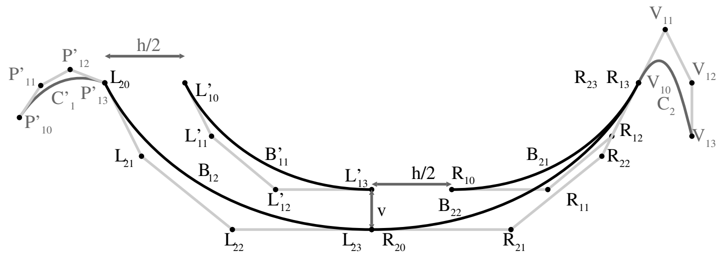


Figure 45. Translation of C_1 towards the right direction in h

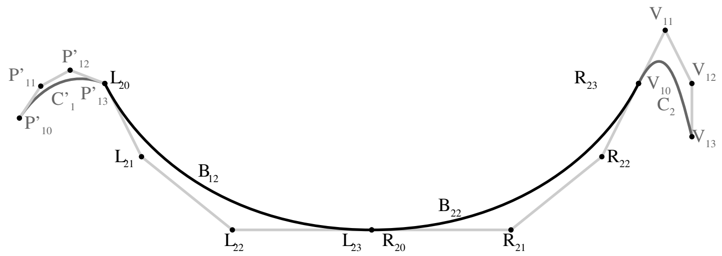


Figure 46. The set C after stretching of h and v

is a stretchable one, the stretching occurs in this letter otherwise the stretching will be in the ligature between the two last connected letters of the word. The two cases are presented separately in the following.

4.2.1 Inter-letters keshideh

The keshideh is a juxtaposition of two curves of type 1 and type 2. When the keshideh appears between two connected letters, the first curve is the horizontal preceding connection below the baseline of the letter in the left and the second part of the keshideh is the horizontal succeeding connection below the baseline of the letter in the right. When we designed the font, we parameterized the horizontal connections exactly on the baseline in a way such that they can be stretched horizontally to the right and vertically downwards. So, when the stretching is null, the connections are simply those exactly on the baseline, otherwise they are horizontal strictly below the baseline. From now, we mean by parameterized horizontal connection exactly on the baseline a connection that support the connection exactly on the baseline and below the baseline. For example, when the word ظفر is to be stretched, the stretching occurs between the letters ظـ and ـر. The letters ظـ and ـر have horizontal connections exactly on the baseline that are parameterized to support stretchability. Then, the Stretching consists on keeping the word as it is and communicating a horizontal and vertical amounts of stretching as parameters to the procedures for writing these two letters. Suppose that we want to stretch the word ظفر of an amount h horizontally and v vertically downwards, then, we have to perform the two actions :

- to stretch the horizontal succeeding connection of the letter ظـ of $\frac{h}{2}$ horizontally and v vertically and
- to stretch the horizontal preceding connection of the letter ـر of $\frac{h}{2}$ horizontally and v vertically.

In the figure 47, an illustration of a stretching in a font size equal to 64 points is given. In the first line, the word ظفر with a null stretching is presented. In the last line, the same word is stretched 12 dp (**d**iacritic **p**oint) horizontally and 0.5 dp vertically. In the second line the letters ظـ and ـر in their initial shapes are showed whereas in the third line, both of the two letters have a stretching of 6 dp horizontally and 0.5 dp vertically.

Most of the handbooks of calligraphy [3, 22, 25], teach that : in the beginning of a word in Arabic letters, the commonly used horizontal connections are oblique, big strictly above the baseline and strictly above the baseline. That's an aesthetic need. When we have to write the word formed of the letters BEH in initial position and AIN in the end without stretching, a letter BEH and AIN with horizontal succeeding connection and preceding connection respectively strictly above the baseline are used. Let بع be such a word (the word to the right on the figure 48), when a stretching is needed, the alternatives of these two letters with horizontal connections below the baseline (they are horizontal connections exactly on the baseline when the amount stretching is null) are used. In this way, the word بع would be used. The two alternatives of the word are in clear on the figure 48.

