UNIVERSITE DE NANCY I
U.E.R DE SCIENCES MATHEMATIQUES

CENTRE DE RECHERCHE EN
INFORMATIQUE DE NANCY

RECONNAISSANCE STRUCTURELLE DE CARACTERES MANUSCRITS ET DE FORMULES MATHEMATIQUES

THESE

Présentée pour l'obtention du
doctorat de spécialité en informatique

Soutenue le 29 octobre 1979

par

Abdelwaheb BELAÏD

devant le jury :

Président M. C. PAIR

Examinateurs MM. M. BERTHOD
J.P. HATON
M. MIGNOTTE
R. MOHR

UNIVERSITE DE NANCY I
U.E.R DE SCIENCES MATHEMATIQUES

CENTRE DE RECHERCHE EN
INFORMATIQUE DE NANCY

RECONNAISSANCE STRUCTURELLE DE CARACTERES MANUSCRITS ET DE FORMULES MATHEMATIQUES

THESE

Présentée pour l'obtention du
doctorat de spécialité en informatique

Soutenue le 29 octobre 1979

par

Abdelwaheb BELAÏD

devant le jury :

Président M. C. PAIR

Examinateurs MM. M. BERTHOD
J.P. HATON
M. MIGNOTTE
R. MOHR

REMERCIEMENTS

Je tiens à remercier Monsieur le Professeur C. PAIR, Directeur du Centre de Recherche en Informatique à Nancy (C.R.I.N.) et Président de l'Institut National Polytechnique de Lorraine (I.N.P.L.) de l'honneur qu'il me fait en présidant ce Jury.

Que Monsieur M. BERTHOD, Ingénieur à l'Institut de Recherche d'Informatique et d'Automatique (I.R.I.A.) trouve ici l'expression de ma profonde reconnaissance pour les échanges de vue intéressants qu'on a pu avoir ensemble et pour les conseils précieux qu'il m'a donnés.

Je remercie également Monsieur le Professeur M. MIGNOTTE de l'intérêt qu'il a porté à mon travail. Je n'oublie pas la formation qu'il m'a donnée tout au long de la maitrise d'informatique à l'Université Louis Pasteur de Strasbourg.
Il me fait, aujourd'hui, l'honneur de siéger à ce Jury.

Que Monsieur le Professeur J.P. HATON de l'Université de Nancy I veuille trouver ici l'expression de ma profonde gratitude pour les conseils précieux qu'il n'a cessé de me prodiguer au cours de cette recherche.
L'esprit qu'il fait régner dans son laboratoire y a facilité mon insertion et m'a permis de m'épanouir dans un cadre de recherche idéal.

Je ne pourrais pas oublier de remercier Monsieur R. MOHR, Maître-Assistant à l'Université de Nancy I de m'avoir proposé ce travail, de l'intérêt qu'il a manifesté tout au long de sa réalisation et de sa direction exemplaire.

Des remerciements tout particuliers à Madame Martine SCHWAAB avec qui j'ai débuté ce travail et qui n'a cessé de manifester son intérêt profond pour cette recherche.

Je tiens à remercier tous mes camarades du laboratoire d'Informatique de Nancy I. Je n'ai jamais cessé d'apprécier leur obligeance et amitié, en particulier G. MASINI, A. KACED et A. ADNAN pour les longues soirées d'hiver que nous avons passées ensemble à veiller le Mitra 15.

J'exprime enfin ma sincère gratitude à tous les responsables du C.R.O.U.S. de Nancy, et surtout Mademoiselle SANCHETTE, pour l'aide qu'ils m'ont apportée pour mener à bien ce travail.

Ce travail a été réalisé en partie dans le cadre de la convention SESORI 76-039.

Abdelwaheb BELAID

SOMMAIRE

CHAPITRE 1

INTRODUCTION

I.1 - PRÉSENTATION DU PROBLÈME

I.2 - ÉTAT ACTUEL DES RECHERCHES

I.3 - APERÇU SUR LES DIFFÉRENTES MÉTHODES D'ANALYSE

I.4 - CHOIX DE LA MÉTHODE STRUCTURELLE POUR LA DESCRIPTION DU TRACÉ MANUSCRIT

I.1 - PRESENTATION DU PROBLEME

La modélisation de systèmes "intelligents" pour la compréhension et l'interprétation automatique de formes complexes tend à se développer, de nos jours, dans le domaine de l'informatique.

L'étude des formes graphiques peut être considérée comme l'une de ces applications intéressantes de la reconnaissance des formes qui, comme l'écriture, le dessin ..., s'offrent à de très larges interprétations et où le concours de l'ordinateur y est très apprécié. C'est une des méthodes de communication intelligente et naturelle qu'on peut avoir avec la machine.

Le problème de l'écriture manuscrite reste encore au stade de la recherche ; les seules tentatives qui ont été faites, dans ce domaine, concernent plutôt la reconnaissance de l'écriture imprimée qui ne pose, à priori, que peu de difficultés à l'identification. L'invention des systèmes optiques pour la saisie instantanée des caractères a été une révélation ; le caractère est isolé dans le texte à l'aide d'une caméra et est transcodé sous forme directement assimilable par ordinateur. L'utilisation des masques pour la classification donne des taux de reconnaissance très satisfaisants (99,9 %).

Peu de recherches ont été faites sur l'écriture manuscrite qui est pourtant à l'origine de toute la mécanographie actuelle. Le traitement est, en effet, beaucoup plus complexe que celui des caractères imprimés (donnés sous un format standard) car il s'agit de tenir compte de toutes les variations de l'écriture et corriger toutes les déformations qui peuvent entacher le tracé du caractère à la saisie. Le rôle de l'ordinateur est donc de se substituer à la perception humaine pour reconnaître automatiquement le caractère déformé ou non en s'aidant éventuellement d'un apprentissage fait au préalable sur tout le jeu des caractères qu'on voudrait reconnaître.

Le dessin du caractère peut, en effet, s'éloigner rapidement de sa représentation idéale. On assiste souvent à un écrasement ou à une distorsion du caractère dus généralement à une écriture hâtive et très mal appliquée. Pour le reconnaître, il faut examiner l'allure générale de son tracé et essayer de lui donner une

structure reconnaissable en fonction de ses composants morphologiques les plus caractéristiques.

Le système qu'on cherche à mettre au point ici doit s'attaquer à cette forme de complexité sans poser de contraintes à l'écrivain ; de plus un traitement en temps réel peut être exigé si on veut activer le processus de reconnaissance et faciliter le dialogue entre l'écrivain et la machine.

L'entrée des caractères manuscrits peut se faire maintenant à l'aide de tablettes graphiques connectées directement à l'ordinateur.

Elles sont munies d'un stylo spécial qui envoie, au contact de leur surface et intervalles de temps réguliers, des coordonnées de points relatifs à un système d'axes. Un caractère de contrôle accompagne chacun de ces points. Il indique sa position dans le tracé : début - milieu - fin ; ce qui nous permet de localiser les levers de crayon (caractère intéressant pour le traitement qui va suivre). Les tablettes graphiques sont maintenant assez répandues, on les utilise également pour l'acquisition de dessins et les contours d'images.

Le choix du système d'acquisition et celui de la méthode de reconnaissance nous conduit à présenter notre travail en quatre parties :

- Une première partie sera consacrée à l'acquisition et au prétraitement des caractères. Un module d'échantillonnage allège le tracé du caractère et ne conserve que les points essentiels nécessaires à son traitement.

