Caractères numériques : introduction

Jacques ANDRÉ

Irisa/Inria-Rennes
Campus de Beaulieu
F-35042 Rennes cedex
jandre@irisa.fr

Résumé. Après avoir montré comment on est passé des caractères en plomb à ceux informatisés, on introduit les principaux concepts de plan de bits, de contour, de machinerie, de *hint* et on montre comment la notion de caractère paramétré (dont *Multiple Masters* est un exemple) permet de retrouver la qualité du plomb.

Abstract. *This is part of a tutorial on fonts. First, the way from hot metal types to digitalized characters is shown. Then, the main concepts used in digital typography (bitmaps, outlines, hints, Multiple Masters, etc.) are explained.*

Avertissement – Cette note[1] présente les principaux concepts des caractères numériques[2] et correspond à un exposé fait lors des journées organisées par l'Association GUTenberg en mai 1997 à Strasbourg. Cet exposé étant suivi d'autres sur le thème plus général des fontes, de nombreux points ne seront pas abordés ni développés ici. On trouvera dans ce *Cahier* deux notes relatives à ces autres exposés [22, 58].

Il existe peu de synthèses sur les notions de caractères numériques, sur la façon dont ils sont créés et utilisés dans les imprimantes à laser ou photocomposeuses d'aujourd'hui. Citons toutefois les ouvrages de Rubinstein [59], Karow [40, 41] et du projet Didot [10, 27, 38].

1. Du plomb aux pixels

L'HISTOIRE de l'imprimerie, du livre et de l'édition a fait l'objet de très nombreux travaux[3], mais il y a peu d'études consacrées spécifiquement à

1. Il s'agit d'une version partielle mais mise à jour de [4].

2. L'expression de « caractères numérisés » est sans doute plus fréquente, mais nous lui trouvons une connotation de « qui a été numérisé » alors qu'un caractère numérique peut exister sans être la copie d'un caractère pré-existant !

3. Parmi les ouvrages français récents, citons BLANCHARD [20], CHARTIER et MARTIN [23], DREYFUS [28, pages 184–214] et FEBVRE et MARTIN [31].

l'histoire de la création de caractères notamment pas après ceux en plomb. René PONOT [56] et Walter TRACY [65], par exemple, ne font pas allusion à la numérisation des caractères. Toutefois, on trouvera chez Alan MARSHALL [48], Lynn RUGGLE [60] et Richard SOUTHALL [63] les éléments essentiels.

Les principales dates de l'histoire de la production de caractères sont finalement peu nombreuses.

1.1. 1435 : caractères mobiles en plomb

LA grande invention de GUTENBERG[4], vers 1435, n'est pas, comme on le croit généralement, celle de l'imprimerie (l'impression à partir de bois en relief est attestée en Chine dès la fin du premier millénaire), ni celle de la presse (en usage alors chez les vignerons), ni celle des caractères mobiles (des caractères en bois ont été utilisés dans des almanachs dès le XIV^{e} siècle). Sans nier ses compétences d'orfèvre (qu'il adapta aux techniques de gravure des poinçons), de chimiste et de métallurgiste (mise au point de l'alliage plomb–étain et plus tard antimoine, techniques de frappe, etc.), son invention est essentiellement celle du processus de fabrication, de production et de réutilisation de caractères.

Cette chaîne de production comprend les étapes suivantes[5] (voir aussi figure 1) :

1. Un poinçon est gravé à la main, à l'aide de gouges (figure 2). C'est une petite pièce d'acier d'environ 5 ou 6 cm de haut et de quelques millimètres carrés de surface. Ce poinçon représente exactement le caractère tel qu'il sera imprimé (figure 1.1).

2. Ce poinçon, en relief, est alors frappé sur une matrice (figure 1.2) où il laisse son empreinte (figure 1.3).

3. Cette matrice est ensuite introduite dans un moule (figure 2) où le caractère va être fondu (figure 1.4). Ce moule est probablement l'outil le plus génial de la chaîne : par un système de tirettes, on va pouvoir donner au caractère une chasse dépendant de la lettre (un « m » est plus large qu'un « i ») tout en gardant corps et hauteur en papier constants (voir les

4. La littérature sur GUTENBERG est impressionnante ! On trouvera dans le livre de BECHTEL [13] une synthèse sur l'histoire de GUTENBERG et sur sa technique de moulage des caractères, mais aussi sur l'histoire de son histoire.

5. La meilleure desription de ce processus reste encore le traité de typographie de … 1763é[32] ! On trouvera des « films » de démonstration de cette technique « fossile » d'une part dans les actes du colloque *The computer and the hand in type design* [29] et, d'autre part, sous forme d'une vidéo réalisée, dans le cadre du projet Didot, à Reading [66].

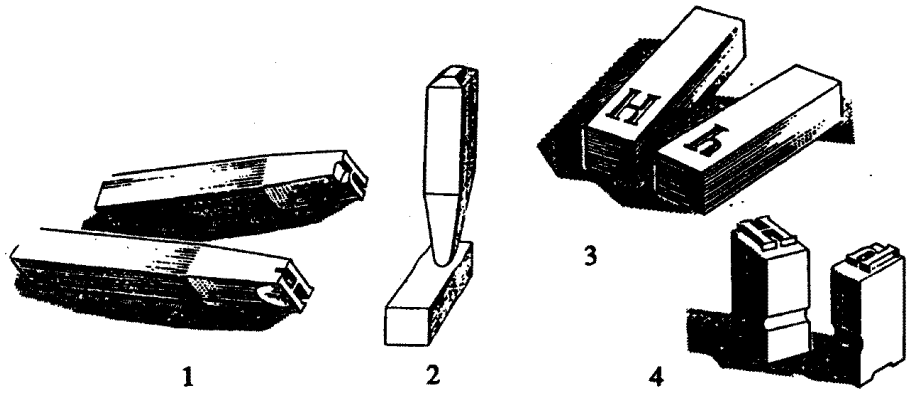


FIGURE 1 – *Principales étapes de la fabrication de caractères en plomb (d'après* DREYFUS *[28, page187] et* MARSHALL *[48, page 44])*

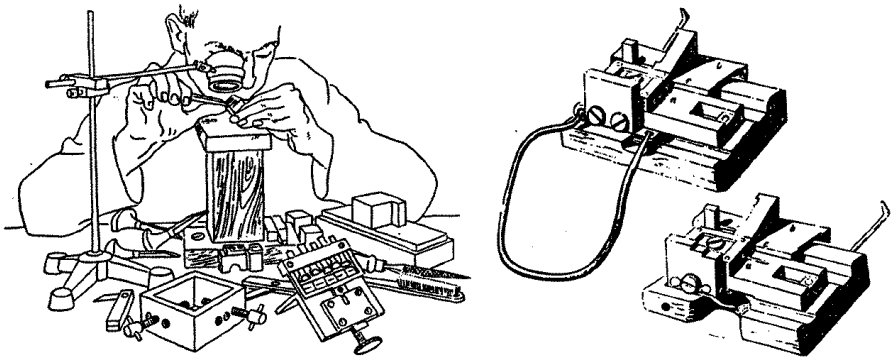


FIGURE 2 – *À gauche : taille d'un poinçon (d'après* TRACY *[65]) ; à droite : détail d'un moule (d'après Bechtel [13, page 328])*

détails techniques dans la description moderne de Stan NELSON [55]). Une fois les réglages faits pour un caractère (ou type), on peut en couler de nombreux identiques.

4. Les caractères sont alors utilisés pour composer un texte.

5. Une fois ce texte imprimé, les caractères sont démontés. Ils peuvent donc être réutilisés pour un autre texte.

6. Après avoir servi plusieurs fois, les caractères finissent par s'émousser. On les refond et le plomb ainsi récupéré permet de mouler de nouveaux caractères.

7. Lorsque l'on remoule des caractères, on part de la matrice. Si vraiment celle-ci est en mauvais état, on peut toujours repartir du poinçon, mais ce cas semble avoir été très rare.

C'est cette technique, à peine modifiée (les principales améliorations ont porté sur la composition de l'alliage et sur l'emploi de masselottes pour le démoulage), qui subsiste dans les rares ateliers où l'on grave encore des caractères (l'Imprimerie nationale à Paris a une équipe de trois personnes utilisant cette technique pour réparer les collections de poinçons conservés dans son *cabinet*).

1.2. 1885 : mécanisation de la taille des poinçons

DÈS le premier quart du XIX^e siècle, de nombreuses recherches ont lieu pour fabriquer des machines à composer qui débouchent sur la *Linotype* : l'idée est de couler non plus les caractères un par un puis de les composer manuellement, mais de composer, à l'aide d'un clavier comme celui d'une de ces nouvelles machines à écrire (la première Remington date de 1873), toute une ligne avec des matrices et de couler celle-ci d'un coup.

L'intérêt de mécaniser la fabrique des matrices amena un Américain, Lynn Boyd BENTON, à déposer en 1885 (c'est aussi l'année du brevet de la Linotype par MERGENTHALER) un brevet pour graver des poinçons. Leur taille est un travail délicat et fastidieux : pour un alphabet qui n'aurait que 80 signes, il faut non pas 80 poinçons mais $80 \times n$ poinçons où n est le nombre de corps désirés. Il faut en effet un poinçon pour le « a » en corps 6, un en corps 8, un en corps 9, etc. L'invention de BENTON consiste en l'emploi d'un pantographe (voir figure 3) qui permet, à partir d'un dessin original, de produire tous les poinçons voulus[6].

6. C'est cette invention, et non la photocomposition ou la numérisation des caractères, qui devrait faire hurler les défenseurs de la qualité typographique (voir section 3.1) !

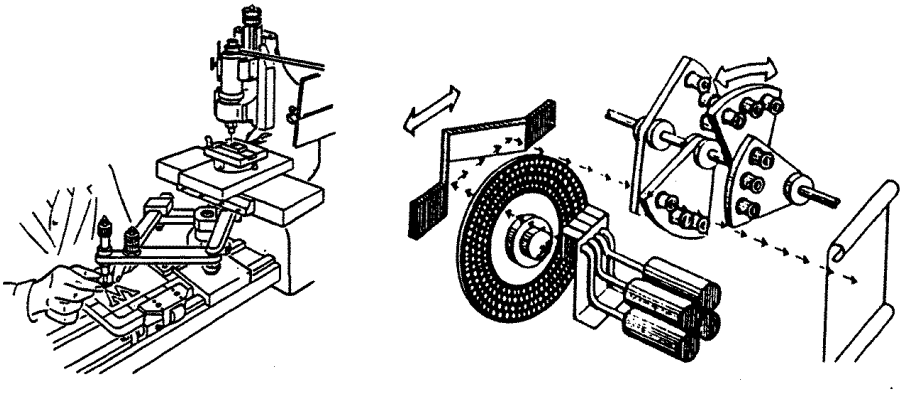


FIGURE 3 – *À gauche : pantographe pour graver les poinçons (d'après* MARSHALL *[48, page 46]) ; à droite : schéma de principe d'une photocomposeuse de seconde génération (d'après* SEYBOLD *[62, page 74]*

1.3. 1946 : photocomposition

DEUX INVENTIONS vont contribuer, au XX^{e} siècle, à la disparition du plomb : l'offset, procédé d'impression à plat basé sur la lithographie (W. RUBEL en 1904 à New York) et la photocomposition, en 1946 par les Français HIGONNET et MOYROUD [48, 49], cette nouvelle technique permettant de produire des films prêts pour les machines offset.

Le principe de la photocomposition est, *a priori*, simple (figure 3) : un faisceau de lumière passe à travers un disque où se trouve une lettre gravée. L'image est alors projetée sur un film photographique où elle sera donc en positif. Un système d'objectifs (puis de zoom) permet d'avoir diverses forces de corps et de placer les lettres les unes après les autres dans une ligne. La réalisation, en revanche, est plus compliquée et il a fallu remplacer les techniques mécaniques par l'emploi d'ordinateurs puis de laser pour avoir des photocomposeuses vraiment efficaces et de très haute définition.

