

Arabic typesetting using a METAFONT-based dynamic font

Amine Anane
ananeamine@gmail.com

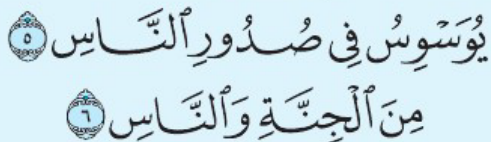
TUG 2019 - Palo Alto, California, USA
August 9-11, 2019

How the project started ?

I cannot find a searchable pdf document of the Quran.

PDF documents are raster images of hand-written book :

- cannot be searchable.
- big size.
- blurry image due to pixelation



Other formats (Office Word, HTML [<http://tanzil.net>]) :

- based on Unicode text file + OpenType Font.

OpenType Font vs Handwritten text

Much smaller font size to accommodate the widest line

وَيَتِيمًا وَأَسِيرًا ﴿٨﴾ إِنَّمَا نَطْعُمُكُمْ لِرُؤْفَةِ اللَّهِ لَا نُزِيدُ مِنْكُمْ جَزَاءً وَلَا نُكْفِرُكُمْ

وَيَتِيمًا وَأَسِيرًا ﴿٨﴾ إِنَّمَا نَطْعُمُكُمْ لِرُؤْفَةِ اللَّهِ لَا نُزِيدُ مِنْكُمْ جَزَاءً وَلَا نُكْفِرُكُمْ

Increase in interword space for other lines

التَّائِبُونَ الْعَابِدُونَ الْحَامِدُونَ السَّائِحُونَ

التَّائِبُونَ الْعَابِدُونَ الْحَامِدُونَ السَّائِحُونَ

Objective

Typeset an electronic copy of the Medina Mus'haf (book) respecting calligraphic rules and having the same Font size, page number and line

Why Medina book ?

- Quranic book is a reference in Arabic calligraphy
- Written by one of the most renowned Arabic calligrapher
- beautiful, clear, easy-to-read style
- each page finishes by a verse and each section (30 sections) in 20 pages (15 lines each)
 - Justification is applied extensively

If objective is reached, we should reach ultimate goal : High-quality Arabic typesetter fulfilling Arabic calligraphic rules

Where to start?

Existing systems and technology for typesetting Arabic text

- Font standards such as OpenType and Apple Advanced Typography
 - ▶ Advantage : Availability of tools
 - ▶ Disadvantage : justification does not conform to Arabic calligraphy such as horizontal stretching using Kashida (Tatweel)
- ditroff/ffortid system
 - ▶ Advantage : based on dynamic PostScript fonts
 - ▶ Disadvantage : does not model the way stretching is done in hand-written Arabic text
- Alqalam : An extension of Arab $\text{T}_{\text{E}}\text{X}$
 - ▶ Advantage : based on METAFONT to design a dynamic font
 - ▶ Disadvantage : Very difficult to define the different context-sensitive shaping rules

Where to start? (Cont'd)

- Oriental project which leads to the development of Lua \TeX : successor of Omega and pdf \TeX
 - ▶ Advantage : Can be customized and extended using Lua and have native support for OpenType fonts,
 - ▶ Disadvantage : does not consider dynamic fonts

Selected Framework :

- Lua \TeX and OpenType (i.e Con \TeX t Lua modules)
 - ▶ OpenType engine will be extended to support dynamic font
- METAFONT to design the dynamic Font
 - ▶ High level structure to describe curve (direction, tension, curl, ...)
 - ▶ Linear equations
 - ▶ Macros to extend the language with new syntax and functions
 - ▶ Integrates well with Lua \TeX (i.e mplib)

First attempt

- Have a beautiful font identical to the hand-written one : FontForge to trace the bitmaps to vector image and convert them to METAFONT
- Lua_TE_X internal OpenType tables to define substitution and positioning rules

Issues

- Takes a lot of time
- Difficult to debug
- Cannot take full advantage of METAFONT : Have to start from cubic curves with constant values

