, which is inserted in zone 1 of the base sign. In the case of '&&&', the base sign is the first argument, and the group is the second argument, which is inserted in zone 2 of the base sign. The two operators can be combined, with the base sign in the middle, whereby two groups are inserted in one base sign. For example, 't^^^w&&&Z2' inserts 't' just above the feet of 'w', and inserts 'Z2' in the upper-right corner, to give . The two primitives together still limit insertion to two zones, and furthermore, it is not apparent from the shape of a base sign alone where the two zones are located.

JSesh is a modern implementation of the MdC.[23] It adds a primitive allowing absolute positioning and scaling. For example, 'S34\R30{{0,357,51}}**G5{{194,0,97}}' expresses that sign 'S34' is to be rotated by 30 degrees, scaled by factor 0.51, and placed at (x, y) coordinate (0.0, 0.357), while 'G5' is scaled by factor 0.97 and placed

[20] Van den Berg 1988: 33.
[21] Van den Berg 1997.
[22] S. Rosmorduc, personal communication, May 15, 2020.
[23] Rosmorduc 2019.

at coordinate (0.194, 0.0); coordinates refer to the top-left corners of bounding boxes of signs. This may appear as . In this syntax, '**' connects a number of signs together that are formatted by absolute scaling and positioning relative to the same reference point (0.0, 0.0). If the triple behind a sign is absent, it defaults to '{{0,0,100}}'.

Absolute scaling and positioning is particularly appropriate for fine-grained, palaeographic reproduction of the original appearance of an inscription. It can be used to express corner insertion, among many other kinds of relative positioning. However, encoding in this way is time-consuming, and if a font is replaced by another in which dimensions and outlines are slightly different, then an existing encoding may no longer result in an acceptable rendering.[24]

The Revised Encoding Scheme (RES) for hieroglyphic text is intended to satisfy a wide range of applications, from lexicography to palaeography, while at the same time creating good prospects for the longevity of encoding.[25] It has a small number of very powerful primitives, whose syntax is specified by a formal grammar, and whose meaning is defined in a simple and self-contained manner, making the encoding independent from any implementation or font. It was demonstrated that these primitives can be captured in a natural way by applying image processing on scans of transcriptions.[26]

In RES, the operators '-', '*', ':' and the parentheses have the same meaning as they have in the MdC. In addition, there is one general insertion primitive, which can be parameterized with the location of the inserted group, which can be in one of the four corners, at one of the four sides, in the middle, or at an arbitrary specified coordinate. The exact scaling and positioning of the inserted group is determined dynamically by the actual shapes of the signs in the used font. Conceptually, the inserted group is initially scaled down to close to a single pixel and placed at an appropriate initial position. The group is then gradually scaled up, and its position is adjusted as appropriate, to make it fit exactly in the indicated place, respecting a minimum distance between any of its pixels and the pixels of the base sign (or more generally, base group). RES further has primitives for overlaying, for cartouches and related enclosing signs, for shading with arbitrary granularity, footnote markers, and empty spaces.

In order to satisfy palaeographic applications, default parameters of the primitives can further be adjusted to fine-tune the appearance. For example, the default distance between signs (or more generally between subgroups) can be increased or decreased as desired, by setting the 'sep' argument of the relevant operator, be it '-', '*' or ':'. There is no 'negative space' in RES. Instead, an operator may be given the argument

[24] Nederhof 2013.
[25] Nederhof 2002; 2019b.
[26] Nederhof 2015.

'fit', whereby the distance between subgroups is determined by a dynamic form of kerning, i.e. the actual shapes of subgroups are inspected, and the subgroups are squeezed together until a minimal distance between pixels of the two subgroups is reached.

The expressive power of RES comes at a cost. Implementation is relatively difficult, and rendering is slow. This is mainly because corner insertion iteratively investigates different scalings and positionings, to determine how to exactly fit the inserted group within the base group. Dynamic kerning is relatively time-consuming as well, as it investigates shapes on a pixel level.

1.3. Unicode

Unicode creates the possibility to represent text in any writing system in a consistent way, including both modern and historical scripts. It assigns a unique code point to each character. A character is to be distinguished from a glyph, which is a particular realization of a character. Some code points stand for control characters, which do not have a visual representation on their own, but influence the appearance of the visible characters.