La fabrication des caractères s'est ramenée alors à la photographie de lettres et à leur gravure sur le disque (avec bien sûr divers problèmes techniques, comme le calage). Toutefois il a souvent fallu, pour des problèmes d'*aliasing* ou de distorsion, redessiner les caractères (voir figure 4).

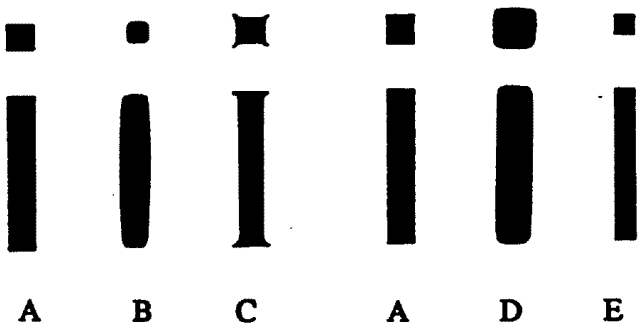


FIGURE 4 – *Les caractères doivent être redessinés en fonction de la technologie employée : en A, le caractère théorique ; en B ce qu'il donnerait en photocomposition ; en C, la façon dont il doit être redessiné pour qu'une photocomposeuse donne le dessin A ; en D, ce que donnerait une machine à écrire à partir de A ; en E le caractère redessiné pour qu'une machine à écrire donne le caractère A (d'après* FRUTIGER *[34]).*

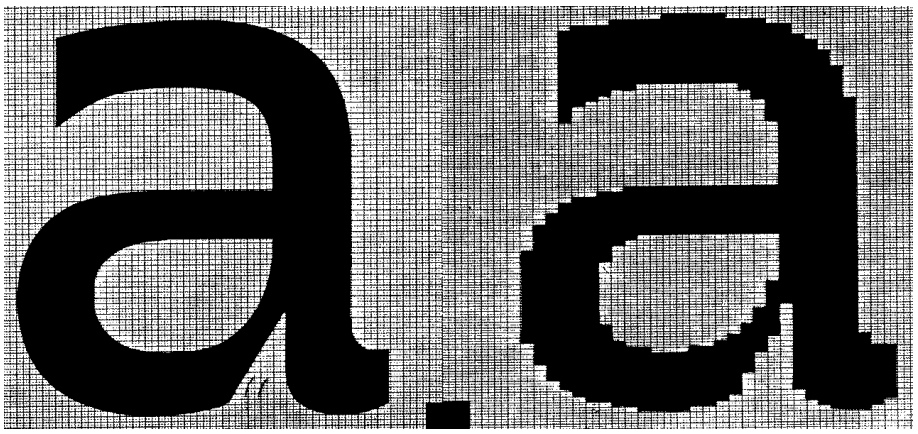


FIGURE 5 – *Les pixels des caractères numérisés ont d'abord été dessinés « à la main ». Ici, un a de* Galfa *conçu par* LADISLAS MANDEL, *à gauche en continu, à droite « prédigitalisé » (extrait de [57, page 46])*

1.4. 1966 : numérisation des caractères

Le passage de la photocomposition aux « fontes numériques » d'aujourd'hui n'est pas si immédiat qu'on pourrait le croire, surtout avec si peu de recul.

1. D'une part la photocomposition a suivi de très près ce qui commençait à se faire pour certains matériels informatiques (imprimantes mais sur-

tout tubes CRT[7] utilisant des images tramées (*rasters*) c'est-à-dire basées sur des *bitmaps* ou plans de bits) : les photocomposeuses sont passées du film photo aux premières *bitmaps* (c'est sur la Digiset 50TI qu'apparurent les premiers caractères numérisés) pour devenir machines de troisième génération avec l'usage de rayons laser. Mais c'est la notion de matrice qui prévaut encore. Les lettres pour photocomposeuses sont alors dessinées directement sur la grille (agrandie), pixel par pixel. C'est la technique employée par les pionniers du dessin de caractères numérisés : aux États Unis Ch. BIGELOW [16] et en France Adrien FRUTIGER[33] avec son *Univers* et Ladislas MANDEL [52], auteur, par exemple, du *Galfa*, un caractère spécialement dessiné pour les tout petits corps des annuaires téléphoniques (voir figure 5).

2. Une seconde convergence (informatique et matériels de secrétariat) a alors donné naissance à ce qu'on appelle parfois la bureautique : le début de la décentralisation des sites informatiques et des système distribués a amené les constructeurs d'ordinateurs à utiliser des machines à écrire comme périphériques ou terminaux, voire comme postes de saisie. IBM – qui était leader et en informatique et en matériel comptable ou de bureau – a alors sorti sa « machine à boule » et sa variante *Selectric* permettant de disposer de caractères à chasse variable. Les deux voies, informatique et secrétariat, ont alors évolué ensemble vers les « systèmes de traitement de texte ».
   Si, pour les secrétariats, c'était un très gros progrès, pour les typographes, ces systèmes étaient tout simplement inadmissibles[8] : non seulement la définition des caractères était très basse (les aiguilles des imprimantes faisaient quelques dizaines de millimètres de diamètre ce qui ne permettait guère de lissages des panses de lettres) mais en plus, pour des raisons d'économie de mémoire, les jeux de caractères étaient très limités, voire incomplets (pas ou peu de lettres accentuées) et souvent erronés (confusion entre l'italique et le penché par exemple). Il n'empêche que cette période, éphémère, a eu deux conséquences complémentaires :

   – d'une part, les secrétariats ont découvert la typographie et commencé à perdre les mauvaises habitudes causées par les contraintes des machines à écrire (emploi du souligné, chasses fixes, non accentuation des capitales, etc.) ;
   – d'autre part, le milieu très fermé des typographes a vu sa tour d'ivoire s'effondrer (mais certains se sont bien ouverts aux nouvelles tech-

7. Selon Lynn RUGGLE [60] la première utilisation d'un tube cathodique pour le tracé de lettres remonterait à 1964.

8. Hélas, les réputations perdurant, nombre de personnes croient qu'on en est encore à ce bas niveau sans ouvrir leurs yeux sur les progrès réels apportés depuis !

nologies !) et même si la formation est loin d'être encore suffisante, la typographie relève désormais du domaine public !

3. Enfin, on verra naître les caractères numérisés d'aujourd'hui définis non plus comme une image de points mais décrits par un contour que l'on remplit. Au delà du changement de méthode, il y a quelque chose de fondamental dans cette « mutation » : remplacer une image par la procédure qui la construit. Les premiers travaux remontent à la fin des années 60 (Mergler et Vargo aux USA). L'apport de Français aura été important : BLOCH, COUEIGNOUX et GUEDJ [26]. Il faudra attendre 1973 pour voir le premier système professionnel de dessin : *Ikarus* des Allemands KAROW, RUBOW et WEBER [40]. METAFONT de KNUTH [42, 44] ne date que de 1979 mais c'est bien plus qu'un système de dessin de fontes : c'est un système de meta-fontes (nous y reviendrons en parlant des *Multiple Masters*). C'est dans le début des années 1980 qu'apparaissent les premières normes en matière de « fontes » toutes liées à un « fondeur » : URW, Bitstream, etc.

4. En fait, la révolution commerciale viendra du laboratoire PARC *(Palo Alto Research Center)* de Xerox ou plutôt de ses transfuges qui sont à l'origine d'Apple et surtout d'Adobe qui a lancé PostScript et la notion de « langage de description de page ». Ce langage a non seulement été la première normalisation réussie de tous ces moyens de restitution (imprimantes, écrans, photocomposeuses, etc.), mais est vraiment à l'origine de la notion de « fonte » actuelle, c'est-à-dire d'un objet informatique que l'on manipule comme un simple fichier, que l'on copie à volonté, etc.

   C'est avec cette notion de fonte à la PostScript que fonctionnent aujourd'hui tous les systèmes commerciaux de saisie (comme *Typo* et *Ikarus*), de dessin (*Fontographer* ou *Fontstudio*) et de modification de caractères (*Fontoshop* ou *Illustrator*).

1.5. Remarques

Avec un peu de recul, on remarque maintenant plusieurs choses.

1. Le caractère, objet à trois dimensions, le « caractère » en plomb (avec la même ambiguïté que pour le mot gravure : quand on dit « caractère », s'agit-il de l'objet ou de sa marque imprimée ?) a été ramené à deux dimensions (le film des photocomposeuses) pour, finalement, être complètement dématérialisé : aujourd'hui, un caractère est un objet asbtrait comme peut l'être une procédure informatique. René Ponot dit d'ailleurs

que *la composition, numérisée ou non, n'a conservé de l'expression « caractère d'imprimerie » que ce qui se rapporte à la trace imprimée dudit caractère* [56].

2. Comme l'a fait remarquer Richard SOUTHALL [63], les rapports entre créateur et outils ont aussi évolué et sont loin d'être clairs aujourd'hui.

3. Chaque nouvelle technologie a d'abord « copié » les caractères de la précédente : le plomb a commencé par copier l'écriture manuscrite, la photocomposition n'a fait que copier les caractères du plomb et les premières fontes numérisées ont été des copies de l'existant. Il faut toujours un certain temps pour voir des caractères adaptés aux nouvelles technologies (comme *Univers* de FRUTIGER [33] pour les premiers caractères spécialement dessinés pour photocomposeuse ou *Lucida* de BIGELOW & HOLMES [17] pour les imprimantes à laser de basse définition).

2. Caractères numériques

L'ÉDITION électronique utilise trois types d'acteurs (produits logiciels ou matériels) : des formateurs (éditeurs ou systèmes de traitement de texte), des organes de sortie (imprimantes, photocomposeuses, écrans) et l'ensemble des « fontes ». Derrière ces outils se trouvent trois types de personnes, respectivement l'auteur (ou le compositeur), le lecteur et le dessinateur de caractères.

Le problème de la typographie numérique est de faire cohabiter les fontes dans ce système, étant donné que ces acteurs peuvent intervenir à des moments différents et que les informations passées varient d'un système à l'autre (la notion de fonte n'étant par exemple pas la même pour TEX, pour Word ou pour Illustrator).

La section précédente a montré que la numérisation des caractères était issue de la convergence d'intérêts divers mais confus. Sur un plan technique, cette évolution, encore en cours, s'est faite dans l'ordre suivant :

1. utilisation directe de plans de bits (ou *bitmaps*) ;
2. description des caractères par leurs contours ;
3. meilleure adaptation des contours à la grille en tenant compte des propriétés typographiques individuelles des caractères ; à ce niveau, on a la qualité de la photocomposition de seconde génération (films photo) ;
4. prise en compte des propriétés typographiques des caractères entre eux (ajustement optique, espacement entre caractères) ;
5. utilisation de niveaux de gris et non plus du seul rapport blanc/noir ;

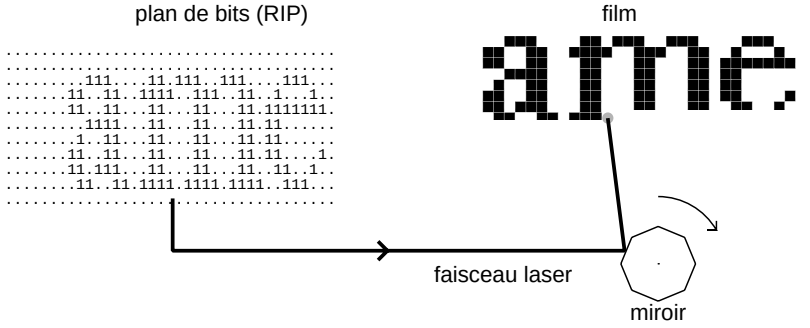


FIGURE 6 – *Schéma de principe d'un « moteur d'impression à trame »*

6. sans oublier les recherches en cours (par exemple sur la modélisation des caractères).