- Un module d'extraction sera chargé, dans une deuxième partie, de représenter chaque caractère par une suite de tracés élémentaires caractéristiques de sa forme. Un automate d'extraction sera exposé ici ; il montrera l'intérêt d'une telle segmentation en ces formes primitives.

- Nous montrerons dans une troisième partie comment se fait l'apprentissage avec cette description structurée et nous réserverons une large part de ce chapitre à la reconnaissance et la décision.

- Une quatrième partie sera réservée à l'étude et l'analyse des formules mathématiques qu'on a choisies ici comme application directe à la reconnaissance des caractères manuscrits. Elle validera nos idées sur la reconnaissance structurelle adoptée sur l'écriture manuscrite et nous permettra d'étudier le caractère dans un contexte plus large.

Une étude bibliographique précédera ce travail. Elle fera le bilan des différents travaux réalisés dans ce domaine et justifiera le choix de la méthode de description.

I.2 - ETAT ACTUEL DES RECHERCHES

Le procédé qu'on suit actuellement pour le traitement et la reconnaissance des caractères imprimés ou manuscrits consiste en la mise au point d'un système hiérarchisé sur trois niveaux (cf. figure 1).

a) **Au niveau inférieur** : saisie et prétraitement

Un module inférieur est chargé de la saisie du caractère et du prétraitement de son tracé.
Les deux modes d'acquisition couramment utilisés sont :

- La tablette graphique (caractère manuscrit uniquement) ; elle présente le caractère sous forme d'un vecteur de coordonnées de points.

- Le système optique (caractère imprimé ou manuscrit) ; il présente le caractère sous forme d'une matrice de points (0,1).

Les techniques couramment utilisées pour le prétraitement sont :

- Echantillonnage (sélection de points caractéristiques dans le vecteur)

- Lissage (correction des irrégularités par des fonctions spécifiques)

- Squelettisation (amincissement de l'épaisseur du caractère dans la matrice de points).

b) **Au niveau intermédiaire** : Traitement et analyse

Une méthode d'analyse est généralement à la base de ce traitement. Elle décrit la forme du caractère et lui substitue une structure exploitable pour son identification.

On distingue actuellement trois classes de techniques de description :

- les techniques statistiques
- les techniques géométriques
- les techniques structurelles ou syntaxiques

Nous reviendrons par la suite sur chacune de ces techniques et nous donnerons quelques exemples d'application.

c) Au niveau supérieur : Reconnaissance et décision

Un module d'identification est chargé ici d'affecter une classe à la forme précédemment décrite et de lui attribuer un taux de reconnaissance. Les systèmes actuellement mis au point sont plus ou moins performants suivant qu'ils traitent des caractères imprimés ou manuscrits.

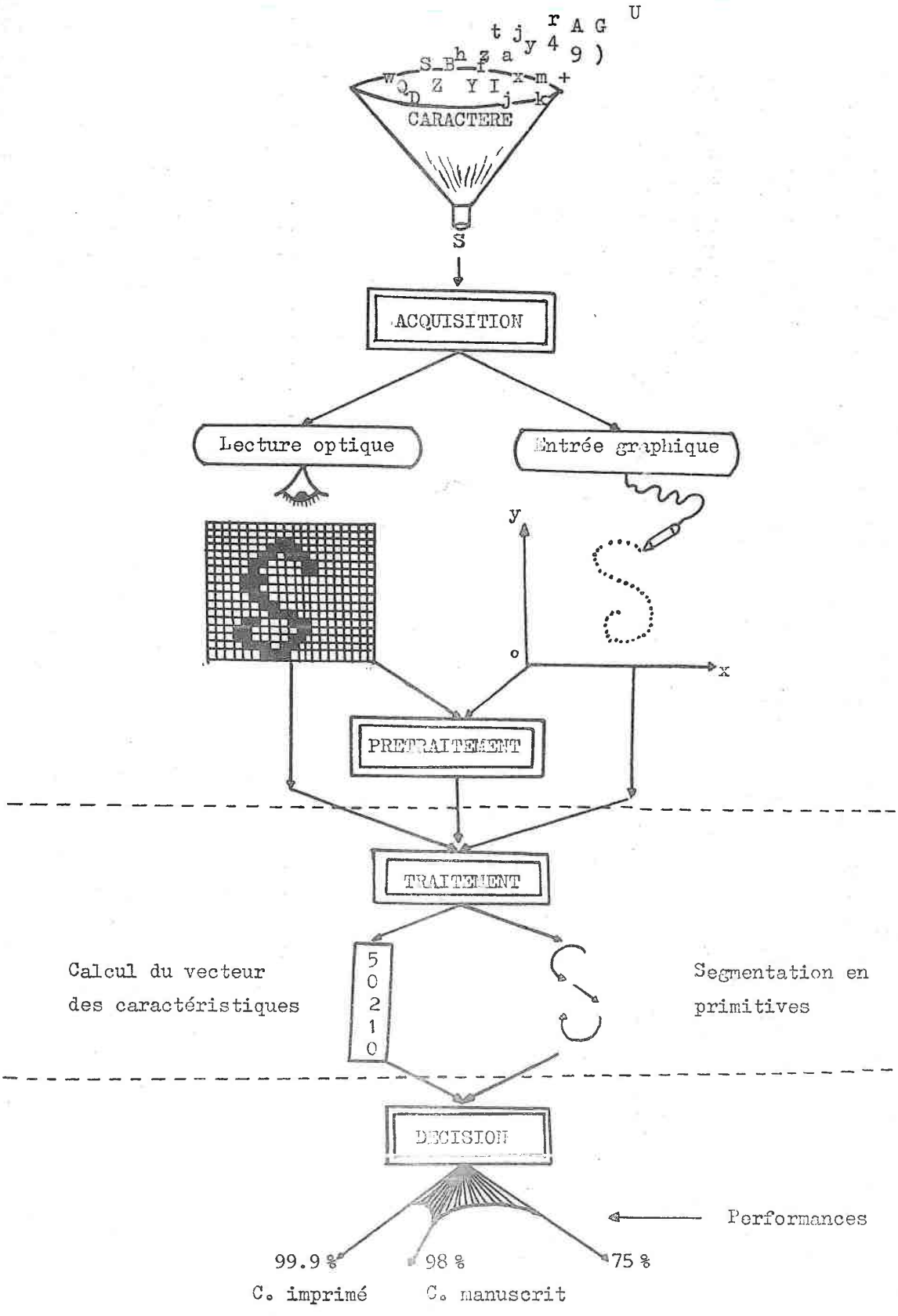


figure 1. Schéma général d'un système de traitement de caractères

I.3 - APERCU SUR LES DIFFERENTES METHODES D'ANALYSE

Diverses techniques ont été développées dans ce domaine pour la classification automatique des caractères.

Les premières d'entre elles ont fait appel à des théories statistiques qui consistent à faire des mesures localisées dans l'image du caractère et d'évaluer en fonction de ces mesures, la probabilité d'appartenance à une classe prédéfinie à l'avance.

La forme inconnue est affectée généralement à la classe définie par la plus grande probabilité.