The earliest attempt to introduce primitives into Unicode for relative positioning of ancient Egyptian hieroglyphs took operators directly from the Manuel de Codage.[27] This attempt was discontinued.

The second attempt only involved three control characters, which were the equivalents of '*', ':' and '&' from the MdC.[28] Objections were raised against the last of these three control characters, due to its undefined and inherently undefinable meaning,[29] and an alternative way forward was proposed.[30] This resulted in a repertoire of nine control characters,[31] which became part of Unicode 12 in March 2019.

The control characters are listed in Table 2, followed by their syntax, specified in Backus-Naur form. In this notation, the pipe symbol '|' separates alternatives. Square brackets '[' and ']' enclose optional elements, round brackets followed by a plus symbol '( )+' enclose elements that are repeated one or more times. A 'fragment' is a valid sequence of Unicode characters that represents a row or column of text, consisting of hieroglyphs and control characters. A 'sign' can be any of the existing 1071 ancient Egyptian hieroglyphs currently in Unicode.

---

[27] Everson 1997; 1999.
[28] Richmond – Glass 2016.
[29] Nederhof et al. 2016b.
[30] Nederhof et al. 2016a; 2017.
[31] Glass et al. 2017; Everson – Glass 2018; Nederhof 2018; 2019a.

TABLE 2. THE CONTROL CHARACTERS AND THEIR SYNTAX

13430	:	EGYPTIAN HIEROGLYPH VERTICAL JOINER
13431	*	EGYPTIAN HIEROGLYPH HORIZONTAL JOINER
13432	𓐲	EGYPTIAN HIEROGLYPH INSERT AT TOP START
13433	𓐳	EGYPTIAN HIEROGLYPH INSERT AT BOTTOM START
13434	𓐴	EGYPTIAN HIEROGLYPH INSERT AT TOP END
13435	𓐵	EGYPTIAN HIEROGLYPH INSERT AT BOTTOM END
13436	𓐶	EGYPTIAN HIEROGLYPH OVERLAY MIDDLE
13437	(	EGYPTIAN HIEROGLYPH BEGIN SEGMENT
13438	)	EGYPTIAN HIEROGLYPH END SEGMENT

```
fragment ::= ( group )+
group ::= vertical_group | horizontal_group | basic_group

vertical_group ::= ver_subgroup ( : ver_subgroup )+
ver_subgroup ::= horizontal_group | basic_group
horizontal_group ::= hor_subgroup ( * hor_subgroup )+
hor_subgroup ::= ( vertical_group ) | basic_group

basic_group ::= core_group | insertion_group
insertion_group ::= core_group insertions
insertions ::= 𓐲 in_group [ 𓐳 in_group ] [ 𓐴 in_group ] [ 𓐵 in_group ]
               𓐳 in_group [ 𓐴 in_group ] [ 𓐵 in_group ]  |
               𓐴 in_group [ 𓐵 in_group ]  |
               𓐵 in_group
in_group ::= ( vertical_group )  |
             ( horizontal_group )  |
             ( insertion_group )  |
```

The control characters ':', '*' and the parentheses have the same meaning as in the MdC. Then there are four corner insertions. If the base sign is a bird, then insertion in the bottom-left corner (for left-to-right text) amounts to placing the inserted group just above the feet. In other words, two related insertion primitives from PLOTTEXT are here merged into one. The names of the inserting control characters have 'START' and 'END' in them rather than left and right; for right-to-left text the 'start side' is at the right and the 'end side' is at the left.

Finally, there is an overlaying control character. Its first argument, a 'flat_hor_group', can be a single sign or a sequence of two or more signs between parentheses arranged horizontally. Similarly, its second argument, a 'flat_ver_group', can be a single sign or a sequence of two or more signs between parentheses arranged vertically. The base in an 'insertion_group' is a 'core_group', which can be a single sign or an overlay. This is followed by one or more of the inserting control characters, each followed by the inserted group, the 'in_group', which can be a 'core_group' itself, or another kind of group enclosed in parentheses. If several inserting control characters follow the same base, then these must occur in a fixed order.