Figure 47. Inter-letters keshideh mechanisms

We can introduce these alternatives in the font to solve the problem of stretchability needs so that the texts processed with the font satisfies the Calligraphic rules. Of course, these alternatives are allowed by the Calligraphic rules and exist. Consider the word بط, the letter ب admits an horizontal connection after *exactly on* the baseline because ط is a strongly rising letter from the baseline. It is the same for the letter ع that admits a preceding connection exactly on the baseline. It is mandatory in a situation such as after the letter ط in the word اطع or after ف inside the word نفع. We can apply the alternatives of letters also when the ligature is oblique, in the case of a null stretching. But it doesn't work in some particular situations such that concerning the stretching of the word خط. Stretching the ligature between the two letters of the word خط has been handled in a particular way. Indeed, when some characters such as خ (or its variants) are in initial position and that they are followed by TAH in final position ط (or its variants), the connections are *oblique* and then, stretching through using the alternative of the letter KHAH and TAH with parameterized horizontal connections exactly on the base line gives a bad result in comparison with the handwritten proof. It took a lot of time to find a solution to this case. It reminds the case where D.E. Knuth spent a lot of time modeling mathematically the letter S in Metafont. The cases are not analogous, technically saying, but in difficulties to find a solution they are. A method to find a solution to this case consists on including the stretching inside

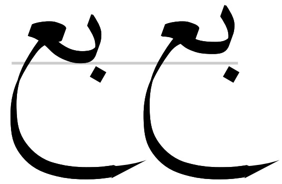


Figure 48. Useful alternatives to stretch

the character TAH. We developed an alternative of the TAH in final position with a connection before consisting of two Bézier curves B_l in $\mathcal{B}_1$ with control points L_0, L_1, L_2 and L_3 on the left, followed by another Bézier curve B_r in $\mathcal{B}_2$ with control points R_0, R_1, R_2 and R_3 on the right such L_3 coincides with R_0. Let ط be that alternative (see figure 50), the keshideh will then be the combination of these curves and there is precisely the place where stretching will take place (see figure 51, the first occurrence of the word in the right is with oblique connections). We conclude that to stretch the word خط, we may keep the letter خ without changes and replace the letter ط (see figure 49) by ط (see figure 50) and then, the stretching can be performed. With this alternative the stretching of the word خط has been appreciated by a calligrapher and became good in comparison to the handwritten proof.

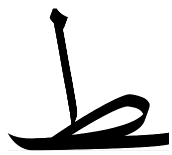


Figure 49. TAH in final position with *oblique* preceding connection

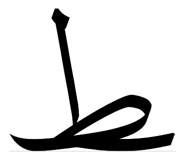


Figure 50. Alternative for stretching the character TAH in final position proceeded by HAH in initial position

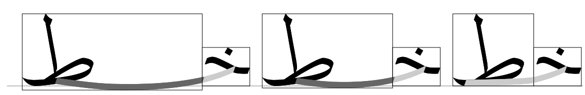


Figure 51. Stretching in the level of TAH

4.2.2 The intra-letter keshideh

When the keshideh is inside a character, say for instance, into the letter QAF, in final position of the word حق, according to the Arabic calligraphy rules, the letter QAF admits an horizontal stretching between 9 and 12 diacritic points. The shape of the letter QAF ق without stretching is quite different from the stretched one, the character ق. ق is only a variant of ق. So, it has the same width of ق increased of 9 diacritic points. It also has the same depth as ق increased of the vertical downward stretching corresponding to the horizontal stretching of 9 diacritic points. To compute the stretching, the letter ق is used and can be stretched horizontally between zero and three diacritic points. The vertical stretching varies in the range corresponding to the horizontal extension from zero to three diacritic points. The keshideh is performed inside ق. The part modeling the keshideh is displayed in gray on the figure 52. The letter QAF in final position is in font size 128 points. The control points of the left bottom corner of the nib's head are given to make it more understandable.