Ces techniques ont été adaptées sur des caractères imprimés cadrés donnés par lecture optique sous forme de matrices binéarisées. Des filtres linéaires d'éclairement de points de l'image sont préconstruits par apprentissage pour chaque classe de caractères et sont appliqués successivement sur l'image du caractère entré et décident ainsi de son appartenance à telle ou telle classe.

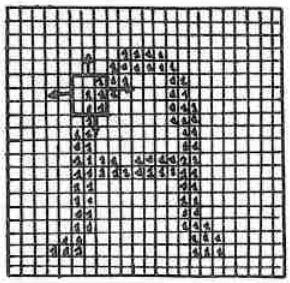
L'effortfourni par ces méthodes réside dans la construction de filtres ou masques. Elles ont abouti à des résultats très bons mais s'avèrent lentes à cause du calcul des distances.

Voici quelques exemples de description basée sur cette technique :

HUSSAIN et DONALDSON (1) représentent l'image du caractère par un vecteur de 25 composantes dont chacune contient le nombre de points trouvés dans la subdivision correspondante de l'image. Ce vecteur de comptage est ensuite comparé à tous les chemins d'un arbre de décision construit par apprentissage pour tous les modèles de caractères à étudier. Le chemin qui présente la même succession de nombres de points conduit au nom du caractère.

PATTERSON (2) travaille sur une image quadrillée. Il fait déplacer une grille 3 x 3 dans le plan de l'image et calcule le nombre de points trouvés dans cette grille (cf. figure 2).

Figure 2



Par combinaison linéaire de ces réponses, il essaie de retrouver le modèle qui offre la réponse la plus proche par calcul de la distance et minimisation des erreurs au sens des moindres carrés.

CASEY et NAGY (3) ont utilisé une approche similaire pour la reconnaissance des caractères chinois imprimés. C'est l'approche par assortiment de modèles (Template matching) qui consiste à construire des matrices de comparaison de mêmes dimensions que la matrice entrée et de les comparer entre elles. La matrice qui offre la même ressemblance avec la matrice d'entrée (même répartition de points) donne le nom du caractère entré.

Notons les travaux de CARCENAC de TORNE (4) qui a conçu une machine d'identification d'adresses postales imprimées pour le tri automatique du courrier. L'adresse est localisée sur l'enveloppe puis chaque caractère est isolé dans sa ligne par un système de segmentation suivi de prétraitement. Ensuite, à chaque caractère est attribué un vecteur de 100 composantes binaires précisant l'absence ou la présence de caractéristiques géométriques nécessaires à sa classification. Ces résultats sont obtenus par application de "Templates" dans des zones prédéfinies de l'image digitalisée du caractère. Après décomposition syntaxique de ces vecteurs et calcul des distances à chaque classe de comparaison, le caractère est assimilé à la classe la plus proche au sens de ces mesures.

SPANJERNSBERG (5) a mis au point deux systèmes de reconnaissance. Le première opère sur les moments d'inertie ; le deuxième sur les "points de vue" par détection de certains motifs caractérisant les propriétés métriques du caractère telles des pentes, des îles ...

Une étude un peu particulière de DEMARTINI (6) qui se trouve à la limite de cette technique. Elle concerne essentiellement l'écriture manuscrite entrée depuis une tablette graphique. Elle évalue la résultante ($\vec{F}$) des forces appliquées par l'écrivain sur la surface de la tablette au cours du tracé.

$$\vec{F} = M \frac{d\vec{Z}^2}{dt^2} + \lambda \frac{d\vec{Z}}{dt}$$

M est la force appliquée.

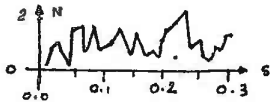
Z : affixe du point en cours de tracé

$\frac{d^2\vec{Z}}{dt^2}$ = accélération du mouvement

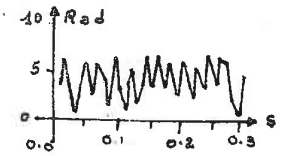
λ : la force de frottement

LETTRE 1 A2

AMPLITUDE FORCE



PHASE FORCE



La classification des caractères se fait par comparaison des vibrations enregistrées. Le graphe le plus proche donne le nom du caractère.

Cette méthode parait assez prométrice, car on peut très bien l'envisager pour la reconnaissance de l'écriture cursive.

Citons enfin le travail de TEITELMAN (7) qui partage la zone du plan du caractère en 4 zones se chevauchant (voir figure 3) et attribue à chacune d'entre elles un vecteur de composantes binaires indiquant la présence ou l'absence de points noirs dans la zone correspondante. Par combinaison linéaire de ces 4 réponses, il déduit le nom du caractère.

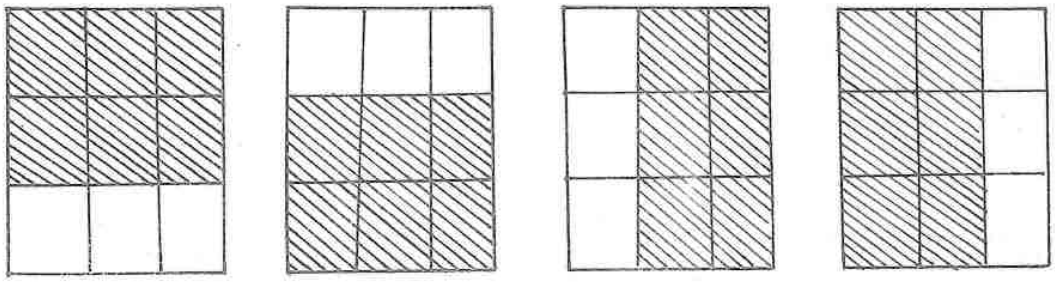


Figure 3 - 4 zones de recherche de points noirs

D'autres travaux importants relevant de la même méthode statistique sont à citer : MONSON (8), TOUSSAINT et DONALDSON (9), OTT (10), CANDREW (11) et SIMON (12).

Une autre technique de classification appelée la technique géométrique consiste à appliquer une série de tests portant sur des traits distinctifs du caractère et de comparer leurs réponses à celles données par apprentissage sur des caractères de référence. Ces tests sont généralement préparés à l'avance et leur ordre d'application peut être dans certains cas important.

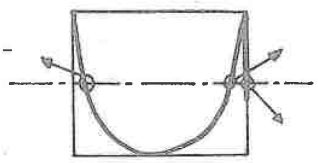
Cette approche a été utilisée aussi bien pour l'identification des caractères imprimés que pour la classification des caractères manuscrits. Sa performance est fonction de la rigidité des tests et de leur mode d'emploi.

MILLER (13) a adapté cette méthode sur des caractères manuscrits latins entrés depuis une tablette graphique. Il divise chaque portion du caractère délimitée par rebroussement et levers de crayon en 6 traits de même longueur. Chaque trait est ensuite codé par sa direction dans le plan du caractère. Il engendre ainsi un vecteur de codes sur tout le caractère qu'il compare par la suite à des codes prédéfinis rangés dans un dictionnaire. La reconnaissance du caractère est améliorée en cas d'ambiguité par des tests portant sur l'absence ou la présence de traits dans certaines portions du tracé.

GAILLAT (14, 15) a également appliqué cette méthode avec tous les avantages qu'elle présente. Une batterie de tests pertinents prédéfinie dans une première phase d'apprentissage, est destinée à être appliquée sur chaque caractère entré. Cette batterie de tests est répertoriée en trois séries :

- une série de 10 tests appliqués sur chaque primitive du caractère concernant la position de leurs extrémités dans des zones privilégiées du plan du caractère.