Current font technology was not designed for writing systems such as ancient Egyptian. One example of a challenging task is scaling down of two tall signs that are arranged one above the other, depending on the summed height of the two signs including the space in between, relative to the available height within a row of text. Font technology such as OpenType is unable to perform addition and division of arbitrary numbers, nor can it dynamically scale down glyphs in a font. This means that for formatting ancient Egyptian, workarounds need to be found, such as manipulating a bounded number of possible widths and heights, and including a bounded number of scalings of each sign within a font. Such workarounds are likely to be too crude to result in an ideal rendering. Another approach is to prepare a font specifically for one text, using external tools that are capable of the necessary computation and scaling to produce preformatted groups, which are stored in a font as single composite glyphs. This generally results in a rendering of higher quality, but it is a less flexible approach, due to the need for the external tools, and the need to recreate the font whenever the text changes.

1.4. Formatting

Because of the limitations of font technology mentioned above, Unicode does not specify in detail how hieroglyphic text is to be formatted. However, one would expect at least a rough approximation of the ideal formatting as it exists in say JSesh and RES. We explain the ideal formatting of an outermost group in terms of an abstract syntax, defined inductively by:

- A group is a horizontal group, a vertical group, or a basic group.
- A vertical group consists of a list of two or more subgroups to be arranged vertically, each of which is a horizontal group or a basic group.
- A horizontal group consists of a list of two or more subgroups to be arranged horizontally, each of which is a vertical group or a basic group.

- A basic group has a core subgroup, which is either a single sign or an overlay, and, for each of the four corners, optionally a group to be inserted in that corner.
- An overlay consists of two lists, each consisting of one or more signs. The signs in the first list are to be arranged horizontally and the signs in the second list are to be arranged vertically, and the two lists are then stacked upon one another, with their center points coinciding.

Formatting of an outermost group is done in two steps:

- First, signs in that group are scaled down if necessary to fit within the available space. This process starts with the smallest nested subgroups.
- Second, as much whitespace as possible is added between subgroups without increasing the total surface area taken up by the outermost group.
- In the following, we describe this scaling and padding in more detail, assuming a row of text. Formatting of a group within a column of text is analogous.

1.4.1. *Scaling*

The space available to a group is determined by a maximum width and a maximum height. These maxima are either ∞ (infinity) to indicate there is no restriction, or are positive numbers no greater than 1 EM, where EM is the height of a row of text (and of the unscaled ‘sitting man’ hieroglyph).

An outermost group has maximum height 1 EM and maximum width ∞. The latter means that if this outermost group is an horizontal group consisting of many basic groups or vertical groups next to one another, then it can have any width. However, the maximum width of the subgroups of a vertical group is 1 EM and the maximum height of the subgroups of an horizontal group is 1 EM. The maximum width and maximum height of a group inserted in a corner of a basic group is determined by the shape of the core subgroup; this will be discussed in more detail later.

The width of an horizontal group is the sum of the widths of the subgroups, plus a certain distance between consecutive subgroups, and its height is the maximum of the heights of the subgroups. The height and width of a vertical group are defined analogously.

The total width and height of an overlay are the maxima of the widths and heights, respectively, of the horizontal list and the vertical list. In the formatting, the geometric center of the horizontal list coincides with the geometric center of the vertical list.

If the width and/or height of a group exceed the available space, then all signs within it are scaled down, including the distances between consecutive subgroups, to exactly fit within the available space. Note that a single sign may be scaled down multiple times, as it may be part of a group that in turn is a subgroup of a larger group, etc., and each of these larger and larger groups may trigger down-scaling of all the signs in them. Signs are never scaled up, so they never appear bigger than their 'natural' size in the font.

1.4.2. *Padding*

After scaling, whitespace is added inside groups, to center individual signs, spread out horizontally the subgroups of an horizontal group, and spread out vertically the subgroups of a vertical group. More precisely, groups are formatted within a rectangle whose size is computed after scaling as described earlier. For an outermost group, the height of this rectangle is the height of a line, so 1 EM, and the width is the width of that group.

A horizontal group that is to be formatted within a certain rectangle is divided into smaller rectangles horizontally, one for each subgroup, and one for each occurrence of whitespace between subgroups; the width of the rectangle of a subgroup is the width of that subgroup, and leftover horizontal whitespace is equally divided over the whitespace that occurs between the subgroups. The formatting of a vertical group is analogous.

A group that is inserted in a basic group is formatted within a rectangle determined by the shape of the core subgroup.