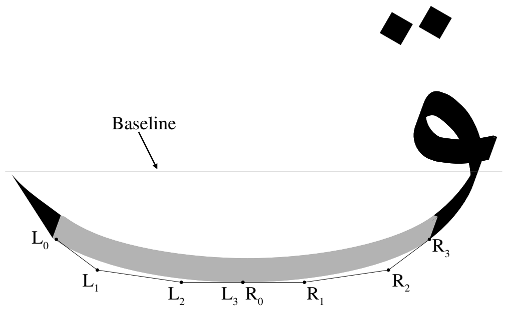


Figure 52. The keshideh in the letter QAF in final position

In the development of our PostScript font, the horizontal stretching is a value communicated to character procedures as a global variable h. As in [8], at the end of each call of procedure, the value of h is put back to zero. The value h is determined by means external to the font, such as the text formatting programs. Then it is communicated to the character procedures in the font. The vertical stretching v is determined from the horizontal stretching in the font through the simple function φ defined in the following :

- h : value of horizontal stretching (towards the right direction),
- v : value of vertical stretching corresponding to h (downward), v depends on h, so we can write $v = \varphi(h)$ where φ is defined by :

$$\begin{array}{ccccc} \varphi & : & [0, h_m] & \rightarrow & [0, v_m] \\ & & h & \mapsto & \frac{v_m}{h_m} h \end{array}$$

- h_m : maximal value of horizontal stretching,
- v_m : maximal value of vertical stretching.

With regard to the definition of the function φ, the curvilinear stretching functions E_{af} and E_{be} defined above can be regarded as linear scaling, but they are not so.

4.3 The nib's head motion modeling approach versus other approaches

In this sub-paragraph, the approach for modeling of the nib's motion is compared in one hand to the Knuth's approach in Metafont [10] and to the Kinch's one in MetaFog [28] in the other hand. These two approaches seek the envelope of an ellipse stroking (a nib with an elliptic head) in two different ways. The Knuth's approach follows the Hobby method [16] to represent the envelope in terms of the raster instead of scalable curves. Kinch solve the problem through representing the envelope in an algebraic and topological way. In our approach, the envelope of the characters static parts is determined using tools outside the PostScript interpreter. A specific program (based on a mathematical idea presented in the sub-paragraph 2.4) helps to determine this envelope. Of course, we also can use Metapost [17], to determine the envelope and cope with other programs to eliminate the fact that some zones are painted twice. The characters static parts have *true* outlines (there are no overlapping curves), the envelope is generated as with MetaFog. Concerning the characters dynamic parts, such as the parts of the keshideh, the envelope is determined in a way different from the Knuth's and Kinch's approaches. In order to well explain and justify our approach, let us consider a left component of a keshideh (see figure 53).

The curve B_1 of type 1 in the figure 53 has the control points (169,205), (170,151), (249,134) and (340,134). The nib's head width and thickness are 31.9643 and 15.9821 points respectively. Here, we have considered a thickness equal to half of the width for clarity. The points where the slops are parallel to the vectors defining the nib when the thickness is a sixth of the width are the same. The vectors $\overrightarrow{u}_1$, $\overrightarrow{u}_2$ and $\overrightarrow{u}_3$ are respectively (10.9177,30.042), (-4.10678,35.4913) and (-15.0244,5.44933). $B_1'(t)$ is parallel to $\overrightarrow{u}_2$ at 0.0323 and to $\overrightarrow{u}_3$ at 0.4923. After applying a stretching with the E_{be} to B_1 with an horizontal stretching equal to 109.1601 points and a corresponding vertical one 9.0966 points, we obtain the curve B_2 with the control points (59.8398972,205), (60.9680176,144.081482), (190.549225,124.903397) and (340,124.903397). Then $B_2'(t)$ is parallel to $\overrightarrow{u}_2$ and $\overrightarrow{u}_3$ at 0.0224 and 0.3980 respectively. The coefficients where the slopes are parallel to the vectors of the nib's head change from a stretching state to another. The reason for this is simply that the functions of stretching used to compute the keshideh are not a *linear scaling*. In order to determine these coefficients and therefore, to be able to determine the envelope as the Kinch's approach seek through computing these coefficients in the PostScript character procedure. Then, printing the character will be very slow. To avoid such problem, we opt for painting the surface razed with ev-