- une série de 2 tests portant sur des caractéristiques particulières du caractère tel le franchissement par le tracé du caractère de l'axe médian.



- une troisième série de 2 tests portant sur l'espacement des extrémités des primitives.

Ces tests ne seront pas appliqués dans leur totalité sur chaque caractère. La procédure de décision choisit automatiquement les quelques tests nécessaires pour séparer les différentes classes et minimiser la probabilité d'erreur.

VOVAN (16) utilise cette même méthode pour la reconnaissance des caractères latins et chinois entrés par lecture optique sous forme de matrices digitalisées. Sa technique consiste à obtenir, par balayage horizontal et vertical de l'image du caractère, une série de paramètres pertinents peu sensibles aux légères variations et permettant son identification.

POWERS (17) extrait du tracé du caractère des primitives de type arcs de cercle définis par une suite de traits formant entre eux des angles de même signe. Il définit ainsi un treillis d'arcs de cercle pour la comparaison.

D'autres travaux utilisant une méthode analogue seront cités en bibliographie (18 à 25).

Notons enfin une troisième classe d'approches appelée l'approche structurelle ou syntaxique . Pour cette méthode, un caractère peut être décrit par une juxtaposition de primitives élémentaires de type trait-courbe-lever par exemple, et de relations topologiques précisant les liens syntaxiques entre elles tels que le raccordement et le rebroussement. Reconnaître un caractère avec cette méthode revient à extraire ces sous-formes de son tracé ou de la matrice de points et de comparer la chaine de primitives engendrées à celles faites par apprentissage sur des modèles de caractères. Le modèle qui offre la même succession de primitives donne le nom du caractère.

BERTHOD (26) utilise cette méthode pour la reconnaissance en temps réel des caractères manuscrits entrés depuis une tablette graphique. Après échantillonnage du tracé (27), un algorithme de suivi de courbe découpe chaque tracé du caractère entre deux levers de crayon, en primitives très évoluées tels le trait, courbe directe, courbe rétrograde et rebroussement. Des grammaires spécifiques à chaque classe de caractères sont appliquées successivement sur le code engendré. La gram-

maire qui décrit complètement le code ressorti donne le nom du caractère entré. En cas d'ambiguité, des tests portant sur des attributs de longueur et de direction sont introduits par la suite ; viennent renforcer le code de chaque primitive.

Figure 4 - code du D

Trait - rebroussement - trait - lever - courbe

L. SHAPIRO (28) a défini un langage de description assez original : le ESP^3 . C'est une extension du langage de programmation SNØBOL 4 . Elle établit, pour chaque forme, un modèle SNOBØL qui décrit la succession des primitives qui la composent et les relations topologiques qui la caractérisent.

Le dessin du caractère T représenté par la figure 5 peut être décrit par le modèle suivant :

```
T = LINE/ANGLE EQUAL 0/ $ L1 ξ LINE
+ / START AT *POINT (L1 , 'MID'),
+ ANGLE EQUAL 270 / $ L2
```

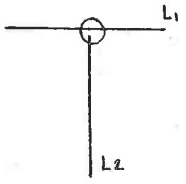


Figure 5

Le modèle regroupe les trois informations syntaxiques caractéristiques du tracé du T, à savoir :
- l'ordre d'entrée des deux lignes
- la direction de chacune de ces deux lignes,
- et la position du point d'intersection dans le plan du caractère.

YOSHIDA et EDEN (29) distinguent sept primitives représentatives du caractère manuscrit chinois (voir figure 6). Chacune de ces primitives est accompagnée par une classe de longueur et une information concernant sa position dans des zones privilégiées du cadre du caractère.Le code généré est comparé à un dictionnaire de codes de caractères de référence .

Figure 6 - 7 types de primitives caractéristiques du caractère chinois

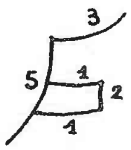


Figure 7 - Exemple de caractère

RAJASEKARAN et DEEKSHATULU (30) travaillent sur une image digitalisée du caractère manuscrit indien (Telugu) et utilisent un codage assez lourd basé sur la quantification des directions des traits joignant les points successifs de l'image. Ils forment dans une phase préliminaire 21 types de primitives (voir figure 8). Chaque primitive regroupe 3 ou 4 classes de directions parmi les 8 directions principales du plan.

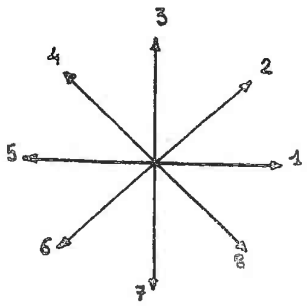


Figure 8 a - Directions principales

A1 = { 1 , 8 , 2 }

A2 = { 1 , 2 , 8 }

B1 = { 1 , 3 , 2 }

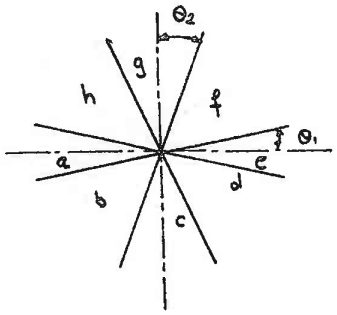
.....

I1 = { 3 , 5 , 4 , 6 }

Figure 8 b - Primitives détectées

Des automates sont utilisés par la suite pour la classification des 2 000 caractères Télégu. Des informations de type nombre et position des points d'intersections des divers tracés, ou nombre d'extrémités de chaque caractère, viennent renforcer la description du caractère en cas d'ambiguité.

C'est suivant ce même principe de quantification des directions que ITO et CHUI (31) ont conçu un système de classification des caractères manuscrits par les primitives. Les classes de primitives retenues sont de type segments de droite de différentes directions et arcs de cercle assez représentatifs du caractère latin. Chaque classe d'arc est définie par un code de directions spécifiques. Des grammaires de modèles sont mises au point pour la classification.



Primitives : J : cbahgf

J
)
v
C
O
S

Citons enfin les travaux de CHEHIKIAN (32) qui opère sur chaque point de l'image du caractère donné par une matrice (32 x 16). Le procédé consiste à analyser l'environnement de chaque point et de le classer comme appartenant à l'un des 4 types de segment de droite : horizontale, verticale, oblique à pente positive et oblique à pente négative.

Nous citerons en bibliographie d'autres travaux basés sur cette description structurelle en primitives (33 à 41).

I.4 - CHOIX DE LA METHODE STRUCTURELLE POUR LA DESCRIPTION DU TRACE MANUSCRIT

Notre choix a porté sur les techniques structurelles qui constituent à notre avis la voie d'approche la plus naturelle pour le traitement et l'identification des formes graphiques en conception assistée.

Nous considérons, en effet, qu'une forme graphique a une structure réductible en ses composants morphologiques les plus élémentaires. L'extraction de ces sous-formes appelées primitives constitue leur ultime reconnaissance. Pour que de telles approches soient néanmoins efficaces et performantes, il faut que ces primitives soient :

- résistantes aux déformations du tracé manuscrit ; c'est-à-dire peu sensibles aux légères variations qui l'entâchent à la saisie,

- représentatives de l'allure générale du tracé pour qu'un nombre limité de primitives puisse suffire à le distinguer,

- faciles à extraire pour qu'on puisse faire la classification en temps réel, ce qui peut avantager ces méthodes par rapport aux autres.