2.1. Notion de plan de bits ou *bitmap*

LES ÉCRANS, les imprimantes à laser ou à jet d'encre et la quasi-totalité des photocomposeuses utilisent aujourd'hui des caractères numériques, affichés, imprimés ou flashés sous forme d'une matrice de points. Ces moteurs d'impression sont tous basés sur le même principe : une photocomposeuse, par exemple, reçoit une matrice de points binaires (0–1, indiqués dans la figure 6 par un point ou le chiffre 1) ; une image de cette matrice est projetée par balayage d'un faisceau scanner (les valeurs binaires 0/1 devenant alors éclairé/non-éclairé). Les photocomposeuses ont souvent ce mécanisme d'impression dans une machine et l'interpréteur PostScript dans une autre (appelée RIP, *raster image processor*), tandis que, en général, les deux sont réunis dans les imprimantes à laser.

Les cases ou points de cette matrice s'appellent « pixels » (contraction de *picture elements*, éléments d'image) et les matrices sont les *bitmaps*, parfois appelées « grille de points discrets » et que nous appelons ici plans de bits. Les dimensions de ces pixels dépendent de la « définition » (traduction correcte de l'anglais *resolution*) de l'imprimante c'est-à-dire de la taille des grains utilisés (en photocomposition on utilise des films photographiques dont les sels d'argent sont très petits ; en xérographie – le procédé de photocopie utilisé actuellement par pratiquement tous les copieurs et imprimantes à laser – on utilise des grains de sélénium qui sont beaucoup plus gros) ou de la taille des gouttes du jet d'encre. Cette définition dépend aussi de la finesse du faisceau laser. On mesure cette définition en nombre de pixels noircis par centimètre mais l'origine américaine des matériels fait que l'on parle de dpi (*dots per inch*)

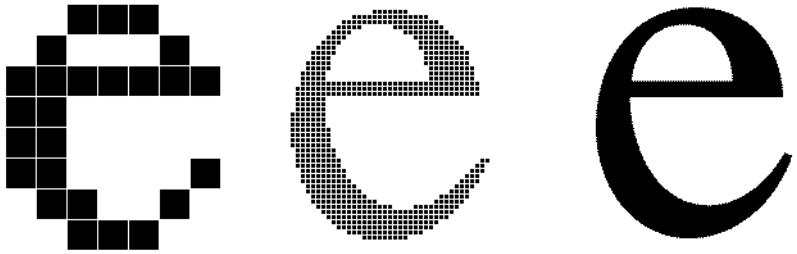


FIGURE 7 – *La qualité de la restitution d'un caractère dépend de la définition du moteur d'impression. Ici le même caractère en corps 10 (agrandi environ 15 fois) avec une définition de 300, 1200 et 4800 dpi. Voir figure 15.*

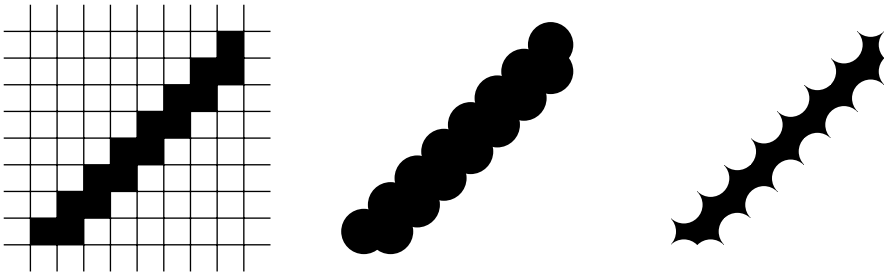


FIGURE 8 – *Un plan de bits* (bitmap) *théorique (à gauche) et son aspect réel issu d'un moteur d'impression à insolation noir sur blanc (au centre) ou blanc sur noir (à droite)*

ou de « ppp » (points par pouce). On a, par exemple, 300 dpi pour une imprimante à laser, 2400 dpi pour une bonne photocomposeuse, etc. (figure 7). Cette définition n'est pas obligatoirement la même horizontalement que verticalement (points rectangulaires de la figure 5).

On a l'habitude de représenter ces pixels comme des carrés ou des rectangles, mais en fait ce sont des taches plutôt circulaires ou ovales (la forme dépend des grains du tonner employé, du foulage de l'encre sur le papier, etc.). Par ailleurs, contrairement à ce que l'on croit, toutes les imprimantes ne mettent pas du noir sur une page blanche. Par exemple, certaines (Xerox, HP) peignent complètement la page en noir puis détourent les lettres par un faisceau de lumière : les points ne sont alors plus des ronds mais des étoiles à quatre branches entre les rond blancs (voir figure 8). Les systèmes de création de fontes peuvent en tenir compte, mais alors il ne faut pas utiliser une fonte d'un type sur une imprimante prévue pour un autre type (MACKAY [47]).

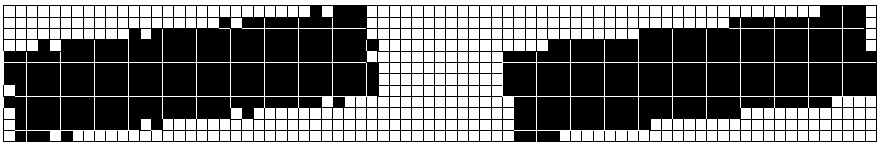


FIGURE 9 – *L'effet d'escalier d'une diagonale (à droite) peut être atténué en répartissant mieux les pixels* (half-bitting) *comme à gauche (d'après Beat* STAMM *[64])*

2.1.1. Défauts dûs aux plans de bits

Les plans de bits, matrices de points, relèvent du « discontinu » alors que les glyphes relèvent du « continu ». En nous inspirant d'une métaphore de Richard SOUTHALL, tout le problème de la typographie numérique est de vouloir faire avec des briques de Lego (et ce sans les casser) une Ferrari de 10 cm de hauteur. Un problème bien connu, par exemple, est celui des diagonales qui prennent presque toujours une allure d'escalier *(aliasing)*. L'art du dessinateur est alors de casser ces escaliers, par exemple en les atténuant (voir figure 9).

Un autre phénomène est celui des pointes qui ont tendance à s'émousser (ou du moins à paraîtres émoussées) et des creux qui ont tendance à se remplir (ou qui le paraissent). Là encore (voir figure 10), il faut ajouter ou supprimer des pixels, technique connue en anglais sous le nom de *half-bitting*.

2.1.2. Quand sont calculés les plans de bits ?

Ces plans de bits, fournis « en entrée » aux moteurs d'impression, sont donc la « sortie » des programmes d'impression. Il existe principalement trois façons de les construire : ou bien on les dessine à la main, ou bien on utilise des programmes et dans ces deux cas ils sont alors chargés dans l'imprimante comme une constante, ou bien on donne à l'imprimante une procédure qui les calcule.

Pendant longtemps ces matrices de caractères ont été créées (à la main ou à l'aide d'une souris sur écran) et ainsi chargées, sous forme de fichiers binaires, dans l'imprimante. Mais cette technique a deux inconvénients majeurs.

1. D'une part, une fonte complète – même après compactage – prend une place énorme : si on ne compte qu'une centaine de caractères pour une dizaine de corps (6, 8, 9, 10, 12, etc.) pour une imprimante à seulement 300 dpi (chaque caractère occupant en moyenne 12 pixels de hauteur sur 6 en largeur) ça fait au bas mot $100 \times 10 \times 12 \times 6 = 72000$ pixels, 4 fois plus (soit 288000 pixels) si on prend un romain, un gras, un italique et un

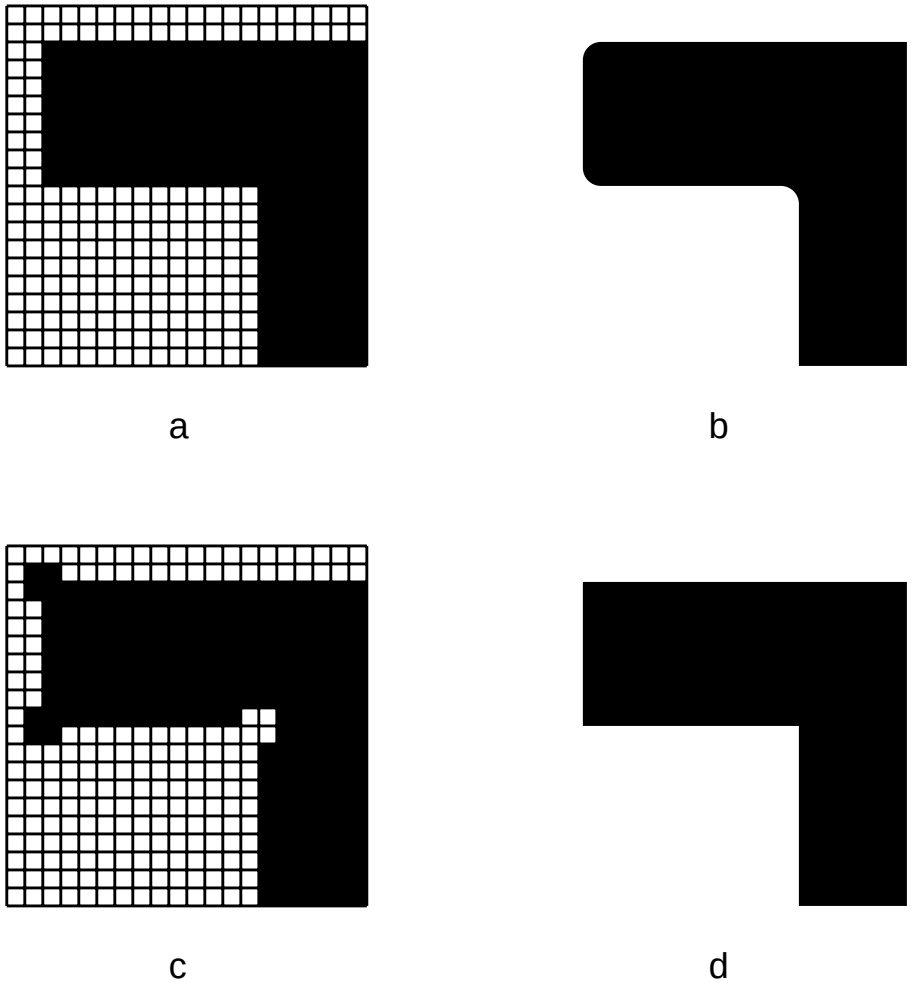


FIGURE 10 – *Si on donne le plan de bits (a), on obtient l'image (b) ; pour avoir l'image (d), il faut tricher et donner le plan (c). Voir aussi la figure 4. D'après* RUBINSTEIN *[59, pages 79–80]*

gras italique ; à 4800 dpi, ça fait dans les 28 millions de pixels. De plus ceci n'est valable que pour un type précis d'imprimante.

2. D'autre part et surtout, on ne peut pas faire subir à ces matrices de pixels des rotations (pour écrire en biais ou sur le pourtour d'un cercle par exemple) ni d'autres transformations géométriques sans risque de dégradation car la topologie de ces caractères est alors perdue. De même, les défauts montrés dans les figures 9 et 10 ne peuvent bien être corrigés qu'en fonction de cette topologie ; on a donc intérêt à la conserver le plus longtemps possible.

Ces caractères matriciels ne sont donc plus construits puis chargés une fois pour toute dans une imprimante. On les utilise toutefois encore dans certains cas.

1. METAFONT peut fournir des fontes au sens de PostScript. Mais en général, on sort ces fontes sous forme de plans de bits qui sont chargés dans l'imprimante au moment de l'impression. Deux raisons à cela : d'une part, METAFONT fait toutes les adaptations à la grille et sort donc des caractères de qualité et, d'autre part, il est rare que les textes produits par TEX aient besoin de faire des rotations sur des caractères, le nombre de fontes nécessaires restant alors raisonnable. L'inconvénient bien sûr est qu'un texte « formaté » pour une imprimante à 300 dpi reste à la définition de 300 dpi même sur une photocomposeuse à 4800 dpi.

