A Visual TrueType Hinting Tool for RoboFont

Jérémie Hornus & Samuel Hornus - Robothon 2015
Contents

• Visual Interactive feedback
• Quadratic Converter
• From VGP to TT ASM
• Inside the UFO
• Inside the TTF

• Some tools to help the hinter’s work
DEMO time
QuadraticConverter

Cubic (PostScript)  →  Quadratic (TrueType)

Max Distance: 1.0
Min Segment Length: 30

Preview  Close  Convert Font

Layer (per-glyph): foregro...  Convert Glyph

WARNING. Un-saved modifications in a UFO will not be converted.
Interactive Feedback

User modifies Visual Glyph Program

grid-fitted outline & bitmap

Update TrueType Assembly

.PTTF

Generate Mini Font

FreeType/freetype-py
Visual Glyph Program

Visual Commands:
- Align
- Single Link
- Double Link
- Interpolate
- Middle Delta
- Final Delta
Visual Glyph Program

Visual Commands:
- Align
- Single Link
- Double Link
- Interpolate
- Middle Delta
- Final Delta
**Visual Glyph Program**

1. **A before B?**
   - B before A?
   - Doesn’t matter?

2. **Set of Rules**
   - Sorts commands by pairs according to TYPE and POINTS involved.

3. **Topological Sort**
   - Sorts commands by pairs according to TYPE and POINTS involved.

4. **Grouping**
   - Groups consecutive commands having similar types (align, single, interpolate, etc.).

5. **TrueType Assembly**
   - Process Batches
Deactivate stems for grayscale and subpixel

<table>
<thead>
<tr>
<th>Index</th>
<th>Active</th>
<th>Code</th>
<th>Lane</th>
<th>Lane1</th>
<th>Lane2</th>
<th>Align</th>
<th>Round</th>
<th>Stem</th>
<th>Zone</th>
<th>Delta</th>
<th>PPM1</th>
<th>PPM2</th>
<th>Mono</th>
<th>Gray</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Program

<table>
<thead>
<tr>
<th>Index</th>
<th>Code1</th>
<th>Code2</th>
<th>Lane1</th>
<th>Lane2</th>
<th>Align</th>
<th>Round</th>
<th>Stem</th>
<th>Zone</th>
<th>Delta</th>
<th>PPM1</th>
<th>PPM2</th>
<th>Mono</th>
<th>Gray</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

BATCH
<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root</td>
<td>Dictionary</td>
<td>(14 items)</td>
</tr>
<tr>
<td>com.fontlab.v2.tth</td>
<td>Dictionary</td>
<td>(5 items)</td>
</tr>
<tr>
<td></td>
<td>alignppm</td>
<td>Number</td>
</tr>
<tr>
<td></td>
<td>codeppm</td>
<td>Number</td>
</tr>
<tr>
<td></td>
<td>stems</td>
<td>Dictionary</td>
</tr>
<tr>
<td></td>
<td>stemsnap</td>
<td>Number</td>
</tr>
<tr>
<td></td>
<td>zones</td>
<td>Dictionary</td>
</tr>
<tr>
<td></td>
<td>com.robofont.robohint.cvt</td>
<td>Array</td>
</tr>
<tr>
<td></td>
<td>com.robofont.robohint.fpgm</td>
<td>Array</td>
</tr>
<tr>
<td></td>
<td>com.robofont.robohint.gasp</td>
<td>Dictionary</td>
</tr>
<tr>
<td></td>
<td>com.robofont.robohint.maxp.maxFun...</td>
<td>Number</td>
</tr>
<tr>
<td></td>
<td>com.robofont.robohint.maxp.maxStorage</td>
<td>Number</td>
</tr>
<tr>
<td></td>
<td>com.robofont.robohint.prep</td>
<td>Array</td>
</tr>
<tr>
<td>com.sansplomb.tth</td>
<td>Dictionary</td>
<td>(2 items)</td>
</tr>
<tr>
<td></td>
<td>bitmapPreviewSelection</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td>deactivateStemWhenGrayScale</td>
<td>Number</td>
</tr>
<tr>
<td></td>
<td>com.typemytype.typefacefont.Cubic cont...</td>
<td>Array</td>
</tr>
<tr>
<td></td>
<td>com.typemytype.typefacefont.foreground...</td>
<td>Array</td>
</tr>
<tr>
<td></td>
<td>com.typemytype.typefacefont.layerOrder</td>
<td>Array</td>
</tr>
<tr>
<td></td>
<td>com.typemytype.typefacefont.segmentType</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td>com.typemytype.typefacefont.sort</td>
<td>Array</td>
</tr>
<tr>
<td></td>
<td>public.glyphOrder</td>
<td>Array</td>
</tr>
</tbody>
</table>
com.fontlab.ttprogram
Visual Glyph Program
encoded in base 64

com.robofont.robohint.assembly
Plain TrueType Assembly
.TTF

FontProgram
<FPGM>
functions

ControlValueTable
.CVAT>

functions size-dependent rasterizer control

GlyphAssembly
<GLYF>

set stems and zones according to the pixel size...