A sign that is to be formatted within a certain rectangle that is bigger than the sign itself is centered horizontally and vertically, that is, any leftover horizontal whitespace is divided equally over padding to the left and to the right of the sign, and any leftover vertical whitespace is divided equally over padding above and below.

1.4.3. *Insertion*

In a typical implementation of corner insertion, the available spaces for the corners are rectangles. Such a rectangle can be fixed for each individual sign that may occur as core subgroup and for each corner where one might expect an insertion into that sign. The rectangle can be determined manually or automatically by analyzing the shape of the sign in a font. The rectangle need not be entirely enclosed in the bounding box of the sign, and typically it extends to the left or to the right. In effect that means an extra strip of space is added on the left or on the right side of a sign if there is an insertion on that side, as in the case of .

Determining appropriate rectangles for corner insertions is more challenging for the case of a core subgroup that is an overlay. This is because the number of overlays is in principle unbounded, and in practice prohibitively large. Analyzing shapes of combinations of signs, once and for all for a given font, is therefore not viable. A more practical solution is to assume that the available space for each corner insertion has some fixed size, say 0.3 EM × 0.3 EM, and extends by a certain distance, say by 0.2 EM, to the left or to the right of the bounding box. In very few of the already rare cases of a combination of overlay and corner insertion would this lead to an undesirable rendering.

1.5. *Outlook*

The current repertoire of control characters is work in progress. For future extensions of this repertoire, the needs of Egyptologists need to be balanced against the technical challenges posed by further control characters. Furthermore, Unicode is specifically not meant to express visual appearance of texts with a high degree of faithfulness, which means that such matters as kerning are outside its scope.

It is expected that center insertion as well as cartouches and other enclosing signs will be revisited, in preparation for a future version of Unicode.[32] Because complete and undamaged texts are the exception in Egyptology, there is also an urgent need for encoding of empty space and shading.

The repertoire of control characters is not entirely independent from the issue of the sign list. For example, there are code points for composite signs that can since recently be encoded using control characters, and similar signs will no longer require an atomic encoding in the future. In the same vein, mirrored and rotated signs would not require separate code points if there were suitable control characters for mirroring and rotation. This needs to be weighed against the challenges of implementing mirroring and rotation in font technology.

2. Examples

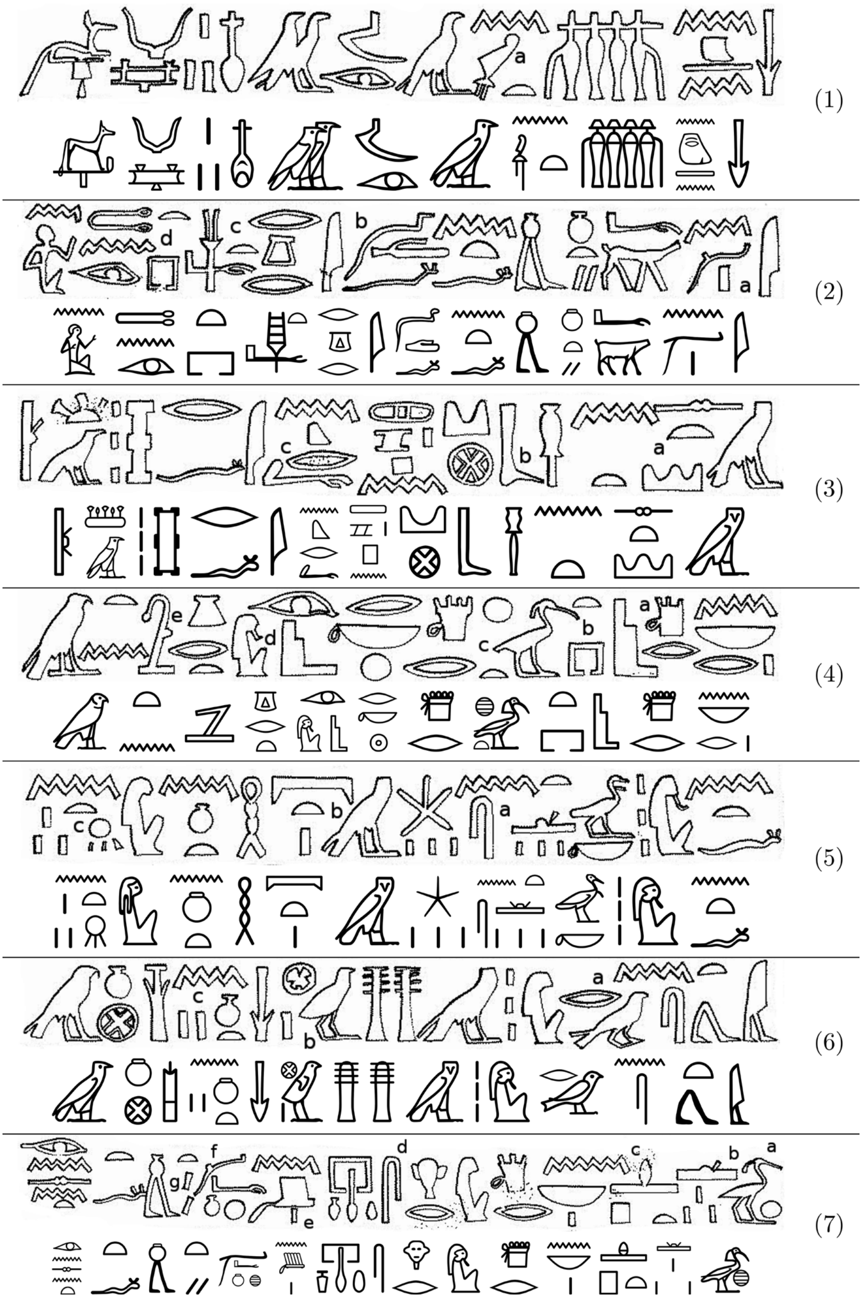
The adequacy of the control characters has been subjected to critical evaluation by systematically encoding texts from different periods. Here, two Middle-Egyptian stelas are investigated, both from the Twelfth dynasty.