2. Les plans de bits sont encore utilisés pour pratiquement tous les écrans : ils ont une définition assez grossière pour ne pas nécessiter trop de pixels par caractère. Et là non plus, on n'utilise guère de transformations affines sur les caractères. Par ailleurs, les techniques de *grey-scale* (caractères à niveaux de gris) permettent de rendre ces caractères d'écran de plus en plus lisibles.

2.1.3. Caractère gris

Pour une imprimante, un pixel n'a que deux couleurs : ou bien il est noir, ou bien il est blanc. Pour les écrans, ces valeurs noir/blanc sont en fait produites par un équilibrage des trois couleurs RVB ; en jouant sur leurs rapports on peut avoir, pour certains écrans, des pixels plus ou moins gris. Selon l'échelle, on a plusieurs « niveaux de gris ».

Ces caractères sont voués à un très grand développement. En effet, ils permettent une bien meilleure lecture sur écran que les caractères traditionnels, ce qui est fondamental pour les développement des systèmes multimédia. La

FIGURE 11 – *Caractères à niveaux de gris (dûs à Claude Betrisey, EPFL)*

notion même de contour (voir 2.2) est remise en cause par certains, comme Richard SOUTHALL [63] qui pense que l'on va ainsi revenir à des techniques de spécification de l'apparence visuelle comme du temps des graveurs de poinçons. Déjà, des typographes comme André GÜRTLER [39] s'y mettent très sérieusement et des produits commerciaux (comme FontoGrapher) permettent de créer des caractères à niveaux de gris. Les lectures de base sur le sujet restent le livre de RUBINSTEIN [59, pages 111–115] et la thèse d'Avi NAIMAN [51].

2.2. Définition des caractères par contours

LA principale méthode utilisée aujourd'hui consiste non plus à donner le plan de bits à l'imprimante, mais à lui donner une description géométrique du caractère et à laisser l'interprète PostScript (qui est dans l'imprimante) calculer ce plan de bits en fonction du corps et de l'orientation du caractère, en fonction de la grille de cette imprimante, mais aussi en fonction des *hints* (indications sur les « propriétés » typographiques du caractère).

Un caractère peut donc être défini par un ensemble non connexe de contours (*outlines* en anglais) orientés (ce qui définit l'intérieur et l'extérieur des zones à noircir). Ces « contours » peuvent être des courbes quelconques : en général, un « o » n'est pas réductible à deux cercles emboîtés ni un « i » à un rectangle surmonté d'un point. Il faut alors les « approcher » au sens mathématique. Le problème est donc de trouver une approximation qui soit peu coûteuse en place, peu coûteuse en temps et qui donne une bonne qualité pour le résultat final. Il faut aussi trouver des algorithmes qui, partant d'une courbe dont on connaît le contour, soient capables de colorier la surface correspondante.

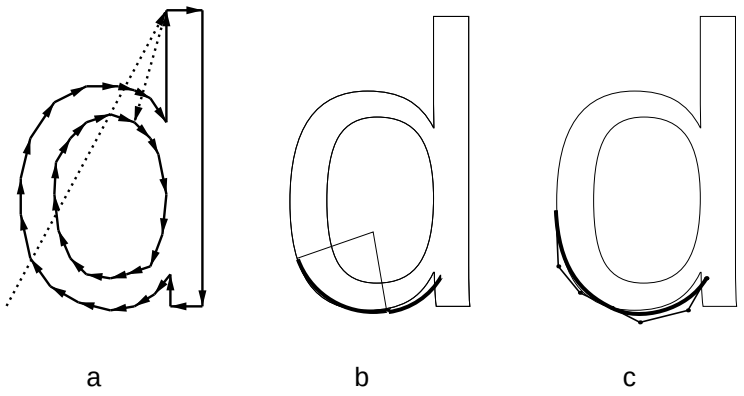


FIGURE 12 – *Approximation d'un « d » à l'aide de vecteurs (a), d'arcs de cercles (b) et de courbes de Bézier (c)*

On a d'abord utilisé, vers les années 1970, des vecteurs[9] (figure 12.a). Là encore, les tables prenaient trop de place et la méthode ne donnait pas des courbes très bien lissées.

On a ensuite remplacé ces vecteurs (qui sont des courbes du premier degré) par des morceaux de courbes décrites par des polynômes du second degré, notamment par des arcs de cercle (figure 12.b). Un arc peut être défini par son centre, le rayon et les angles des points d'extrémité et de départ. Des fontes entières ont été codées ainsi, notamment par *Bitstream* [62, page57]. URW puis *True Type* ont ensuite utilisé des quadratiques (coniques).

Mais les raccords entre morceaux de courbes ne sont pas toujours fameux avec des courbes du second degré. Maintenant, les caractères sont définis par des morceaux de courbes qui se mettent bout à bout et qui doivent avoir une certaine continuité à leurs jonctions (continuité d'ordre 1, tangente commune, par exemple) et que l'on appelle des *splines*. Parmi ces courbes se trouvent les polynômes du troisième degré (cubiques) dont les courbes de Bézier sont un cas particulier (figure 12.c).

L'utilité de courbes de plus haut niveau (notamment de degré 5, des quintiques) est encore à étude. Il semble qu'elles puissent avoir quelque avenir, notamment en calligraphie et dans l'étude de la jonction des caractères manuscrits [46].

9. C'est de là que vient probablement l'expression « police vectorielle » que l'on utilise encore (mais à tort puisqu'il n'y a plus de vecteurs dans la description actuelle des contours par courbes de Bézier).

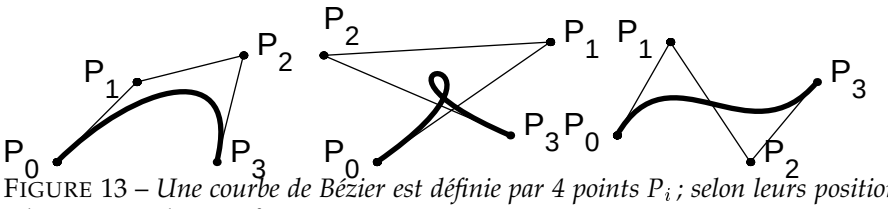


FIGURE 13 – *Une courbe de Bézier est définie par 4 points P_i ; selon leurs positions relatives, on a diverses formes*

Courbe de Bézier Ces courbes du troisième degré ont été mises au point par le Français Pierre BÉZIER alors qu'il travaillait chez Renault sur des programmes de dessin assisté par ordinateur de capots de voitures. On trouvera dans de nombreux ouvrages des études complètes de ces courbes, par exemple [30, 38].

Les cubiques sont définies par les équations paramétriques

$$\begin{aligned} x(t) &= a_x + b_x \cdot t + c_x \cdot t^2 + d_x \cdot t^3 \\ y(t) &= a_y + b_y \cdot t + c_y \cdot t^2 + d_y \cdot t^3 \end{aligned}$$

celles de Bézier se ramenant au polynôme

$$P(t) = P_0 \cdot (1-t)^3 + P_1 \cdot 3 \cdot t(1-t)^2 + P_2 \cdot 3 \cdot t^2(1-t) + P_3 \cdot t^3$$

avec $t \in [0, 1]$.

Voici quelques propriétés de ces courbes.

- Étant donné 4 points P_0, P_1, P_2, P_3, il existe une seule courbe de Bézier passant en P_0 et P_3, ayant P_0–P_1 et P_2–P_3 comme directions de leurs tangentes ; P_0, P_1, P_2, P_3 sont les points de contrôle.
- Les points P_0, P_1, P_2 et P_3 donnent déjà une idée de la forme de la courbe dont ils sont une enveloppe (ceci n'est pas vrai pour un arc de cercle : son centre, le rayon et ses angles d'extrémités ne donnent pas une idée visuelle de la courbure).
- Selon la position relative de ces points, on peut avoir des formes variées (figure 13). En déplaçant un seul point, on obtient des variations subtiles.
- Il existe une méthode simple de construction, dérivée du théorème de DE CASTELJOU [30] ; elle est basée sur la décomposition de la courbe $P_0P_1P_2P_3$ en deux courbes $P_0P'_1P''_1M$ et $MP''_2P'_2P_3$ (voir figure 14) en

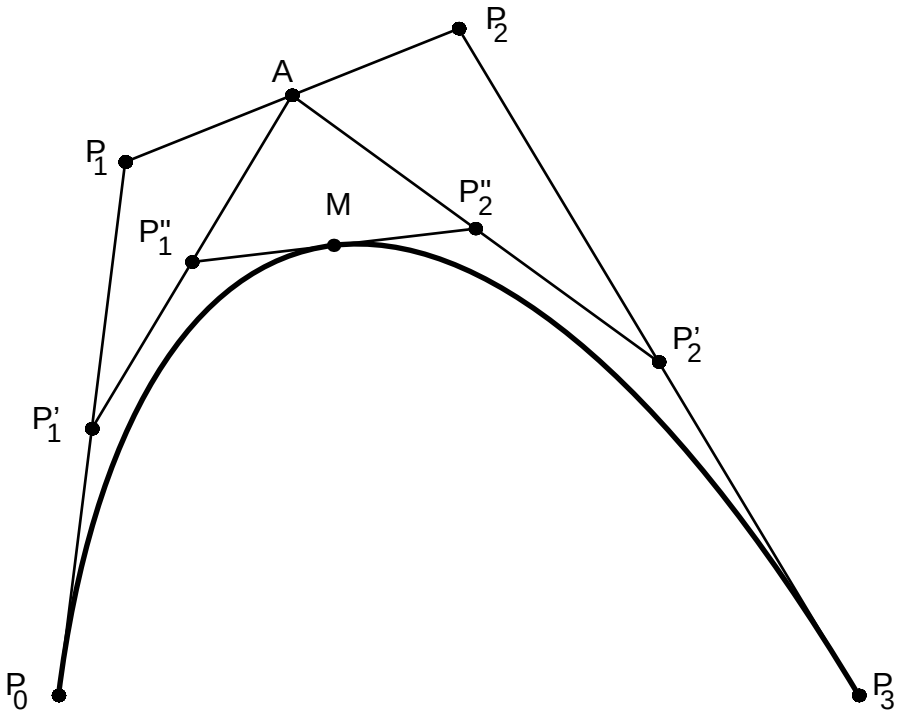


FIGURE 14 – *Une courbe de Bézier se construit récursivement à partir de ses points de contrôle*

calculant successivement les coordonnées des points

$$\begin{aligned} P'_1 &= (P_0 + P_1)/2 \\ A &= (P_1 + P_2)/2) \\ P'_2 &= (P_2 + P_3)/2) \\ P''_1 &= (P'_1 + A)/2 \\ P''_2 &= (A + P'_2)/2 \\ M &= (P''_1 + P''_2)/2 \end{aligned}$$

et en continuant récursivement sur chacune des moitiés ; basée sur des divisions par 2, cette méthode est très rapide, ce qui rend l'emploi des courbes de Bézier très efficace (voir ci-dessous 2.2.1).

– Ces courbes se raccordent facilement entre elles pour former des *splines* : elles ont les mêmes tangentes (continuité de la première dérivée) ce qui donne une impression de continu. Elles permettent de découper le contour d'un caractère en peu de morceaux (voir figure 15). Ce sont justement ces tangentes que proposent des produits commerciaux comme *Ikarus, Illustrator* ou *Fontongrapher*.

Description d'un caractère Voici (figure 15) comment est défini, en PostScript, le tracé des contours d'un « e » d'un *Times-Roman* pour un corps 1000. Ce n'est évidemment pas de cette façon qu'un caractère va être, en général, défini par un dessinateur typographe (voir section 2.5), mais c'est la forme finale interne des caractères.

La figure 15 montre, en grisé, le « e » correspondant et, en noir, les points de contrôle des courbes de Bézier associées.