Solution

- Develop a graphical tool to easy the task : **Visual METAFONT**

Visual METAFONT

- Based on QT5 (C++)
- mplib : project supported by LuaT_EX team in order to turn MetaPost into system library that can be used by many different applications.
- Harfbuzz : Text shaping engine supporting OpenType and AAT used in Android, Chrome, ChromeOS, Firefox, GNOME, GTK+, KDE, LibreOffice, OpenJDK, PlayStation, Qt, XeTeX, etc

Visual METAFONT - Glyph View

The screenshot displays the Visual METAFONT software interface. On the left is a code editor showing METAFONT source code for the character 'r'. The central workspace shows the glyph 'r' rendered on a blue dot grid, with a black bounding box and several small colored dots representing control points. On the right is a Properties panel with various attributes and values.

```
metzoon.metz before:rh
File Edit View Items Help
Metzoon Code
font | Log | Text
%!FontName=metz before:rh-5-5-5-5-5
%!fontdir=C:\Programme\FontTools\METAFONT\out\metz\images
metzoon.metz:app-155-156-150-8-8-8-8
%!beginpreamble
  r1 = (200,60,72812);
  r2 = (524,2);
  r3 = (204,296,2);
  r4 = (48811,48,1122);
%!beginstatement
  %!chartr = 4; %!numeric
  leftHalfway = chartr - 0.5;
  rightHalfway = chartr + 0.5;
  %!fill_vertically_start = 0; %!fill_vertically_end = 88; 8
  %!fill_vertically_start = 0; %!fill_vertically_end = 88; 8
  %!fill_vertically_start = 0; %!fill_vertically_end = 88; 8
  left_bottom_corner = (1 + chartr * 0.115);
  %!fill_bottom_halfway = 1.7; %!fill_bottom_halfway = 1.7; 8
  r1 = 0;
  r2 = y1;
  %!x1 = x1;
  %!y1 = y1;
  %!x2 = x2;
  %!y2 = y2;
  rightBottom_corner = (x11 - x1, y11 - y1);
  leftBottom_corner = (x22 - x2, y22 - y2);
  r3 = r1 + (jornwidth,0) related 80;
  r4 = r3 + (jornwidth,0) related 70;
  %!chartr:baton
  %!beginpath
  P8
  (88,50) .. (11,204) .. tension 1 and 11,204 ..
  (11,0) ..
  (18,60) .. (19,246) .. tension 2,41666 and 0,79 ..
  (19,245,39,8456) .. tension 0,79 and 2,76079 ..
  (94) .. (20,115,115,377) .. tension -1,62821 and 1,90385 ..
  (42,75,115,377) .. tension 1,21842 and 10,000 ..
  ..
  ..
  ..
  (0,1) .. (60,476,0,2213) .. tension 1,03671 and 2,03600 ..
  (40) .. (42,357,38,7408) .. tension 1 and 2,76227 ..
  (24,35,0,49982) .. tension 1 and 1,6800 ..
  (0,1) .. (28,822,76,7282) .. tension 2,77468 and 3 ..
  ..
  ..
  P8
  (200,325,112,805) .. tension 3,20495 and 0,80462 ..
  (40) .. (40,25,68,7068) .. tension 3,05846 and 4,09143 ..
  (40) .. (25,154,154,51) .. tension 0,75 and 1,0549 ..
  (0,0) .. end
  ..
  %!handbaton
  %!beginpath
  leftHalfway = 0;
  setCharwidthCharWidth = L1,88chartr;
enddefChar;

```

Properties

name	metzoon.metz before:rh
width	5
height	-1.00
depth	-1.00
leftHalfway	
rightHalfway	
stroke	
approximate	
r1	
x	(200,60)
y	(7,00)
r2	
x	(524,2)
y	(2,00)
r3	
x	(204,30)
y	(11,00)
r4	
x	(4,80)
y	(48,11)