---

[32] Polis 2018: 344.

2.1. Intef son of Senet

Table 3. Stela of Intef son of Senet (BM EA581), with kind permission from Richard Parkinson and ©Trustees of the British Museum



(1)
(2)
(3)
(4)
(5)
(6)
(7)

Table 3 shows line drawings of consecutive rows followed by the closest representation that is achievable in Unicode. There are many examples of corner insertion, such as (2a-b) and (7a). At (2b), one may well argue that also the viper is inserted. There are two insertions in one base sign at (4c) and (6b), although in the latter case one may equally well argue that the stroke follows the chick, rather than being inserted into it. At (2c), there is a corner insertion into an overlay; the overlay here happens to be also available as atomic sign in Unicode. One may furthermore interpret the 𓏏 near the top-left of the overlay as another corner insertion. At (7e), the original inscription is closer to a kind of insertion, but here the subgroup is encoded with vertical arrangement. At (7f) it is not clear whether one sign is inserted, with two more signs below, or whether all three signs are inserted as in the presented encoding. This reminds us that primitives of encoding are modern concepts, which can at best approximate original inscriptions.

There are examples of tabular arrangements that involve several levels of nesting, such as (5a) and (6c), where there is a vertical arrangement within a horizontal arrangement within a vertical arrangement. Another example of a deeply nested tabular arrangement is (5c).

At (3a) and (4a), the signs are squeezed towards one another. This can be encoded using the ‘fit’ feature in RES, and using negative space in some other types of encoding. Such ‘kerning’ is beyond the capabilities of Unicode however. There are further examples of ‘kerning’ in the line drawings at (2c-d), (3b-c), (4b), (5b) and (7b-c). At (4d), one may alternatively argue that the god is to the left of the eye, again with ‘kerning’, rather than underneath it. The relative positioning at (6a) is difficult to characterize in terms of the available control characters, and other normalized transcriptions may be equally valid.

Unicode currently cannot represent rotation nor mirroring, and the exact appearance at (1a) and at (7d) can at best be approximated. For some signs, the exact graphical variant may not be available, as at (4e) and (7g), and the unbearded gods in lines (6) and (7).

2.2. Ity and Iuri

TABLE 4. STELA OF ITY AND IURI (BM EA586), WITH KIND PERMISSION FROM RICHARD PARKINSON AND ©TRUSTEES OF THE BRITISH MUSEUM

(1)

(2)

(3)

(4)

In Table 4, there are corner insertions at (1b), (1d), (3a) and (3c). The 'kerning' at (4a) and (4c) could alternatively be encoded as corner insertion. There are several more cases where groups are squeezed together, such as (1c), (2a-b), (3b) and (4b).