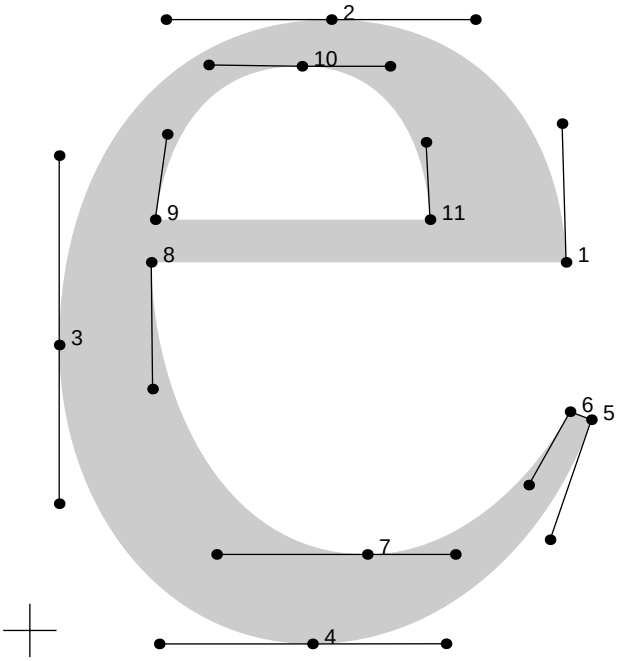
2.2.1. Algorithme de conversion ponctuelle

Supposons, dans un premier temps (nous y reviendrons en 3.1), que les caractères soient tous identiques (proportionnellement), c'est-à-dire qu'un « e » en corps 10 soit identique à un « e » en corps 1000, mais 100 fois plus petit et que la définition de la grille n'ait pas d'influence sur le caractère définitif.

Le problème est alors : ayant la description d'un caractère par contours (comme celle correspondant à la figure 15), étant donné la grille physique de l'imprimante, déterminer quels en sont les points qu'il faut noircir (pour obtenir par exemple l'un des « e » de la figure 7). On emploie pour celà un programme de conversion ponctuelle (*scan conversion algorithm*) ce qui permet alors de convertir la surface (correspondant au caractère) en un ensemble de points à noircir. Ces algorithmes très complexes ont fait l'objet de beaucoup de travaux, mais relativement peu ont été publiés. On trouvera toutefois des descriptions de ces techniques dans [38] ; voir figure 16.

2.2.2. Adaptation des caractères à la grille

Avec une photocomposeuse à 4800 dpi et pour des corps usuels (par exemple un corps 12), les caractères ainsi tracés ont un très bon rendu (voir figure 7.c). Il n'en est pas de même pour de très petits corps (comme les corps 4 ou 5 des annuaires téléphoniques ou les corps 6 des contrats d'assurance) pour une photocomposeuse, ni pour des corps 10 sur une imprimante à laser (figure 7.a) : dès que le nombre de pixels utilisés pour un caractère devient petit – par exemple lorsqu'un caractère a moins de 20 pixels de hauteur – les caractères risquent d'être dégradés.




```
\e{
402 276 moveto                    % départ point 1
399 380 334 458 226 458 curveto   % courbe de 1 à 2
102 458 22 356 22 214 curveto     % courbe de 2 à 3
22 95 97 -10 212 -10 curveto      % courbe de 3 à 4
312 -10 390 68 421 158 curveto    % courbe de 4 à 5
405 164 lineto                    % droite de 5 à 6
374 109 319 57 253 57 curveto     % courbe de 6 à 7
140 57 92 181 91 276 curveto      % courbe de 7 à 8
closepath                         % fermer contour extérieur
94 308 moveto                     % départ au point 9
103 372 134 424 204 423 curveto   % courbe de 9 à 10
270 423 297 366 300 308 curveto   % courbe de 10 à 11
closepath} def                    % fermer contour intérieur
```

FIGURE 15 – *Un caractère est défini par quelques points de contrôle et tangentes. En haut, le schéma d'un « e » ; en bas, le programme PostScript le décrivant.*

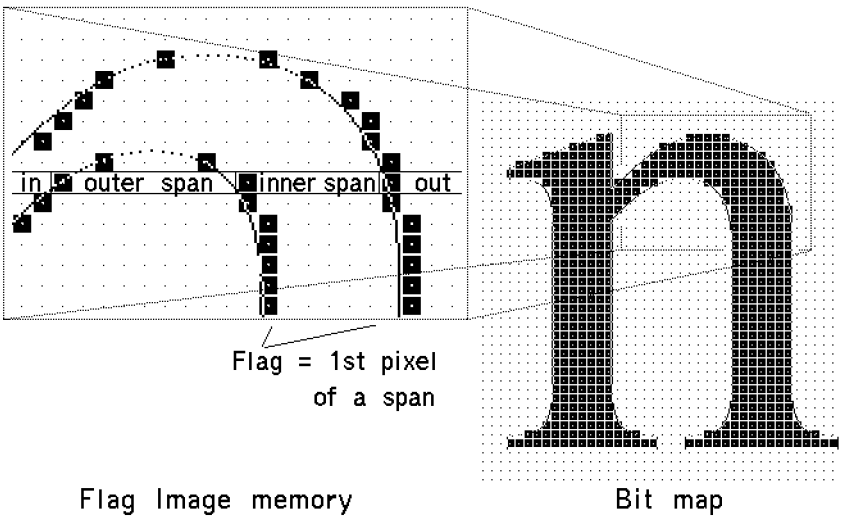


FIGURE 16 – *Détermination des pixels à noircir à partir d'une image où sont marqués les bords de zones intérieures* (inner span) *et extérieures* (outer span) – *d'après* R HERSCH *[38]*

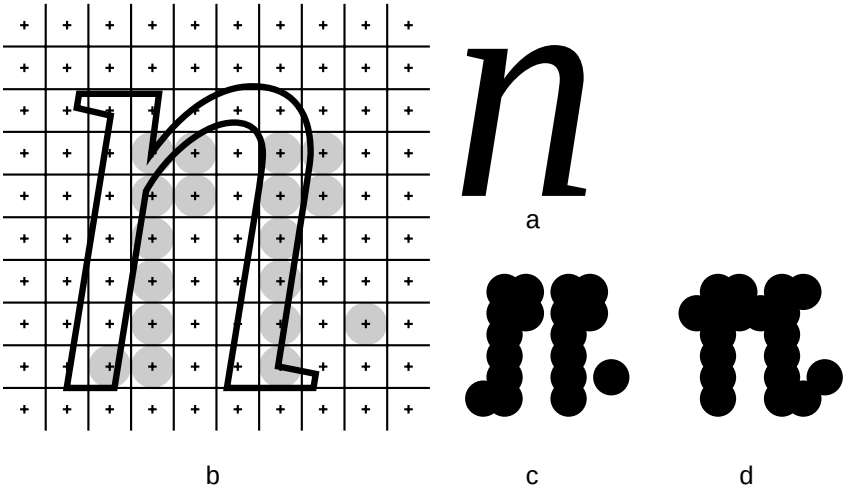


FIGURE 17 – *Adaptation d'un très petit « n » Zapf Chancellery à la grille : a) le caractère désiré, b) la grille et la projection du contour théorique, c) le caractère obtenu sans correction (notez les trous) et d) après correction*

Ces dégradations peuvent avoir des raisons diverses. En voici quelques-unes.

1. Prenons un « n » qui devrait ressembler à celui de la figure 17.a. Compte tenu du corps désiré et de la résolution de cette grille, seuls vont être « allumés » les points marqués en gris (figure 17.b) ce qui donnera l'impression de la figure 17.c : le centre du pixel de l'arche du « n » n'étant pas à l'intérieur de la surface théorique reste blanc, ce qui donne un trou. De même à droite de la jambe de droite.
2. Prenons maintenant le « n » de la figure 18.a. Cette fois, le contour théorique tombe de telle façon qu'un seul pixel se trouve (par ligne horizontale) à l'intérieur de la jambe de gauche alors qu'il y en aura deux dans celle de droite (figure 18.b). Il y aura déséquilibre, le « n » ayant deux jambes inégales.

Comment corriger ces erreurs ? En utilisant une « connaissance » de la topologie du caractère. On voit bien (figure 18-b) qu'il suffit de déplacer très légèrement la jambe de droite pourqu'elle ait la même graisse que celle de gauche (figure 18-d) ; de même en remontant très légèrement l'arche du « n » de la figure 17-b on obtient le « n » sans trou de la figure 17-d. C'est cette connaissance topologique qui est donnée par des *hints* (indications). Il existe plusieurs façons de donner ces indications – ce sont les standards *TrueType*, *Type1*,

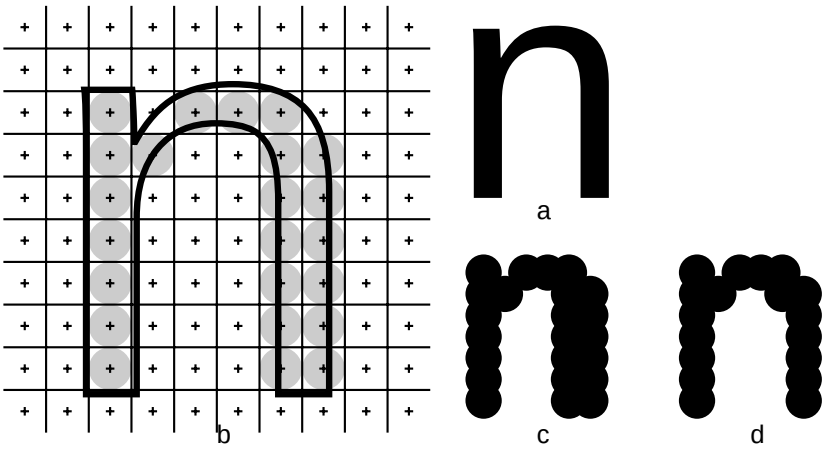


FIGURE 18 – *Adaptation d'un très petit « n » Helvetica à la grille : a) le caractère désiré, b) la grille et la projection du contour théorique, c) le caractère obtenu sans correction (notez la différence d'épaisseur des deux jambes) et d) après correction*

etc. En fait, on trouve soit des méthodes déclaratives (par exemple *Type 1*), soit des méthodes impératives (comme *TrueType* et METAFONT[10] dont il est issu). Cette dernière méthode considère qu'une fonte est un programme et que l'adaptation à la grille relève de ce programme, les instructions étant des instructions de déplacement des courbes avec toutes les instructions nécessaires de comparaison d'épaisseur de fûts, etc. Dans *TrueType* on a par exemple les instructions

```
17> MDRP[abcde]
23> ALIGNRP[]
33> SHPIX[]
```

qui s'appliquent aux points de contrôle numéro 17, 23 et 33 respectivement pour les déplacer *(MDRP=Move Direct Relative Point)* de façon à contrôler l'épaisseur ou la chasse d'un glyphe, aligner un point relativement à un autre ou déplacer un point selon un vecteur *(SHift point by a PIXel amount)*. De ce fait, il est très difficile pour un « artiste » de toucher à ces commandes.

10. On trouvera dans [37] un exemple d'utilisation des *hints* ou plutôt de *raster optimization* pour le dessin de caractères berbères.

En revanche, les commandes déclaratives sont séparées de la description du contour et viennent, en quelque sorte, en plus de la description des contours. Un système de création de fonte peut donc les inclure séparément dans la fonte à partir de commandes données par un typographe à l'aide de la souris. Ces commande sont du type « pas de trou à tel endroit » ou « la largeur ici doit être la même que là ». Elles concernent non plus les déformations des courbes mais de la grille elle-même. C'est pourquoi ces commandes sont liées aux *lignes bleues*[11] qui servent de cadre à un caractère. Par exemple

```
/BlueValues[-16 0 320 340 ...] ND
```

précise que la ligne de correction optique des bas de casse est en $y = -16$, que la ligne de base est en $y = 0$, que la ligne des minuscules est en $y = 320$ que celle de correction optique des minuscules est en $y = 340$, etc. Pour le « e » de la figure 19, on définira des lignes verticales *vstem* ou horizontales *hstem* avec une largeur définissant des zones sans chevauchement (c'est-à-dire des épaisseurs à respecter), par exemple

```
20  60 vstem
250 60 vstem
```

Il s'agit bien sûr d'indications propres à un caractère donné pour une police donnée et non quelque chose de général (du style « tous les « n » ont leurs jambes égales ») : c'est donc un typographe qui doit les donner, caractère par caractère, tout comme, autrefois (il y a au moisn dix ans !), seul un typographe corrigeait un par un les pixels d'un plan de bits.