Une première utilisation de ces approches a été faite, dans notre laboratoire, pour l'interprétation de dessins dans le système MIRABELLE (42). Le dessin est entré à la main depuis une tablette graphique et segmenté par la suite en primitives élémentaires de type segment de droite et arc de cercle. Le dessin est ensuite envoyé à un analyseur syntaxique qui, aidé d'une grammaire spécifique au modèle traité, reconstruit le dessin. Les terminaux de la grammaire sont ici les primitives extraites du dessin.

L'emploi de ces primitives pour la classification des caractères manuscrits a nécessité quelques améliorations. Des attributs topographiques de type longueur relative, direction et variation angulaire ont été rajoutés aux codes des primitives.

MIRABELLE est un système général pour des classes très variées de dessin (43). Malgré les améliorations apportées sur les primitives, les performances de ces approches sur la reconnaissance des caractères s'en ressentent (notre système est le contraire d'un système ad hoc !).

CHAPITRE 2

PRESENTATION DU SYSTEME

A la suite des remarques faites précédemment sur l'originalité et l'efficacité des méthodes syntaxiques pour une description en temps réel des formes graphiques, nous avons été amenés à élaborer dans ce sens un système général de reconnaissance structurelle des caractères manuscrits avec application à la reconnaissance des formules mathématiques entrées depuis une tablette graphique.

Deux organes d'analyses, situés à deux niveaux différents, sont à développer dans ce système :

Au niveau du caractère :

Un organe chargé d'isoler le caractère dans la formule et de le reconnaître suivant une technique structurelle basée sur une description très large en primitives. La reconnaissance se fait par comparaison du code du caractère à ceux rangés par apprentissage dans un arbre de décision.

Au niveau de la formule :

Un analyseur syntaxique guidé par une grammaire de description de formules fournie au départ par l'utilisateur, essaie d'engendrer la formule entrée. Les terminaux de la grammaire sont ici les caractères reconnus au niveau inférieur avec en plus de leurs noms des informations supplémentaires telles le taux de reconnaissance calculé sur le caractère et sa position dans la formule.

L'aspect original de ce système se manifeste par l'interaction qui subsiste entre les deux niveaux d'analyse (voir figure 9). En effet, l'automate chargé de la segmentation du caractère en primitives peut fournir plusieurs codes possibles. L'arbre de décision ne retient que ceux qui ont un cheminement possible et renvoie tous les caractères reconnus à l'analyseur de formules. En cas de réponses ambigües, l'analyseur doit sélectionner la réponse qui conviendrait à la reconstruction de la formule.

II.1 - FONCTIONNEMENT GENERAL DU SYSTEME DE RECONNAISSANCE

Très schématiquement, le système de reconnaissance des caractères, étudié ici, peut être représenté de façon très modulaire comme le montre la figure 9. Une interaction subsiste entre les deux derniers étages du système. Elle se fait dans la version actuelle de façon descendante. Il n'est donc pas possible de remettre en cause l'un des modules supérieurs en cas de mauvaise reconnaissance.

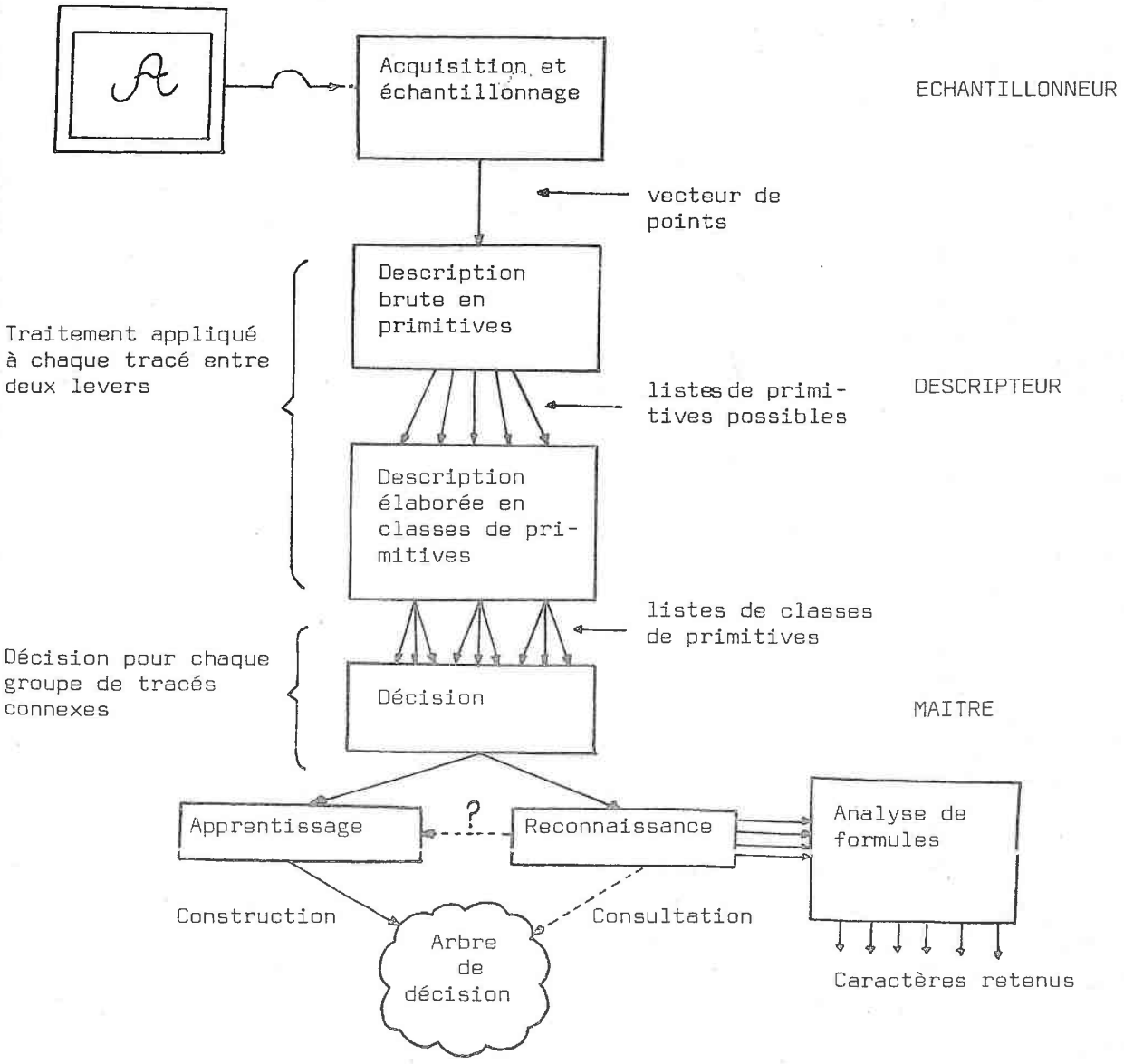


Figure 9 - Schéma du fonctionnement général du système de reconnaissance

II.2 - ACQUISITION ET ECHANTILLONNAGE DU TRACE MANUSCRIT

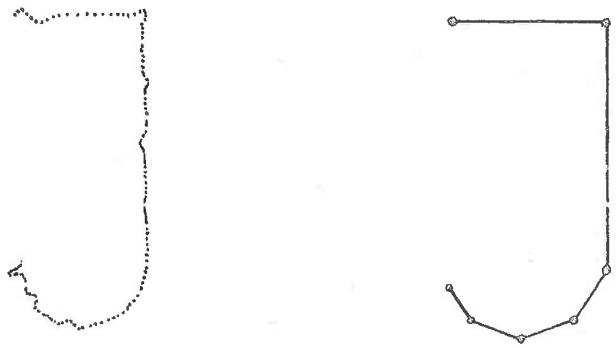
La double fonction de cette phase préliminaire du système est assurée par un module spécial appelé Echantillonneur qui :

- actionne la tablette à la demande de l'écrivain et se charge de l'acquisition propre d'un ou de plusieurs caractères. La chaine de caractères ainsi écrite est représentée par un vecteur de saisie où se trouvent rangés dans l'ordre de leur entrée tous les points et levers de crayon transmis par la tablette graphique. La fin de saisie est prise en compte directement par l'ECHANTILLONNEUR à la rencontre d'un point dont les coordonnées se trouvent dans une zone spéciale de la tablette et qui a été réservée à l'arrêt.