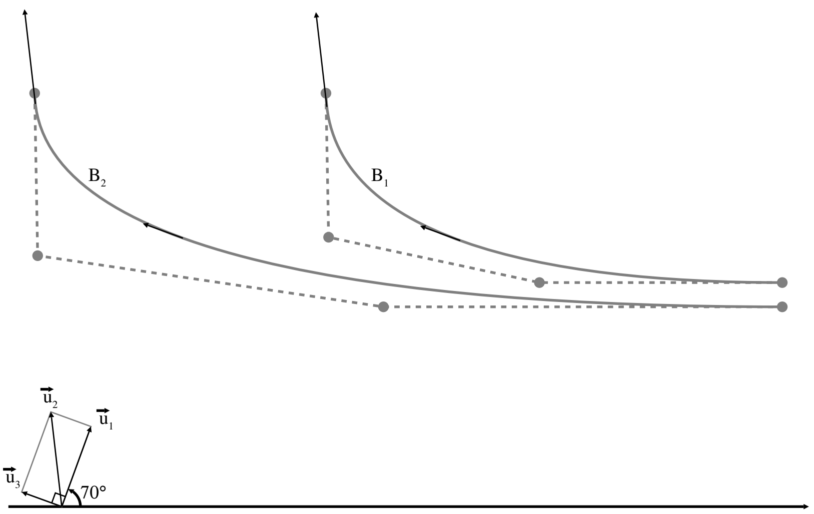


Figure 53. Identification of points where slops are parallel to vectors defining the nib's head

ery edge of the nib's head. So, some parts are painted twice or more causing a larger CPU time consuming but it is still less than the time used to compute the outlines of the curve. As we have seen before, some parts of the surface razed with the nib's head can be lost in the neighborhood of the points where the slops are parallel to the vectors $\overrightarrow{u}_2$ and $\overrightarrow{u}_3$. In the figure 54, the real surface razed with the nib's head according to the curve B_1 is showed. Whereas, in the figure 55, we have drawn the surface that is on the figure 54 and then drawn the envelope (in gray) with our adopted technique on it. We remark that we can not distinguish the lost zones (that must be in black). The reason is that the keshideh curves have curvature vectors [6] with small magnitudes (Or the radiuses of the osculating circles are big) especially on the points where the derivative is collinear to $\overrightarrow{u}_2$ and $\overrightarrow{u}_3$. We can accept then this inaccuracy because the results are considered more in their visual aspects. Reeves [32] call this phenomenon "Visual Accuracy". When we consider a nib's head that respect the ratio existing between the width and the thickness of the nib's head (one a sixth) the result would be more satisfying visually.

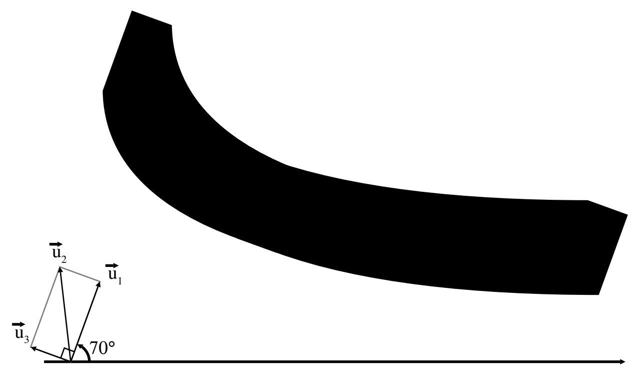


Figure 54. The real razed surface

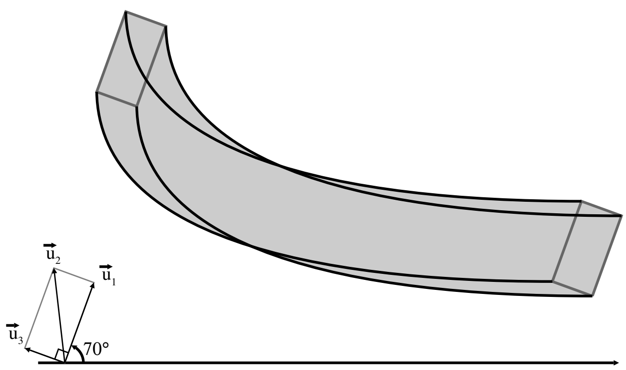
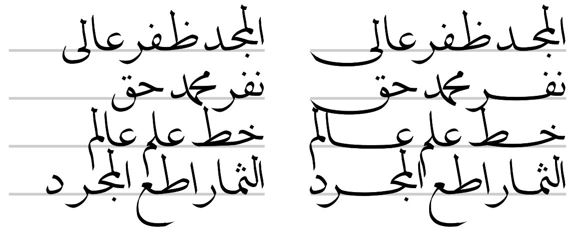