De nombreux autres cas doivent ainsi être indiqués, par exemple pour éviter que deux capitales n'aient pas la même hauteur, pour qu'un « O » ne soit ni pointu ni aplati, pour traiter les diagonales ou les hampes d'italiques, etc.

Ce problème de l'adaptation des points à la grille est en fait un problème de conservation de propriétés géométriques, notamment des éléments de structure horizontaux et verticaux et il y a baucoup à attendre sur les recherches en matière de modèles topologiques.

2.3. Calcul des plans de bits des glyphes

Voyons maintenant comment ces divers algorithmes fonctionnent et utilisons PostScript comme exemple [2, 5].

11. Cette dénomination serait un hommage à Bob Dylan !

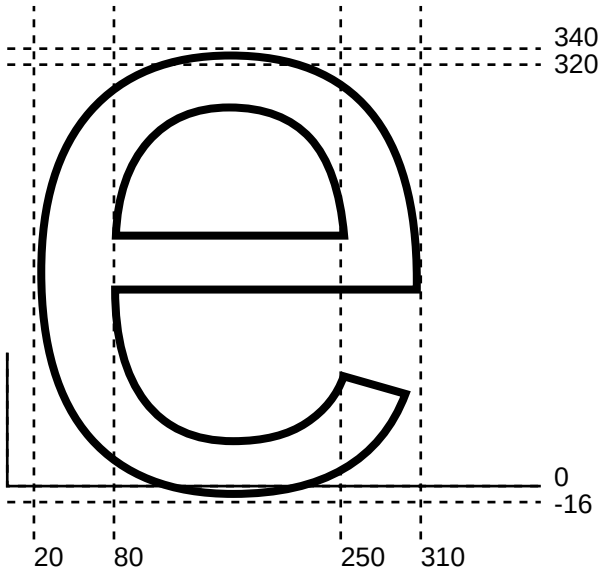


FIGURE 19 – *Indications* (hints) *à l'aide de « lignes bleues » (horizontales et verticales) pour garantir la conservation des propriétés géométriques du caractère*

2.3.1. Machinerie des fontes

Les algorithmes de *scan conversion* et de *hinting* sont effectivement appelés pour chaque caractère à imprimer. Mais comme ce sont des algorithmes très coûteux en temps, PostScript a prévu un mécanisme de sauvetage des plans de bits : on ne calcule le plan de bits d'un caractère donné (par exemple un « a » de *Times-Roman* en corps 12 orienté à 30 degrés compte tenu de la grille de telle définition, etc.) que si ce plan de bit n'est pas déjà en mémoire « cache » du programme en cours. S'il y est, on se contente de le recopier.

L'instruction `(a) show` provoque l'exécution du programme suivant :

```
consultation de l'état graphique
prise des informations globales sur la fonte

si le plan de bits de (a) n'a pas été dejà calculé
alors début
        appeler l'algorithme de scan conversion pour (a)
        appliquer les calculs de hinting
        construire le plan de bits
        le sauver en mémoire cache
      fin
prendre le plan de bits en mémoire cache
le copier dans la page au point courant
mettre à jour le point courant en fonction de la chasse de (a)
```

Le gros avantage de cette machinerie est que même si elle est coûteuse en temps, elle évite d'avoir à précalculer tous les plans de bits possibles et imaginables en fonction de la taille, de l'orientation des caractères et en fonction de la façon dont chaque imprimante est physiquement constituée.

2.3.2. Métrique des fontes

Les caractères en plomb avaient une largeur (appelée « chasse ») dépendant du caractère (un « m » est plus large qu'un « i ») et une hauteur (appelée « corps »). Voir figure 20, gauche. De part et d'autre de l'« œil » (la surface imprimable), se trouvaient des espaces de façon que les caractères imprimés ne se touchent pas, ni dans une même ligne (« approches » droite et gauche), ni d'une ligne à l'autre (« talus » de tête et de pied).

Ces notions « physiques », liées aux trois dimensions du caractère, n'ont plus de raison d'être maintenant. Mais, pour sauver le plan de bits d'un caractère, l'interpréteur PostScript a besoin de connaître la surface occupée : il demande

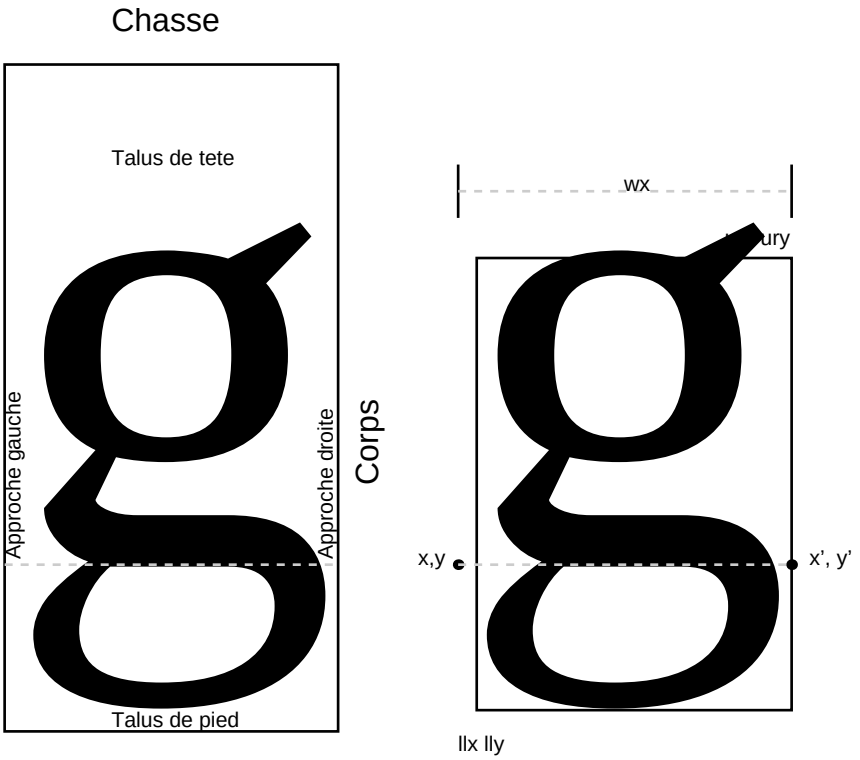


FIGURE 20 – *Comparaison de la métrique d'un caractère en plomb (à gauche) avec celle d'un glyphe en PostScript (à droite)*

alors qu'à chaque caractère soient associées les coordonnées du plus petit rectangle exinscrit – *bbox (bounding box)* – à ce caractère.

Une autre information indispensable à PostScript est la valeur de la chasse du caractère, c'est-à-dire de combien il faut avancer la valeur du point courant pour placer le caractère suivant sans qu'il touche au premier. Nous avons donné dans [3, 5] certains détails et raisons de ces « métriques » qui sortent du cadre du présent texte. Disons simplement ici que toutes ces informations sont regroupées dans un fichier, appelé TFM *(TEX Font Metric)*, voir [24, 58], ou AFM *(Adobe Font Metric)* [2] et que ce fichier est accessible par les formateurs (comme Word, FrameMaker ou LaTeX) qui doivent disposer de ces valeurs métriques pour faire tous les calculs de composition (notamment de justification et de division des mots) et de mise en page.

Enfin, PostScript utilise un mécanisme de codage *(encoding scheme)* des caractères : pour PostScript chaque caractère a un nom (par exemple *A* pour « A », *eacute* pour « é », *colon* pour « : »). Une table permet alors de passer des codes d'entrée (par exemple codes ASCII, ISO-LATIN1, EBCDIC, etc.) au caractère voulu (voir [8]). Ces tables de codage, assurant la portabilité, font aussi partie de la fonte. Pour (LA)TEX, voir [58].

2.3.3. Unités typographiques

Les typographes utilisent le « point typographique » comme unité de mesure. Le problème est qu'actuellement il y a au moins 4 définitions différentes de ce point. Avant de comparer ces valeurs, un peu d'histoire (voir [21]).

Pendant longtemps, on a donné des noms[12] aux tailles de caractères. Ces noms étaient, en général, basés sur des titres de livres ayant été composés dans ce corps. Ils étaient souvent charmants (*nompareille, mignone, gaillarde, triple canon,* etc.), mais ne correspondaient pas toujours à la même taille ; aussi, dès la fin du XVII^{e} siècle, on commença à essayer de normaliser ces valeurs. Il y eut d'abord TRUCHET puis surtout (car les travaux de cet académicien restèrent inutilisés en pratique) FOURNIER qui proposa (vers 1740) une première série de valeurs (proches du *pica* américain). Mais c'est François Ambroise DIDOT qui définit, en 1783, la valeur encore en usage aujourd'hui sous le nom de « didot » : un point didot valait alors un sixième de la ligne de pied du roi, c'est-à-dire à 1/72 de pouce (français) soit aujourd'hui 0,3759 mm.

Peut-être parce que cette unité était trop récente, le point didot n'a pas été intégré dans le système métrique lors de sa création vers 1800. Toutefois, l'Im-

12. On a aussi fait la même chose pour les papiers, par exemple *jesus*, et on fait encore la même chose en cette toute fin du XX^{e} siècle lorsque l'on parle de caractère « gras » ou « extra-maigre » !

TABLE 1 – *Les quatre points typographiques*

1 point didot	1/72 pouce français	0,3759 mm
1 point millimétrique		0,4000 mm
1 point pica	1/72,27 pouce	0,3515 mm
1 point DTP	1/72 pouce	0,3528 mm

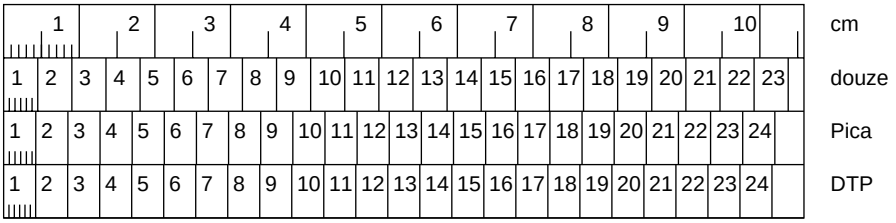


FIGURE 21 – *Comparaison des principaux points typographiques (d'après [5]) ; N.B. 1 douze = 12 didots, 1 Pica = 12 picas*

primerie royale, devenue nationale, a défini un « point millimétrique » de 0,40 mm, qu'elle seule semble avoir jamais utilisé.

Le point didot a été adopté pratiquement partout en Europe et dans le monde, sauf en Grande Bretagne, aux USA et dans les pays anglophones, où le vieux système de noms prévalait (et prévaut encore parfois !). En 1866, l'association des fondeurs américains se décida à adopter le système français, 1/72 de pouce, mais comme le pouce anglais était différent du pouce français, on obtint une unité légèrement différente : le *pica* dont le nom vient d'un caractère (de taille voisine du *Cicero*) appelé ainsi à cause de sa « couleur » : *pica* signifie « la pie » en latin !. Le développement des matériels américains a bien sûr fait que ce point pica a commencé à supplanter le point didot.

Vers 1897, la valeur du pica, est passée de 1/72 à 1/72,27 de pouce…Ce rapport 1/72,27 étant souvent difficile à calculer, les informaticiens ont pris, c'est notamment le cas de PostScript, l'habitude de l'arrondir à 1/72. D'où un nouveau point pica, que l'on appelle parfois DTP *(DeskTop Point)*.