- fait subir à chaque tracé saisi un prétraitement spécial appelé Echantillonnage.

Ce prétraitement consiste à supprimer parmi les points acquis ceux qui ne sont pas significatifs de la forme générale du tracé et corriger les irrégularités (par exemple bavures) qui entâchent le dessin du tracé suite à un lever très rapide du crayon.

Ainsi, le tracé se trouve donné à la sortie de cette étape, sous la forme d'une suite de quelques vecteurs directement assimilables par l'automate de segmentation (voir figure 10).



Tracé brut Tracé échantillonné

Figure 10 - Echantillonnage d'un tracé manuscrit

II.3 - DESCRIPTION STRUCTURELLE DU TRACE

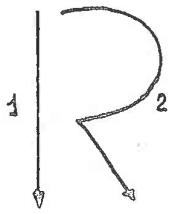
Le descripteur sera chargé, à cette étape de l'analyse, de décrire chaque tracé du caractère entré par une séquence temporelle de primitives et de préciser pour chacune d'entre elles certains attributs topographiques nécessaires à sa classification.

Les primitives qu'on cherche à extraire ici sont semblables à celles qu'on avait décrites précédemment, à savoir :

- Trait (segment de droite) (T)
- Courbe (arc de cercle) (C)
- Lever de plume (L)

Le dessin du caractère R de la figure 11 peut être représenté par la liste suivante :

Trait - Lever - Courbe - Trait - Lever



Les chiffres 1, 2 indiquent sur la figure l'ordre d'entrée des tracés à la saisie et les flèches leur sens d'écriture.

Figure 11 - Code du R :
T + L + C + T + L

Cependant, cette codification par les primitives peut ne pas suffirepour distinguer chacun des caractères de l'alphabet latin ; elle risque au contraire d'introduire de nouvelles ambiguïtés de codage comme dans le cas de la figure 12 qu'il serait difficile d'écarter par la suite

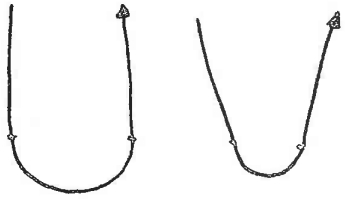


Figure 12 a - Code : T + C + T + L

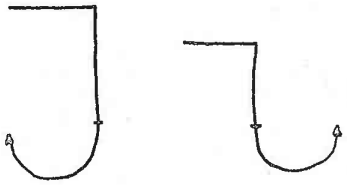


Figure 12 b - Code : T + T + C + L

Il est donc nécessaire d'introduire déjà à ce niveau de nouvelles informations plus discriminantes sur ces primitives pour pouvoir les distinguer dans chaque dessin. Nous remarquons dans l'exemple de la figure 12 b que le sens de la courbe peut en effet suffire pour discriminer les deux caractères.

Dans le cas de la figure 12 a, la direction des traits peut ne pas suffire pour lever l'ambiguïté (identique dans certaines représentations). Des informations de type : variation angulaire de la courbe et la longueur relative par rapport à la longueur totale du tracé seraient par contre plus discriminantes.

Ces attributs topographiques seront introduits par le descripteur, dans une deuxième étape de la segmentation, et ceci pour chaque primitive extraite.

Pour favoriser l'interaction entre les deux niveaux d'analyse : segmentation et reconnaissance , le descripteur essaie de fournir à ce niveau et pour chaque caractère toutes les segmentations possibles qu'on pourrait distinguer à la simple vue de son dessin (voir figure 13). Il ne serait donc pas nécessaire pour le MAITRE de revenir sur la segmentation et la corriger. Toutes les réponses données à ce niveau, par le DESCRIPTEUR, doivent couvrir toutes les réponses possibles nécessaires à la classification du caractère.

Figure 13 - Codes possibles : C + C + L
ou C + T + C + L
ou C + C + T + L
ou C + T + C + T + L

Nous reviendrons, par la suite, sur les détails de la segmentation et nous exposerons le fonctionnement de l'automate d'extraction.

II.4 - APPRENTISSAGE ET RECONNAISSANCE DES CARACTERES

Le caractère est fourni par le module de segmentation sous la forme d'un arbre de codes de primitives possibles.

Ces listes de primitives sont ensuite comparées par le MAITRE à tous les chemins d'un arbre de décision construit par apprentissage. Le chemin de l'arbre qui décrit la même liste de primitives que la liste entrée donne le nom du caractère.

L'arbre de décisions contient au départ, un noyau réduit de caractères qu'on complète par la suite, en cas de mauvaise reconnaissance, au cours du traitement.

L'exploration de l'arbre de décision se fait depuis la racine par comparaison des valeurs des arcs (code d'une primitive)et ainsi de suite jusqu'à la feuille.

La segmentation fine faite à l'apprentissage par la classification des primitives peut ne pas suffire pour discriminer certains caractères comme les ensembles { + , L , T } { X , Y } et { P , D } : Ils présentent le même code de classes de primitives. Des tests de lever d'ambiguïté sont introduits dans l'arbre de décision, au niveau des feuilles, pour séparer les "représentations identiques". Ils distinguent pour chaque caractère les traits distinctifs nécessaires à sa classification. La réponse de ces tests guide ensuite la reconnaissance pour le choix du bon caractère.

Le MAITRE, qui est chargé de l'apprentissage et de la reconnaissance des caractères doit, d'une part, construire l'arbre et introduire les tests nécessaires chaque fois qu'une ambiguïté se présente et procède, d'autre part, à la reconnaissance de chaque caractère entré. Un taux de plausibilité est établi par le MAITRE chaque fois qu'il reconnait un caractère. Ce taux est fonction du taux calculé à la segmentation et d'un taux de reconnaissance calculé pendant le cheminement dans l'arbre de décision.

La séparation des caractères de la formule entrée peut se faire automatiquement dans l'arbre de décision, par comparaison de toute la liste de primitives des caractères confondus (voir figure 14). Nous considérons qu'on a reconnu un caractère chaque fois qu'on arrive à une feuille. On remonte ensuite avec

la liste restante jusqu'à la racine de l'arbre pour une nouvelle comparaison.

La figure 14 donne un exemple des caractères confondus (qui se chevauchent) qui peuvent être isolés automatiquement dans l'arbre de décision (voir figure 15).