Figure 55. The razed surface with our approach



(a) (b)

Figure 56. a) Arabic text lines before justification, in 18 pt, b) Arabic text lines after justification.

5. Conclusion and perspectives

Our main goal consisted on designing a font that helps in producing Arabic documents looking like handwritten proof. So, it was necessary to identify all kinds of ligatures and to introduce keshideh in an adequate way. We were brought to scan the handwritten proofs and then to work on these proofs with simple graphic tools like Kontour [19] under Linux [24], in order to get the encoding of the letters in terms of Postscript Bézier's curves. The existing font tools were not enough to take into account the motion of the nib's head, the metrics in diacritic point, the possibility to correct some errors allowed for the calligrapher in handwritten proofs etc. We worked almost manually to design the PostScript encoding of some characters that represent most of the cases and we got so a *mini-font*. This font has been used to give the example in the figure 56. This is a set of lines (with no meaning in Arabic) to assess justification through keshideh. The requirements, support and development of a tool to assist in designing Arabic fonts should be a separate work. In any way, there will be no satisfying compromise to respect the rules of Arabic calligraphy unless a geometrical model for all the characters and symbols is built.

References

[1] Adobe Systems Incorporated, *Adobe Type 1 Font Format*, Part No. LPS0064, 1990.

[2] Adobe Systems Incorporated, *PostScript Language Reference Manual*, MA:Addison-Wesley , Third edition,Massachusetts, 1999.

[3] Ahmed El Houssaini, *The Arabic Calligraphy*, Dar Attaila, 1994.

[4] Azzeddine Lazrek, "A package for typesetting Arabic mathematical formulas", *Die TEXnische Komodie*, DANTE e.V., Vol.13, No. 2, pp. 54–66, 2001.

[5] Azzeddine Lazrek, "CurExt, Typesetting variable-sized curved symbols", *EuroTEX 2003 Pre-prints : 14th. European TEX conference,* Brest, France, pp. 47-71, 2003.

[6] Brian A. Barsky, *The Beta-splines : A Local Representation Based on Shape Parameters and Fundamental Geometric Measures*, Department of Computer science, Ph.D. dissertation, University of Utah Salt Lake City, December 1981.

[7] Brian A. Barsky, *Arbitrary Subdivision of Bézier Curves*, Technical Report UCB.CSD 85/265, Computer Science Division, University of California, 1985.

[8] Daniel M. Berry, "Stretching Letter and Slanted-Baseline Formatting for Arabic, Hebrew and Persian with dittroff/ffortid and Dynamic PostScript Fonts", *Software-Practice and Experience*, vol. 29, no. 15, pp.1417-1457, 1999.

[9] D.E. Knuth, *The TEXBook, Computers and Typesetting*, Reading MA : Addison-Wesley, Vol.A, 1984.

[10] D.E. Knuth, *The MetafontBook, Computers and Typesetting,* MA: Addison-Wesley, Vol.C, 1986.

[11] D. Weise, D. Adler, *TrueType and Microsoft Windows 3.1*, Technical Report, Microsoft Corporation, Redmond, WA, 1992.

[12] Jacques André, B. Borghi, "Dynamic Fonts", *PostScript Language Journal*, Vol 2, no 3, pp. 4-6, 1990.

[13] Jacques André, Irène Vatton, *Contextual Typesetting of Mathematical Symbols Taking Care of Optical Scaling*, Technical report No 1972, INRIA, October 1993.

[14] Jacques André, Irène Vatton, "Dynamic Optical Scaling and Variable-sized Characters", *Electronic Publishing,* Vol 7, No 4, pp. 231-250, December 1994.