The encoding at (1a) is little satisfactory, for two reasons. First, there is no consensus as yet on encoding of cartouches and other enclosing signs. In particular, it is to be decided whether this ought to be done in terms of a prospective center-insertion primitive. For now, we treat the open-cartouche and the close-cartouche as individual signs, here with the control characters for horizontal arrangement. Secondly, there is no solution as yet to encode that, for example, a 'sky' sign should stretch out to the width of the group below it. It is an open question whether this should be handled in the same way as cartouches stretch out proportionally to the width (or height, for columns of text) of the enclosed group.

3. Acknowledgements

Serge Rosmorduc provided vital assistance during writing of this chapter, by providing scans of documents and contributing first-hand knowledge of historical developments. Stéphane Polis gave substantial feedback on an earlier version, which was incorporated in the present text.

4. References

Baines, J. – Griffin, C. (1988). "Automated typesetting of the Gardiner hieroglyphic font". In *Informatique et Égyptologie* 4: 85–94.

Beinlich, H. – Saleh, M. (1989). *Corpus der hieroglyphischen Inschriften aus dem Grab des Tutanchamun*. Oxford.

van den Berg, H. (1988). "Textual analysis on computer: Some additions to the encoding system of the manual". In *Informatique et Égyptologie* 4: 32–43.

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: 5/8/2020].

Buurman, J. (1985). "Printing of Egyptian hieroglyphs by means of a computer". In *Informatique et Égyptologie* 1, Paris: 13–17.

Buurman, J. (1988a). "GLYPH 2.1 – the continuing story". In *Informatique et Égyptologie* 5: 4–7.

Buurman, J. (1988b). "The program GLYPH: Printing of Egyptian hieroglyphs by means of a micro computer". In *Informatique et Égyptologie* 4: 6–16.

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

Buurman, J., Grimal, N., Hainsworth, M., Stief, N., Vergnieux, R. & van der Plas, D. (1985). "Resolution de la table ronde du CNRS – informatique appliquée a l'égyptologie". In *Informatique et Égyptologie* 1, Paris: 225.

Buurman, J. – de Moel, E. (1987). "GLYPH – computer programs for Egyptologists". In *Informatique et Égyptologie* 3: 4–7.

Buurman, J. – Schimmelpenninck van der Oije, A.H.J. (1976). "The composing of hieroglyphic texts by means of a computer". *GM* 19: 7–15.

Daniels, P.T. – Bright, W. (eds.) (1996). *The World's Writing Systems*. New York.

Davies, V. – Laboury, D. (eds.) (2020). *The Oxford Handbook of Egyptian Epigraphy and Palaeography*. Oxford.

Diop, C.M. (1992). "Hiéroglyphes et informatique". *ANKH* 1: 105–119.