Undo Stack Properties 162,868 - 114,958



Examples of expandable glyphs

- Medial meem : $[-0.3,6]$ dots from the left, $[-0.3,6]$ dots from the right



- Expandable final Kaf



- Expandable final feh



OpenType Rules

OpenType rule = list of sequences to match

If a sequence of the input matches a sequence of the rule

→ Apply **substitution** or **positioning** to the matched input sequence

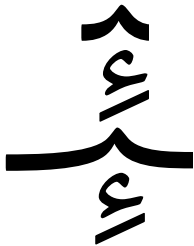
substitution : replace one or more glyph by one or more glyph : composition, decomposition, ligature, alternates

positioning : kerning, cursive attachment, mark over base, mark over mark

organized into **lookups** which are organized into **features**

OpenType positioning extension

- Positions are not fixed anymore : depend on parameter values applied during shaping
- Should be a function which calculates positions at runtime



OpenType feature file

Anchor function example

```
lookup hamzaabovebelow {  
  
    feature mkmk;  
  
    lookup hamzaabovebelow.11 {  
        pos base /^behshape.medi/ <anchor function defaultbaseanchorforlow>  
        markClass hamzaabove <anchor function defaullofmarkanchor> @hamzaabove ;  
    } hamzaabovebelow.11;  
  
    lookup hamzaabovebelow.12 {  
        pos mark hamzaabove <anchor function defaultbaseanchorforlow>  
        markClass [kasra kasratan kasratanidgham] <anchor function defaullofmarkanchor> @markclass3 ;  
    } hamzaabovebelow.12;  
  
    pos /^behshape.medi/ hamzaabove'lookup hamzaabovebelow.11 [kasra kasratan kasratanidgham]'lookup hamzaabovebelow.12;  
}  
hamzaabovebelow;
```

- Anchor functions which take as parameters class name, glyph name and applied expansion parameters

OpenType feature file

Other extensions example

```
pos [ fathatanidgham kasratanidgham dammatanidgham /^noon/]'lookup lgray
([/^alef.fina/ /^alef.isol/ /^yehshape/] | NULL)'
space'
(/^behshape.init/ twodotsdown | [ /^meem.init/ /^waw.isol/ ])' lookup green
[@marks]'lookup green
(@marks | NULL)'lookup green
;
```

- Regular expression : All initial behshape glyphs : /[^]behshape[.]init/
- | operator : 1 rule instead of 6 rules in the example above

Visual METAFONT - OpenType View

The screenshot displays the Visual METAFONT OpenType View interface. On the left, a list of glyphs is shown, including various ligatures and diacritics. The main area displays the Arabic text: **إِنَّ الَّذِينَ كَفَرُوا سَوَاءٌ عَلَيْهِمْ أُنذِرْتَهُمْ أَمْ لَمْ تُنذِرْهُمْ لَا يُؤْمِنُونَ ﴿٦﴾ خَتَمَ اللَّهُ عَلَى قُلُوبِهِمْ وَعَلَى سَمْعِهِمْ وَعَلَى أَبْصَارِهِمْ غِشْوَةً وَلَهُمْ عَذَابٌ عَظِيمٌ ﴿٧﴾** **وَمِنَ النَّاسِ مَنْ يَقُولُ آمَنَّا بِاللَّهِ وَيَأْتِيهِمُ الْآخِرُ وَمَا هُمْ بِمُؤْمِنِينَ ﴿٨﴾** **يُخَادِعُونَ اللَّهَ وَالَّذِينَ آمَنُوا وَمَا يُخَادِعُونَ إِلَّا أَنفُسَهُمْ وَمَا يَشْعُرُونَ ﴿٩﴾** **فِي قُلُوبِهِمْ مَرَضٌ فَزَادَهُمُ اللَّهُ مَرَضًا وَلَهُمْ عَذَابٌ أَلِيمٌ** **بِمَا كَانُوا يَكْفُرُونَ ﴿١٠﴾** **وَإِذَا قِيلَ لَهُمْ لَا تُفْسِدُوا فِي الْأَرْضِ قَالُوا إِنَّمَا نَحْنُ مُصْلِحُونَ ﴿١١﴾** **إِنَّمَا أَنفُسُهُمْ هُمُ الْمُفْسِدُونَ وَلَكِنْ لَا يَعْلَمُونَ ﴿١٢﴾** **وَإِذَا قِيلَ لَهُمْ عَابِدُوا كَمَا عَابَدْنَا قَالُوا أَنُؤْمِنُ كَمَا عَابَدْنَا قَالُوا لَا نَفْعَ لَنَا إِذَا نَحْنُ مُسْتَهْزَؤُونَ ﴿١٤﴾** **إِنَّهُ سَيَنفِرُ فِيكُمْ وَيُمَدُّهُمُ فِي طَعْفِهِمْ** **بِعَمَلِهِمْ ١٥** **أَوَّلِكَ الَّذِينَ اسْتَنزَوْا الصَّلَاةَ بِالْهَيْدَىٰ فَمَا رَحِمَتْ جُنُوبَهُمْ وَمَا كَانُوا مُهْتَدِينَ ﴿١٦﴾**