[15] Jeffrey M. Lane, Richard F. Riesenfelf, "Bounds on Polynomial", *BIT*, Vol 21, No 1, pp. 112-117, 1981.

[16] John Douglas Hobby , *Digitized Brush Trajectories*, Ph.D. Thesis, Department of Computer sciences, Stanford University, August 1985.

[17] John Douglas Hobby, *A User's manual for MetaPost*, Technical Reports Report, AT& T Bell Laboratories Computing Science 162, 1992.

[18] Klaus Lagally, *ArabTEX User Manual,* Universitat Stuttgart, Fakultat Informatik, Version 4.00, 2004.

[19] KOffice. '*Kontour : Koffice drawing tool*, *http://www.koffice.org/Kontour*, 2001.

[20] Leslie Lamport, *LATEX-A Document Preparation System*, Reading MA: Addison Wesley, 1985.

[21] LYX Team, *The LYX User Manual*, *http://www.LYX.org*, Version 1.3.4, 2004.

[22] Mahdi Essaid Mahmoud, *Learning Arabic Calligraphy : Naskh, Roqaa, Farsi, Thuluth, Diwany*', Ibn Sina publisher, Cairo- Egypt, 1994.

[23] Mamoru Hosaka, Fumihiko Kimura, "A Theory and Methods for Free Form Shape Construction", *Journal of Information Processing,* Vol 3, No 3, pp. 140-151, (1980).

[24] Mandrake Soft, *Mandrake Linux 8.2*, *http://www.mandrakeSoft.com*, 2002.

[25] Mohammed Hachem El Khattat , *Arabic Calligraphy Rules, A Calligraphic set of Arabic Calligraphy Styles*, Books Univers, Beyrouth- Lebanon, 1986.

[26] M. J.E. Benatia, M. Elyaakoubi, A. Lazrek , "Arabic Text Justification", *TUG 2006 Conference Proceedings*, Volume 27, No 2, pp. 137-146, 2006.

[27] Mostafa Banouni, Mohamed Elyaakoubi, Azzeddine Lazrek, "Dynamic Arabic Mathematical Fonts", *Lecture Notes in Computer Science (LNCS)*, Springer Verlag, Volume 3130, pp. 158-168, 2004.

[28] Richard J. Kinch, "MetaFog : Converting Metafont Shapes to Contours", *Preprint: Proceedings of the Annual Meeting*, pp:1001-1011, June 1996.

[29] Ronald N. Goldman, "Using Degenerate Bézier Triangles and Tetrahedra to Subdivide Bézier Curves", *Computer-Aided Design*, Vol 14, No 6, pp. 307-311, November 1982.

[30] Yannis Haralambous, "The Traditional Arabic Type-case Extended to the Unicode Set Of Glyphs", *Electronic Publishing*, Vol. 8, No. 2-3, pp. 111-123, 1995.

[31] Yannis Haralambous, John Plaice, "Multilingual Typesetting with Ω, a Case Study : Arabic", *International Symposium on Multilingual Information Processing'97* , Tsukuba (Japon), pp. 137-154, 1997.

[32] William T. Reeves, *Quantitative Representations of Complex Dynamic Shapes for Motion Analysis*, PhD thesis, University of Toronto, 1981.

Abdelouahad BAYAR is professor of mathematics and computer science at Cadi Ayyad University (école Supérieure de Technologie) in Safi, Morocco since 1996. He gets a CEUS (Certificat d'Etudes Uinversitaire Supérieures) in applied mathematics and computer science in 1996. He integrates then the university as a research-teacher. He holds his Doctorat of the thirth cycle in 1999 in Computer science in communication networks. Since 2007, Bayar is a member of the group of Information Systems and Communication Networks LSNC in Information Systems Ingeniering Laboratory ISIL. His actual investigations deal especially with dynamic fonts and calligraphic rules formalization.

Khalid SAMI is professor of mathematics at Cadi Ayyad university (Faculté des Sciences) in Marakech, Morocco since 1983. Since years, he's intrested in multilingual scientific text formatting. Some of the work relevant to this area is presented in the url www.ucam.ac.ma/fssm/rydarab.