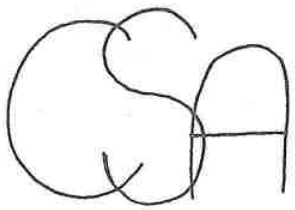


Figure 14 - Caractères confondus

Figure 15 - Caractères isolés

II.5 - APPLICATION A LA RECONNAISSANCE DES FORMULES MATHEMATIQUES

L'application choisie dans le cadre de notre étude consiste à reconnaître en temps réel des formules mathématiques entrées à la main depuis une tablette graphique.

Par leur contenu : caractères alphanumériques, opérateurs et signes mathématiques, et leur structure spéciale : fractions, intégrales ... (voir exemple donné par la figure 16), les formules mathématiques constituent un bon support pour tester l'efficacité de la reconnaissance structurelle sur un jeu important de caractères et élargir d'autre part nos horizons sur les différentes techniques syntaxiques pour l'interprétation des formes complexes.

$$\frac{SIN\ (A + B)}{C - \frac{D}{E}} \times \sqrt{\frac{A}{B}} \qquad \int_{-\infty}^{+\infty} \frac{(C + X)}{X}\, dX$$

Figure 16 - Exemples de formules

Les données de l'analyseur de formules qu'on a écrit sont les différents caractères isolés et reconnus par le niveau inférieur avec des réponses multiples valuées et accompagnés d'attributs relatifs à leur taille et position spatiale dans la formule.

Le rôle de l'analyseur consiste à retrouver parmi les différentes réponses possibles le bon ensemble de caractères qui décrit la formule à partir des règles de la grammaire associée.

L'analyse de la structure de la formule sera dans notre cas une version adaptée d'un algorithme ascendant-descendant, utilisé dans notre laboratoire pour l'interprétation de plusieurs classes de dessins.

Nous montrerons ici comme on l'a adapté aux formules mathématiques pour corriger d'une part les ambiguïtés sur les caractères et décrire, d'autre part, les formules mal écrites présentant certains défauts d'alignement et de cadrage. Une autre adaptation de cet algorithme général a également été réalisée pour la reconnaissance de phrases parlées (MARI).

CHAPITRE 3

ACQUISITION ET ECHANTILLONNAGE DE TRACES

III.1 - MATERIEL D'ACQUISITION

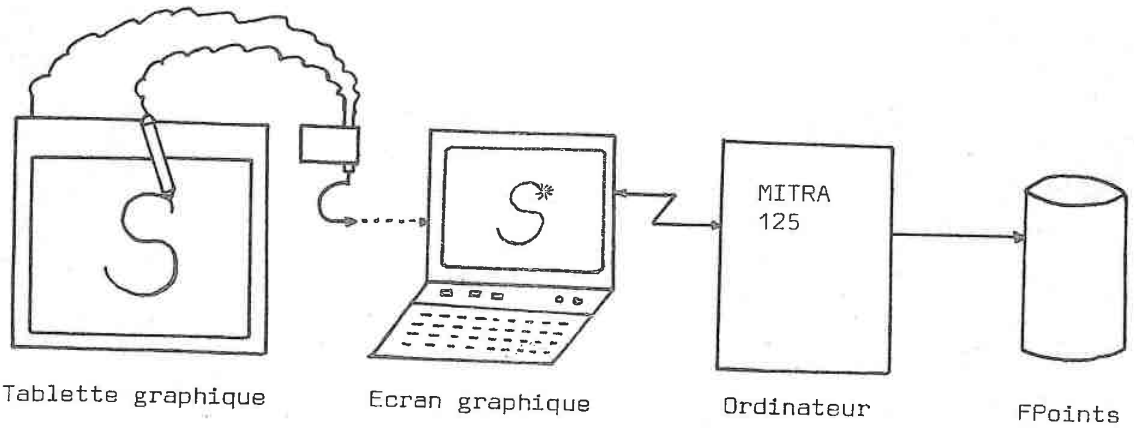


Figure 17 - Matériel d'acquisition

Le matériel d'acquisition et de traitement dont on dispose (voir figure 17) est composé :

- d'une tablette graphique TEKTRONIX 4953 à principe magnétique. C'est une surface magnétique lisse de 1024 x 1024 points, soit environ 28 x 28 cm.

Elle est constituée de fils magnétiques croisés parcourus par un courant électrique qui envoient au contact d'un stylo spécial et à fréquence régulière des signaux bidimensionnels convertibles en coordonnées réelles X et Y relatifs à la position du stylo dans le système d'axes de la tablette.

Un caractère de contrôle accompagne chaque signal ou chaque point. Il indique si le stylo est en position d'écriture ou en position de lever. Ceci nous permet de localiser les divers tracés des caractères entre deux levers de crayon.

- d'un écran graphique TEKTRONIX 4012 sur lequel se reproduit au cours de la saisie le dessin du caractère. Il nous permet de contrôler la saisie et d'afficher certains résultats au cours du traitement.

- d'un ordinateur (MITRA 125) d'une capacité mémoire de 96K mots qui reçoit par l'intermédiaire de lignes asynchrones les points envoyés par la tablette et les range dans l'ordre de leur entrée dans un vecteur de saisie (FPOINTS) afin d'être traités par la suite.

III.2 - ECHANTILLONNAGE DU TRACE MANUSCRIT

L'échantillonnage constitue la phase préliminaire de cette étude. Il se fait en temps réel pendant l'écriture sur chaque tracé saisi, délimité dans le dessin par deux levers successifs du crayon, et ceci de façon totalement indépendante du nom et de la forme générale du caractère écrit.

Le rôle de l'échantillonneur chargé de ce travail consiste à sélectionner parmi les nombreux points du tracé, ceux qui suffisent à le reconstruire. Ceci allège la représentation du tracé et permet de le redécrire de façon beaucoup plus simple par une suite de quelques traits (vecteurs joignant les points successifs) dont la succession dans le plan préserve l'allure et la forme générale du tracé. Ces traits pourront servir d'autre part de base à la représentation structurelle du tracé par les primitives.

Avant de passer à l'exposé de l'algorithme d'échantillonnage, nous allons d'abord voir comment se présente le tracé manuscrit brut à cette phase de prétraitement et comment s'opère ensuite le choix des points caractéristiques dans les diverses régions du tracé.

En effet, le tracé manuscrit se trouve souvent donné par la tablette graphique par un nombre important de points assez mal répartis et pas forcément significatifs de son dessin. L'abondance de ces points est due à la grande vitesse de réception de la tablette (environ 45 points par seconde) si bien qu'un trait vertical de petite taille (∿ 4 cm) et dessiné à une vitesse moyenne peut être donné par environ 30 points (voir figure 18 a) alors que deux points seulement (réduits aux extrémités A et B dans la figure 18 b) suffiraient pour le reconstruire.

Figure 18 a - Trait brut

Figure 18 b - Trait échantillonné

Le rôle de l'échantillonneur consiste dans ce cas à éliminer tous les points intermédiaires et ne sauvegarder que les deux extrémités du trait qui conservent bien l'allure de son tracé (direction verticale) et sa longueur imposée par l'écrivain.

La succession des points se trouve généralement plus abondante dans les régions de forte courbure (voir figure 19 a) à cause d'un ralentissement de la vitesse d'écriture. Il faut donc échantillonner dans ces régions plus de points pour conserver l'allure et la variation angulaire du tracé.