On the right, a preview window shows the text in a serif font, with the same Arabic text as the main area. Below the preview, there are labels for 'Related OpenType lookups' and 'Relief'.

Justification example

- Space justification

التَّيِّبُونَ الْعَبِيدُونَ الْحَمِيدُونَ السَّيِّحُونَ
الرَّاكِعُونَ السَّاجِدُونَ الْأَمْرُونَ بِالْمَعْرُوفِ

- Justification using glyph expansion.

التَّيِّبُونَ الْعَبِيدُونَ الْحَمِيدُونَ السَّيِّحُونَ
الرَّاكِعُونَ السَّاجِدُونَ الْأَمْرُونَ بِالْمَعْرُوفِ

All the final letters of each word are expanded by the same amount

Comparison with handwritten text

التَّيِّبُونَ الْعَبِيدُونَ الْحَمِيدُونَ السَّيِّحُونَ
الرَّاكِعُونَ السَّاجِدُونَ الْأَمْرُونَ بِالْمَعْرُوفِ

التَّيِّبُونَ الْعَبِيدُونَ الْحَمِيدُونَ السَّيِّحُونَ
الرَّاكِعُونَ السَّاجِدُونَ الْأَمْرُونَ بِالْمَعْرُوفِ

● ligature decomposition (first line, third word) : $الْحَا \rightarrow اَلْحَا$

● alternate glyph initial Kaf (second line, first word) : $ك \rightarrow ك$

PDF generation - Font

Generate Type3 Font on the fly

Advantage : Can use arbitrary PDF graphics operators.

→ no limit in the use of METAFONT (as draw with pen)

Drawback : no hinting mechanism for improving output at low resolutions

Should not be anymore an issue since nowadays most devices have high resolution.

Type 3 character (verse 6) with color using other graphic objects :



PDF generation - Searching and Copy&Paste

ToUnicode which supported by most PDF Reader cannot be used :

- Cannot encode many glyphs to one as in *beh* → *behshape + dot_below*
- position order ≠ logical order → reversed glyph especially with diacritic marks

Use ActualText construct :

- Group base glyph with following diacritics into the same ActualText
- In special case (glyph over another) group two base glyphs together otherwise they will be reversed
- Supported only in Acrobat Reader (PC version)

Other features

- overlapping glyphs detection
- adjust mark/cursive/kern positioning graphically

Future Works

Short and medium term

- Justification Algorithm
 - ▶ Add other justification techniques such as ligature decomposition
 - ▶ During expansion, stretch the two adjusting letters
 - ▶ Add the necessary shrinking rules
- Automatic mark adjustment algorithm to eliminate overlapping : especially during line shrinking when glyphs are very tight
- Clean the code and publish it as open source
- Implement this work in Lua \TeX

Other interesting works to consider in the long term

- Extend \TeX breaking paragraph to take into account glyph stretching and shrinking
- Extend METAFONT to allow to use a rotating pen with changing breadth

Thank you

شكرًا