Everson, M. (1997). *Proposal to encode basic Egyptian hieroglyphs in Plane 1*, <https://www.unicode.org/L2/L1997/97267-n1637.pdf> [accessed: 5/22/2020].
Everson, M. (1999). *Encoding Egyptian hieroglyphs in Plane 1 of the UCS*, <https://www.unicode.org/L2/L1999/N1944.pdf> [accessed: 5/22/2020].
Everson, M. – Glass, A. (2018). *Glyphs for Egyptian hieroglyphic control characters*, <https://www.unicode.org/L2/L2018/18036-n4926-egyptian-controls.pdf> [accessed: 5/22/2020].
Fischer, H.G. (1977). "The evolution of composite hieroglyphs in Ancient Egypt", *Metropolitan Museum Journal* 12: 5–19.
Glass, A. – Hafemann, I. – Nederhof, M.-J. –Polis, S. – Richmond, B. – Rosmorduc, S. –Schweitzer, S. (2017). *A method for encoding Egyptian quadrats in Unicode* <http://www.unicode.org/L2/L2017/17112r-quadrat-encoding.pdf> [accessed: 5/22/2020].
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*, Liège: 89–101.
Graefe, E. (1994). *Mittelägyptische Grammatik für Anfänger*. Wiesbaden.
Grimal, N. (1990). "Hiéroglyphes et ordinateurs". *Bulletin de recherches sur l'information en sciences économiques humaines et sociales* 15: 57–60.
James, T.G.H. (1985). "Computer printing of hieroglyphs at the University Press Oxford". In *Informatique et Égyptologie* 1: 81–92.
Klinkenberg, J.-M. (2018). "Entre dépendance et autonomie. pour une définition de l'écriture dans les sciences du langage et du sens". *SIGNATA* 9: 103–129.
Lesko, L.H. (1974). "Brief report on the computer printing of hieroglyphs". *GM* 14: 17–20.
Marti, B. – Piolle, J. (1985). "Une approche globale des traitements informatiques des documents complexes et son application a l'égyptologie". In *Informatique et Égyptologie* 1: 149–196.
Meeks, D. (2017). "Ancient Egyptian composite hieroglyphs: a typology". In *Egyptian and Maya Writing: Comparing Hieroglyphic Domains. International Conference June 9–11, 2017, Basel*.
Nederhof, M.-J. (2002). "A revised encoding scheme for hieroglyphic". In *Proceedings of the 14th Table Ronde Informatique et Égyptologie*, on CD-ROM.
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*, Liège: 103–110.
Nederhof, M.-J. (2015). "OCR of handwritten transcriptions of Ancient Egyptian hieroglyphic text". In *Altertumswissenschaften in a Digital Age: Egyptology, Papyrology and beyond. Proceedings of a conference and workshop in Leipzig, November 4–6, 2015*, <https://nbn-resolving.org/urn:nbn:de:bsz:15-qucosa-201704> [accessed: 5/22/2020].
Nederhof, M.-J. (2018). *A note on the syntax of Ancient Egyptian hieroglyphic control characters*, <https://www.unicode.org/L2/L2018/18236-nederhof.pdf> [accessed: 5/22/2020].

Nederhof, M.-J. (2019a). *A note on OpenType implementation of Ancient Egyptian hieroglyphic text*, <https://www.unicode.org/L2/L2019/19331-egyptian-opentype.pdf> [accessed: 5/22/2020].

Nederhof, M.-J. (2019b). *RES (Revised Encoding Scheme)*, <http://mjn.host.cs.st-andrews.ac.uk/egyptian/res> [accessed: 5/22/2020].

Nederhof, M.-J. – Rajan, V. – Lang, J. – Polis, S. – Rosmorduc, S. – Richter, T.S. – Hafemann, I. – Schweitzer, S. (2016a). *A comprehensive system of control characters for Ancient Egyptian hieroglyphic text (preliminary version)*, <http://www.unicode.org/L2/L2016/16177-egyptian.pdf> [accessed: 5/22/2020].

Nederhof, M.-J. – Rajan, V. – Lang, J. – Polis, S. – Rosmorduc, S. – Richter, T.S. – Hafemann, I. – Schweitzer, S. (2017). *A system of control characters for Ancient Egyptian hieroglyphic text (updated version)*, <http://www.unicode.org/L2/L2016/16210r-egyptian-control.pdf> [accessed: 5/22/2020].

Nederhof, M.-J. – Rajan, V. – Richter, T.S. – Hafemann, I. – Schweitzer, S. – Polis, S. – Rosmorduc, S. (2016b). *Comments on three control characters for Egyptian hieroglyphs*, <http://www.unicode.org/L2/L2016/16090-comment-ctl-char-egyptian.pdf> [accessed: 5/22/2020].

van der Plas, D. (1985). "Informatique appliquée a l'égyptologie". In *Informatique et Égyptologie* 1: 217–219.

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

Richmond, B. – Glass, A. (2016). *Proposal to encode three control characters for Egyptian hieroglyphs*, <https://www.unicode.org/L2/L2016/16018r-three-for-egyptian.pdf> [accessed: 5/22/2020].

Rosmorduc, S. (2019). *JSesh*, <http://jseshdoc.qenherkhopeshef.org> [accessed: 5/8/2020].

Sampson, G. (1985). *Writing Systems. A Linguistic Introduction*. Stanford.

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

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

Stief, N. (1988a). "Demonstration of a computer textoutput system for hieroglyphics, Coptic, Arabic, etc.". In *Informatique et Égyptologie* 4: 84.

Stief, N. (1988b). "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.

Werner, R. (1991). *Kleine Einführung ins Hieroglyphen-Luwische*. Freiburg – Göttingen.