PreProgram
<PREP>

or

ControlValueProgram
<prep>

stems and zones definitions
Helper Tools
Helper Tools

Auto-stem
Finds typical stem widths
**Helper Tools**

**Auto-stem**
Finds typical stem widths

**Auto-hint**
Generates a Visual Glyph Program
Helper Tools

**Auto-stem**
Finds typical stem widths

**Auto-hint**
Generates a Visual Glyph Program

Stems and VGP provide a starting point for the font hinting job.
Helper Tools

**Auto-stem**
Finds typical stem widths

**Auto-hint**
Generates a Visual Glyph Program

**Auto-match**
Transfers VGP to another font [in progress]

Stems and VGP provide a starting point for the font hinting job.
Defining a ‘stem’

‘stem’ = ‘special pair of control points’
Defining a ‘stem’

Configuration of two points A and B:
- Position
- In- and Out- Tangents
- Clockwise
**Defining a ‘stem’**

The two points A and B form a stem when:
Defining a ‘stem’

The two points A and B form a stem when:

- A and B have two parallel tangents
- A lies on the inked side of B’s tangent and vice-versa
- Segment [AB] is fully inked
Finding ‘stems’

- A and B have two parallel tangents
- A lies on the inked side of B’s tangent and vice-versa
- Segment \([AB]\) is fully inked

Find all stems and sort them into horizontal stems and vertical stems.
Finding ‘stems’

Is AB a stem?
horizontal: yes
vertical: no
Auto-stem

[filling the CVT with relevant stem widths]
Auto-stem

1. Gather stem widths into an array (of numbers)
Auto-stem

1. Gather stem widths into an array (of numbers)
2. Cluster them into groups (k-means)
Auto-stem

1. Gather stem widths into an array (of numbers)

2. Cluster them into groups (k-means)

3. Take the mean of each group as a relevant stem width
Auto-stem

1. Gather stem widths into an array (of numbers).
2. Cluster them into groups ($k$-means).
3. Take the mean of each group as a relevant stem width.

Control Value Table

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>20</th>
<th>40</th>
<th>60</th>
<th>80</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>45</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>73</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>102</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

X Stems

<table>
<thead>
<tr>
<th>Name</th>
<th>Width</th>
<th>1 px</th>
<th>2 px</th>
<th>3 px</th>
<th>4 px</th>
<th>5 px</th>
<th>6 px</th>
</tr>
</thead>
<tbody>
<tr>
<td>X_91</td>
<td>91</td>
<td>0</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td>60</td>
</tr>
<tr>
<td>X_50</td>
<td>50</td>
<td>0</td>
<td>33</td>
<td>50</td>
<td>66</td>
<td>83</td>
<td>100</td>
</tr>
<tr>
<td>X_74</td>
<td>74</td>
<td>0</td>
<td>25</td>
<td>37</td>
<td>50</td>
<td>62</td>
<td>75</td>
</tr>
<tr>
<td>X_152</td>
<td>152</td>
<td>0</td>
<td>12</td>
<td>18</td>
<td>25</td>
<td>31</td>
<td>37</td>
</tr>
<tr>
<td>X_125</td>
<td>125</td>
<td>0</td>
<td>16</td>
<td>25</td>
<td>33</td>
<td>41</td>
<td>50</td>
</tr>
</tbody>
</table>

Detect

Apply | OK
Auto-hinting

[Automatically generates a VGP]

Two steps:
1. Analyse the glyph by structuring its set of points
2. Use this structure to generate the VGP
Alignments

horizontal alignments for vertical hinting
Alignments

vertical alignments for horizontal hinting
Groups
Groups

3 groups of alignments
Hinting the groups (in $X$)
Hinting the groups (in X)
Hinting alignments in a group (in X)

leader alignment; received the interpolate command.
Hinting alignments in a group (in X)
Hinting alignments in a group (in X)

Finally, the points in an alignment are single-link’d.
VGP transfer via contour matching
VGP transfer via contour matching
VGP transfer via contour matching
VGP transfer via contour matching

1. Measuring dissimilarity of 2 points
2. Matching 2 pointed contours (with a startpoint)
3. Matching 2 contours
4. Matching 2 glyphs
We choose $\mathcal{D}(A, B)$ so that it is small when $A$ and $B$ are close and have similar tangents.

$$
\mathcal{D}(A, B) = \left( f(t_A^i, t_B^i) + f(t_A^o, t_B^o) \right) \times ||A - B||^2
$$

where $f(t_A, t_B) = \frac{2}{1 + t_A \cdot t_B}$

For example $\mathcal{D}(A, B) < \mathcal{D}(A, C)$
Matching two pointed contours

What is a matching?
Matching two pointed contours

- Maps each point of source contour to a point of target contour
Matching two pointed contours

- Maps each point of source contour to a point of target contour
- Maps source startpoint to target startpoint
- Crossing arrows are forbidden
Matching two pointed contours

Score of this pointed matching is

$$\sum \mathcal{D}(\bigcirc, \bigcirc)$$

arrows
Matching two pointed contours

Find the **best pointed matching** (lowest score) with dynamic programming
Matching two contours

- Fixed startpoint
- Try all points as startpoint
Matching two contours

Fixed startpoint

Try all points as startpoint

Find best matching by trying all pointed matchings and picking the best one
Matching two glyphs

source glyph

target glyph
Matching two glyphs

source glyph
Matching two glyphs

source glyph

target glyph

source glyph
VGP Transfer