En résumé, on a aujourd'hui 4 points typographiques (table 1). Ces différences ne sont pas grandes, mais peuvent ne pas être négligeables quand on cumule ces valeurs (voir figure 21) ou pour placer des signes diacritiques.

2.4. Fonte numérique

FINALEMENT, une fonte PostScript (mais c'est pratiquement la même chose pour les autres formats ; voir [58] pour (LA)TEX) est une structure informatique complexe comprenant de nombreux champs ou « formats » [5]:

1. pour l'interpréteur PostScript lui-même :
   - des informations globales sur la fonte (hauteur des capitales, des descendantes, etc.),
   - pour chaque caractère, un algorithme de dessin,
   - pour chaque caractère, des indications sur ses propriétés topologiques *(hints)*,
   - des tables de codage,
   - des tables de métrique ;
2. pour les formateurs en amont :
   - la métrique des caractères,
   - des plans de bits précalculés (par exemple pour affichage sur écran dans les systèmes Wysiwyg).

Ce sont ces divers fichiers que l'on achète sous le nom de « fonte ».

2.5. Saisie des fontes

Comment sont numérisés aujourd'hui les caractères, c'est-à-dire comment trouver leurs courbes de Bézier ou comment donner des indications pour mieux les adapter à la grille spécifique où ils seront tracés ?

2.5.1. Utilisation de tablette à numériser ou de scanner

C'est aujourd'hui la méthode la plus fréquente et en tout cas celle employée par les professionels d'aujourd'hui (avec des produits comme Ikarus ou Fntographer).

1. Le caractère à numériser est dessiné à la main (ou reproduit photographiquement et retouché), avec une grande taille (par exemple sur une feuille A4, en gros un corps 1000).
2. Un opérateur marque, à l'aide d'une tablette à numériser, les points des contours du caractère qui lui paraissent être les points importants (points d'inflexion, angles, etc.) ou bien on utilise un logiciel d'*autotracing*.

3. Le système en déduit les points de contrôle des courbes de Bézier ou les extrémités de lignes.
4. En déplaçant ces points à l'aide d'une souris, l'opérateur peut alors corriger le tracé final.
5. Enfin, il reste à mettre au point les chasses, approches, crénages, etc. et c'est là la grande utilité de ces logiciels qui offrent effectivement beaucoup d'outils ou fonctions appropriées pour ça !

2.5.2. Création directe sur écran

Des produits commerciaux comme *FontoGrapher* ou *FontStudio* permettent, en cliquant avec une souris, de tracer directement des courbes sur un écran. De nombreuses procédures permettent ensuite de modifier ou corriger ces courbes. Mais il est quasi impossible de définir une fonte de la sorte, ces outils servant plutôt à modifier des fontes du commerce à des fins graphiques. Voir cependant le travail de Hans MEIER en Suisse pour Barbedor [50].

2.5.3. Programmation

Une autre méthode est de programmer directement dans le langage utilisé (PostScript. METAFONT, etc.). Mais elle est très rarement employée ! C'est toutefois ce qu'ont fait, par exemple, Knuth pour *Computer Modern*, Yannis Haralambous pour des caractères non-latins [37], nous même avec le Delorme [6] et tout récemment (1996) Richard Southall et Ladislas Mandel pour dessiner un caractère spécial pour l'annuaire téléphonique de Detroit (USA). Mais dans tous ces cas, le but était de faire des caractères paramétrés (voir ci-dessous) ce qui n'est pas possible avec les outils cités plus haut.

2.5.4. Modélisation

Une façon complètement différente de dessiner des caractères est de considérer que ceux-ci sont formés d'éléments simples (par exemple un R est formé d'une barre verticale, d'un demi-cercle et d'une oblique) qu'il « suffit » de coller ensemble. Divers auteurs [1, 25, 54] ont proposé des modèles relevant de ce principe mais en vain, peut-être faute d'avoir su alors régler les problèmes de jonctions et de modélisation ?

2.6. À propos des mots fonte et police

Terminons cette section par une remarque. Les deux termes, police et fonte, ont été relativement malmenés ces temps derniers. Du temps du plomb, une

fonte correspondait à l'ensemble des types d'un « caractère » (par exemple le *Garamond*). Ça se vendait à la tonne ; on livrait avec une liste des types présents, par exemple « 1000 a romains en corps 12, 800 b romains en corps 12, etc. ». Cette liste s'appelait une police (ce mot vient de l'italien, *polizia* = liste ; c'est le mot que l'on retrouve dans « police d'assurance »). Lors de l'avénement de la Linotype puis de la photocomposition, on a appelé « police » non plus la liste des caractères, mais les caractères eux-mêmes, leur support matériel, matrice ou disque[13]. Mais le *Times*, par exemple, était formé de plusieurs polices, non seulement pour différencier le gras, l'italique, etc., mais aussi pour avoir plusieurs games de corps (par exemple pour les petits corps, pour les moyens et pour les grands).

Lorsque les fontes numérisées sont apparues sur le marché des systèmes de traitement de textes, les commerciaux ne se sont pas fatigués et ont traduit *font*[14] par *fonte* alors que, à l'époque, le mot police convenait mieux. Mais avec la notion de *font* telle qu'elle se trouve en PostScript, voire en METAFONT, il faut distinguer deux choses. Nous proposons donc d'appeler

fonte ce que l'on achète et qui permet de créer un certain nombre de polices (concept important avec les *Multiple Masters*) ;

police ce que l'on utilise à un moment donné, c'est-à-dire le résultat de

```
/FONT findfont [ . . . . . .] makefont setfont
```

En particulier, nous associons à « fonte » tout ce qui est lié à la métrique des caractères, à leur topologie, aux *hints*, etc. Tandis que « police » correspond au seul aspect « visuel ». C'est à peu près la même différence qu'entre « caractère » et « glyphe » (le premier étant l'entité théorique et le second la réalisation, ou l'instanciation, de cette entité – voir par exemple l'article de CH. BIGELOW sur Unicode [18]).

3. Caractères paramétrés

LES CONCEPTS que nous venons de montrer sont ceux utilisés dans les fontes commerciales les plus fréquentes d'aujourd'hui. Toutefois, depuis quelques années à peine (1991), on trouve certains fontes, comme les *Multiple Masters*

13. On a donc ici un bel exemple de glissement de sens, équivalent à celui du mot « bureau » qui a d'abord désigné une pièce d'étoffe, puis la table qui la portait, puis la pièce où se trouvait la table, puis l'immeuble, puis l'entité abstraite, etc.

14. Et ce serait une grossière erreur de croire que le mot *font* a, en anglais, un sens précis. Voir à ce sujet [65].

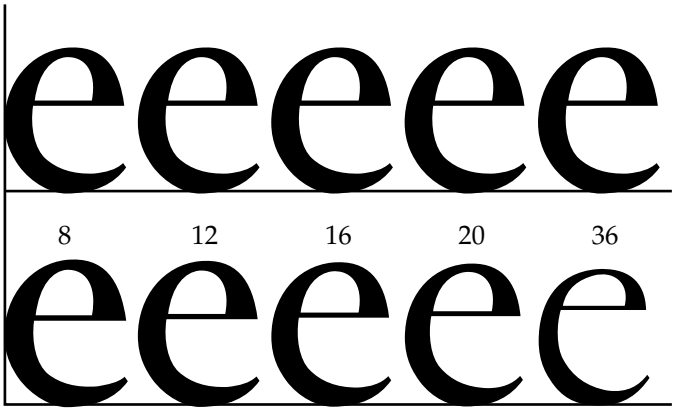


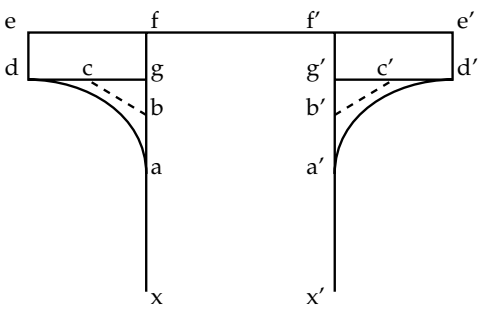
FIGURE 22 – *Caractères* Garamond *du corps 8 au corps 36, ramenés à la même taille ; en haut, sans ajustement optique, tous les caractères sont identiques ; en bas, avec ajustement optique, ils sont tous différents.*

d'Adobe, qui offrent de nouvelles possibilités, reprenant en ça des concepts existant dans certains systèmes comme METAFONT depuis longtemps !

L'idée de base (on trouvera dans [4, 12] de nombreux détails sur ce concept) est qu'au lieu de donner les courbes de Bézier définissant un caractère avec des valeurs numériques (comme en figure 15 où on écrit par exemple `402 276 moveto`) on peut utiliser des variables analytiques ou des paramètres (par exemple `x y moveto`). Avant de montrer comment on passe ces paramètres, voici quelques applications.

3.1. Ajustement optique

Les fontes ordinaires de PostScript sont telles que l'on passe d'un « e » Garamond corps 12 à un corps 36 en multiplant chaque coordonnée par 3 : les caractères ont donc tous la même forme (figure 22.haut). Or traditionnellement ce n'était pas le cas (figure 22.bas) : plus le caractère est petit, plus ses traits doivent être (proportionnellement) épais pour être visibles et plus son contre-poinçon (le blanc de la boucle) doit être gros (pour ne pas être bouché par l'encre). Il « suffit » pour celà de décrire chaque point de contrôle par une fonction du corps.




```
line x a ;
if serif
    {if garalde  {curve a b c d} ;
     if mecane  {line a g ; line g d} :
     line d e ; line e f}
    {line a f}
line f f′ ;
if serif
    ...
```

FIGURE 23 – *Description paramétrée d'un fût avec ou sans empattements*

metafont
metafont
metafont
metafont
metafont
metafont
metafont

FIGURE 24 – *À partir d'une seule description paramétrée, on peut produire diverses variantes d'un même caractère ; ici* Pandora *d'après [19])*

3.2. Famille de fontes

Les familles de caractères se déclinent selon certains styles ou allures. On a par exemple le même caractère en gras (ou demi-gras ou extra-gras), avec ou sans empattement, ou bien encore en italique, etc. Toutes ces variations peuvent également se décrire à l'aide de paramètres (figure 23).

FIGURE 25 – *Le caractère « aléatoire » Punk de Knuth [45] a été trancrit en PostScript type 3 [11], ce qui permet d'avoir un dessin différent pour chaque lettre (comparez par exemple les divers k, n et u).*

3.3. De METAFONT aux *Multiple Masters*

Pour permettre cette paramétrisation des fontes, plusieurs solutions sont possibles.

METAFONT Ce système étant procédural et calculant lui même les plans de bits, il n'y a aucune difficulté (théorique) à paramétrer une fonte. Le meilleur exemple est *Computer Modern* de Knuth [43]. C'est aussi la méthode employée par Neenie Billawala pour *Pandora* [19] (figure 24), Dan Berry [14], Yannis Haralambous [36], etc.

PostScript type 3 PostScript offre une variante à la machinerie des fontes qui permet de recalculer le plan de bits d'un caractère à chaque occurrence. C'est très couteux en temps, mais ça permet de paramétrer des caractères en fonction du contexte [12] et même de définir des caractères aléatoires (exemple : figure 25). Mais ces fontes ne sont pas compatibles avec les systèmes de *hints* et aucun logiciel commercial ne permet aujourd'hui de leur fournir les paramètres désirés !

Multiple Masters Adobe propose donc une solution intermédiaire qui consiste à construire à partir d'une fonte « maître » une fonte spécifique (par exemple avec telle graisse et tel type de patins) au moment où on charge le programme dans l'imprimante.

On trouvera dans [35, 61] et ici dans ce *Cahier* [22] plus de précisions !

4. Conclusion

Les notions de caractères et de fontes numériques ne sont donc pas simples. Ce qui complique probablement le problème, c'est que l'on peut avoir plusieurs visions de ces concepts et qu'effectivement chaque système (commercial ou non : TEX, Word, SGML, etc.) croit détenir la vérité !

Mais ce que nous voudrions surtout avoir montré, c'est que même s'il est vrai que les premières fontes numérisées avaient de très nombreux défauts, aujour-

d'hui la technique permet de faire aussi bien que du temps du plomb ; mais encore faut-il savoir ce qu'il faut faire, comment le faire et … l'utiliser.

Bibliographie

[1] D. ADAMS, « abcdefg, a better constraint driven environment for font generation », in [9, pages 54–70].

[2] Adobe Systems Incorporated, *Adobe Type 1 Font Format*, Addison-Wesley: Reading, 1990.

[3] Jacques ANDRÉ, « Métrique des fontes en typographie traditionnelle », *Cahiers GUTenberg*, n° 4, décembre 1989, 9–21.

[4] Jacques ANDRÉ, *Création de fontes en typographie numérique*, Mémoire d'habilitation à diriger les recherches, Université de Rennes I, septembre 1993.

[5] Jacques ANDRÉ & Justin BUR, « Métrique des fontes PostScript », *Cahiers GUTenberg*, 8, mars 1991, 29–50.

[6] Jacques ANDRÉ et Christian DELORME, « Le *Delorme* : un caractère modulaire et dépendant du contexte », *Communication et langage*, 86, 1990, 65–76.

[7] Jacques ANDRÉ, Jakob GONCZAROWSKI and Richard SOUTHALL, *Proceedings of the 3rd Conference "Raster Imaging and Digital Typography"*, numéro spécial *Epodd*, Wiley ed., vol.6(3), 1993.

[8] Jacques ANDRÉ et Michel GOOSSENS, « Codage des caractères : de l'Ascii à Unicode », *Cahiers GUTenberg*, n° 20, mai 1995, pages 1-53.

[9] Jacques ANDRÉ et Roger HERSCH (eds.), *Raster imaging and digital typography*, Cambridge University Press, 1989.

[10] Jacques ANDRÉ and Roger D. HERSCH, « Teaching Digital Typography », *EPODD – Electronic Pubishing, Origination, Dissemination and Design*, vol. 5, n° 2, June 1992, 79–90. Paru auparavant en français : « Enseigner la typographie numérique », *Bigre*, n° 79, mars 1992 ; et *Publication Interne Irisa*, n° 636, 1992.

[11] Jacques ANDRÉ et Victor OSTROMOUKHOV, « *Punk* : de METAFONT à PostScript », *Cahiers GUTenberg*, 4, 1989, 23–28.

[12] Jacques ANDRÉ et Irène VATTON, « Dynamic Optical Scaling and Variable Sized Characters », *EP-ODD – Electronic Publishing, Origination, Dissemination and Design*, 7(4), 1994, 231–250.

[13] Guy BECHTEL, *Gutenberg et l'invention de l'imprimerie – une enquête*, Fayard, 1992.

[14] Daniel BERRY and Johny SROUJI, « Arabic formatting with ditroff/ffortid », *Electronic Publishing – Origination, Dissemination and Design*, vol. 5, 1992, 163–208.

[15] Charles BIGELOW and Lynn RUGGLES (eds.), *The Computer and the Hand in Type Design*, numéro spécial de *Visible Language*, XIX, 1, hiver 1985.

[16] Chuck BIGELOW et Donald DAY, « La typographie numérique », *Pour la Science*, octobre 1983.

[17] C. BIGELOW & K. HOLMES, « The Design of Lucida: an Integrated Family of Types for Electronic Literacy », *Text processing and document manipulation* (van Vliet ed.), Cambridge University Press, 1986, 1–17.

[18] Charles BIGELOW and Kris HOLMES, « Création d'une police Unicode », *Cahiers GUTenberg*, 20, mai 1995, pages 81-102.

[19] Neenie BILLAWALA, « Pandora: an experience with METAFONT », in [9], 34–53.

[20] Gérard BLANCHARD, *Pour une sémiologie de la typographie*, thèse présentée à l'École pratique des hautes études en sciences sociales, Paris, 1980. {Les planches ont été publiées, sous le même titre, par les Rencontres de Lure et l'École des Beaux-Arts de Besançon chez Rémy Magermans, éditeur à Andenne, Belgique, 1979. La seule version disponible commercialement en est la traduction italienne : *L'eredita Gutenberg*, Gianfranco Altieri Editore, 1989.}

[21] Andrew BOAG, « The point : a chronology », *Typography papers*, n° 1, University of Reading (GB), 1996.

[22] Thierry BOUCHE, « Minion MM : installer une famille de fontes multimaster », *Cahiers GUTenberg*, n° 26, mai 1997, 45–70.

[23] Roger CHARTIER et Henri-Jean MARTIN (sous la direction de), *Histoire de l'édition française*, Fayard, 1990.

[24] Alain COUSQUER et Éric PICHERAL, « Polices, TEX et Cie », *Cahiers GUTenberg*, n° 9, juillet 1991, 3–31.

[25] P. COUEIGNOUX, « Character Generation by Computer », *Computer Graphics and Image Processing*, 16, 1981, 240–269.

[26] P. COUEIGNOUX and R. GUEDJ, « Computer generation of color planar patterns on TV-like rasters », *Proceedings of the IEEE*, vol. 68, n° 7, juillet 1980.

[27] Piero DE MACCHI, *L'avventura Didot: caratteri da stampa e nuove technologie*, Demacchi Progetti Grafici, Turin, 1994 (en anglais et en italien).

[28] John DREYFUS et François RICHAUDEAU (sous la direction de), *La chose imprimée*, Retz, 1977.

[29] Henk DROST, « Punch Cutting Demonstration », *in* [15, pages 99–105].

[30] Gerald FARIN, *Curves and Surfaces for Computer Aided Geometric Design*, Academic Press, San Diego, CA, 1982.

[31] Lucien FEBVRE et Henri-Jean MARTIN, *L'apparition du livre*, coll. L'évolution de l'humanité, Albin Michel, 1971.

[32] FOURNIER le jeune (Pierre, dit) *Manuel typographique utile aux gens de lettres & à ceux qui exercent les différentes partie de l'Art de l'Imprimerie*, Paris MDCCLXIV, 2 tomes. Réédition avec un troisième tome : H. CARTER, *On type foundry* (Londres, 1937) ré-éditée à son tour avec une introduction de James MOSLEY, Darmstadt, 1995.

[33] Adrien FRUTIGER, *Type Signe Symbol*, ABC Zurich, 1981 (édition trilingue).

[34] Adrien FRUTIGER, « Typography with the IBM Selectric Composer », *Journal of Typographic Research*, vol. 1, n° 3, 285–292.

[35] Mihel GOOSSENS, Sebastian RATZ, and Robin FAIRBAIRNS, « Using Adobe Type 1 Multiple Master Fonts with TEX », *TUGBoat*, vol. 16, n° 3, June 1995, 253–258.

[36] Yannis HARALAMBOUS, « Parametrization of PostScript Fonts through METAFONT – an Alternative to Adobe Multiple Master Fonts », [7], 1994, 145–158.

[37] Yannis HARALAMBOUS, « Typesetting Khmer », *EP-ODD – Electronic Publishing, Origination, Dissemination and Design*, 7(4), 1994, 197–216.

[38] Roger HERSCH (ed.), *Visual and Technical Aspects of Types*, Cambridge University Press, 1993. {Actes de l'« École Didot » tenue à Lausanne en septembre 1991.}

[39] R.D. HERSCH, J. BUR, C. BETRISEY, A. GÜRTLER, « Perceptually-Tuned Generation og Grayscale Fonts », SIGGRAPH, 1995.

[40] Peter KAROW, *Digital Formats for Typefaces*, URW Verlag, Hamburg (RFA), 1987.

[41] Peter KAROW, *Fonttechnology*, Springer-Verlag, Heidelberg, 1993. Paru en allemand sous le titre *Schrifftechnologie* chez le même éditeur en 1992.

[42] D.E. KNUTH, « The concept of a meta-font », *Visible language*, XVI,1, 1982, 3–27. Paru en français : « Le concept de metafonte », *Communication et langage*, n° 55.

[43] D.E. KNUTH, *Computer Modern Typefaces*, Addison-Wesley, Reading, 1986.

[44] Donald KNUTH, *The* METAFONT *Book*, AddisonWesley: Reading, 1984.

[45] Donal KNUTH, « A punk meta-font », *TUGboat*, vol. 9, no. 2, August 1988, 152–168

[46] Michael KOKULA, « Automatic Generation of Script Font Ligatures based on Curve Smotthness Optimization », *EPODD-Electronic Publishing*, vol. 7, n° 4, december 1994, 217–230.

[47] Pierre MACKAY, « Looking at the Pixels. Quality Control for 300 dpi Laser Printer Fonts, especially METAFONT s », in [53], 205–215. Traduit en français dans le *Cahier GUTenberg* n° 12, décembre 1991, 21–37.

[48] Alan MARSHALL, *Ruptures et continuités dans un changement de système technique – le remplacement du plomb par la lumière dans la composition typographique*, thèse, Grenoble, 18 décembre 1991. Parue comme *Publication interne Irisa*, n° 638, mars 1992.

[49] Alan MARSHALL (sous la direction de), *La Lumitype photon ...*, Musée de l'Imprimerie et de la banque, Lyon, 1996.

[50] Hand Ed. MEIER, « On the design of Barbedor and Syndor », in [38], 148–164.

[51] Avi C. NAIMAN, *High-Quality Texts for Raster Displays*, PhD Dissertation, University of Toronto, janvier 1985. Aussi : « Technical Report » (CSRI-253) Computer Systems Research Institute at the University of Toronto.

[52] Ladislas MANDEL, « L'écriture typographique : vers une prise de conscience », *Communication et langages*, n° 77, 1988, 5–30. Voir aussi *EPODD*, vol. 6, n° 1, 1993.

[53] Robert A. MORRIS & Jacques ANDRÉ (eds.), *Raster Imaging and Digital Typography II*, The Cambridge Series on Electronic Publishing, Cambridge University Press, 1991.

[54] M. NANARD, J. NANARD, M. GAMDARA & N. PORTE, « A declaratrive approach for font design by incremental learning », in [9], 1989, 71–82.

[55] Stan NELSON, « Mould Making, Matrix Fitting, and Hand Casting », *in* [15, pages 106–121].

[56] René PONOT, « Caractères d'imprimerie », art. de *Les sciences de l'écrit*, sous la direction de Robert Estival, éd. Retz, Paris, 1993, 103–106.

[57] Rencontres de Lure, *L'écriture télématique, année zéro*, Cahiers de Lure, 1985.

[58] Denis ROEGEL, « Les formats de fichiers DVI, GF, TFM et VF : que contiennent-ils et comment les visualiser ? », *Cahiers GUTenberg*, n° 26 (ce numéro), mai 1997.

[59] Richard RUBINSTEIN, *Digital Typography, An Introduction to Type and Composition for Computer System Design*, Addison-Wesley, Reading (USA), 1988.

[60] Lynn RUGGLE, *Letterform design Systems*, Standford University Technical Report STAN-CS-83-971, 1983.

[61] Jonathan SEYBOLD, « Adobe's 'MultiMasters' Technology: Breakthrough in Type Aesthetics », *The Seybold Report on Desktop Publishing*, vol. 5, n° 7, march 4, 1991, 3–7.

[62] J. SEYBOLD et F. DRESSLER, *La micro-édition selon Seybold*, Dunod, Paris, 1987.

[63] Richard SOUTHALL, « Character Description Techniques in Type Manufacture », in [53, pages 16–27], 1991.

[64] Beat STAMM, « Object-orientation and extensibility in a font-scaler », in [7], 159–170.

[65] Walter TRACY, *Letters of Credit – A view of type design*, Gordon Fraser, Londres, 1986.

[66] Michael TWYMAN *et alii*, *Making type by hand*, film vidéo, Typographic & Graphic Communication Department, Université de Reading, Angleterre, 1993.