Le nombre de points sélectionnés sera ici fonction de l'ouverture de la courbe et de sa dimension par rapport à tout le tracé. Ces points doivent être en plus bien répartis pour qu'on puisse décrire la courbe comme une suite de quelques traits de longueurs sensiblement égales et formant entre eux des angles relativement égaux.

Nous voyons que dans le cas des deux figures ci-dessous que plus de points ont été retenus pour la courbe du U que pour celle du V.

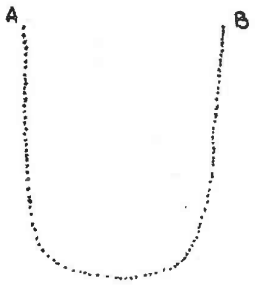


Figure 19 a - Tracé brut

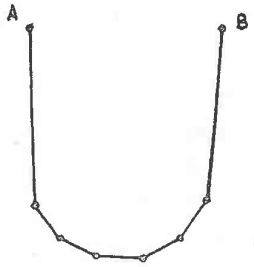


Figure 19 b - Tracé échantillonné

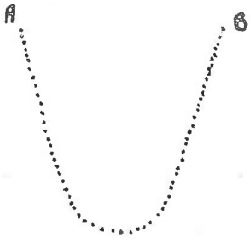
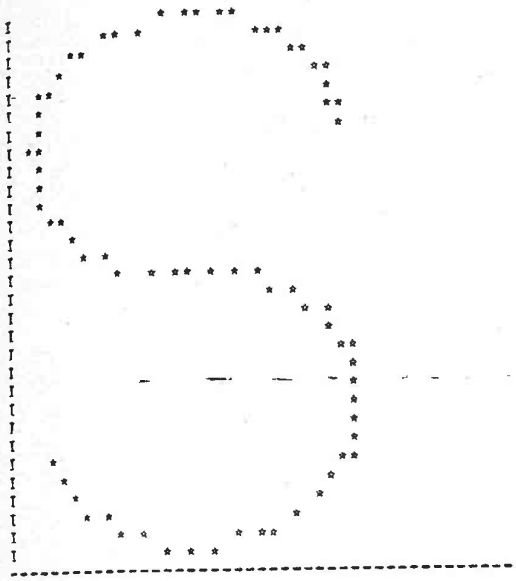


Figure 20 a - Tracé brut

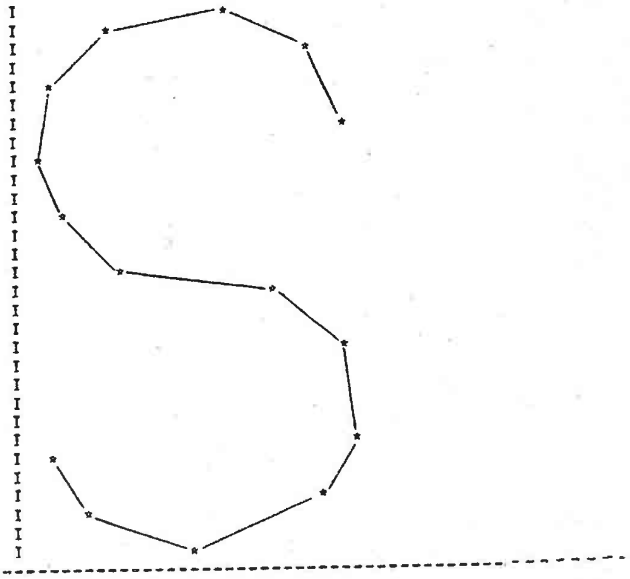
Figure 20 b - Tracé échantillonné

TRACE DU CARACTERE : S

Tracé brut

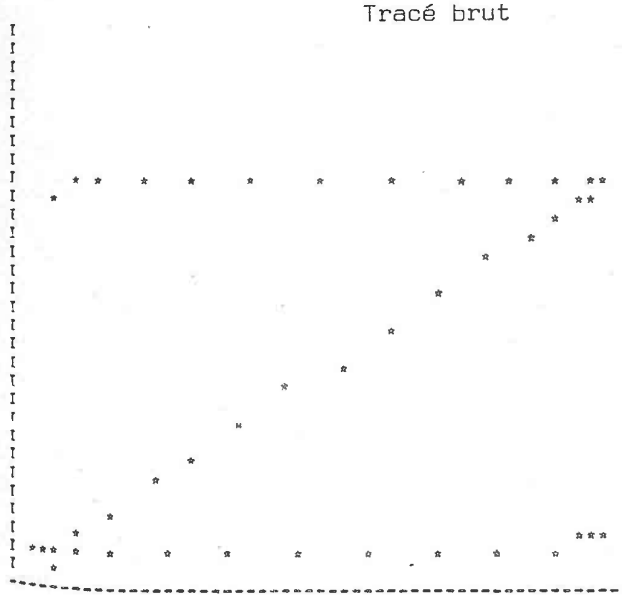


Tracé échantillonné

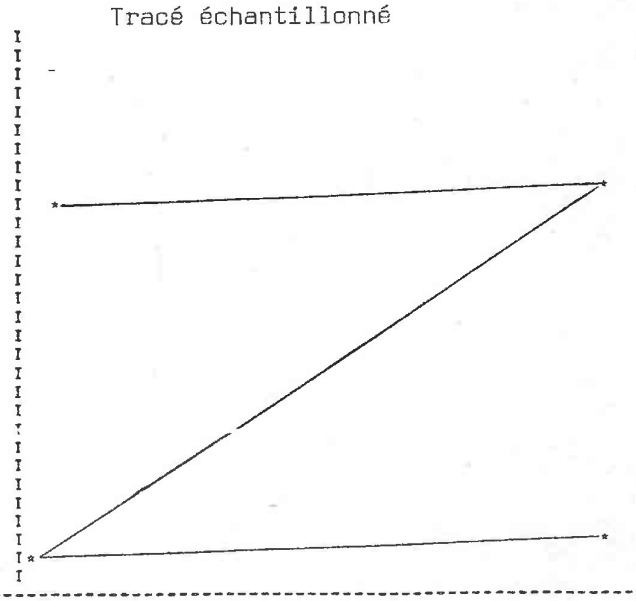


TRACE DU CARACTERE : Z

Tracé brut



Tracé échantillonné



Exemple d'échantillonnage réalisé sur deux tracés différents :
- tracé courbé (S)
- tracé en zigzag (Z)

Technique d'échantillonnage :

D'après les remarques faites précédemment, nous pouvons dire que le procédé d'échantillonnage consiste à ne retenir dans le tracé que les sommets d'une ligne polygonale ne s'éloignant que légèrement du tracé. Reste donc à trouver la bonne technique pour localiser ces points dans le tracé.

D'autre part, il est assez difficile de dessiner à la main un tracé parfait (bien arrondi dans les régions courbées) ; des déformations (voir figure 21 a) et des bavures (voir figure 21 b) viennent entâcher le tracé et rendent cette recherche difficile. Il faudrait une technique d'échantillonnage assez élaborée qui serait peu sensible aux légères variations du tracé.

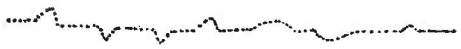


Figure 21 a - Déformations

Figure 21 b - Bavures

ITO et CHOI (31) ont proposé dernièrement une technique d'échantillonnage assez robuste qui consiste, en partant du premier point du tracé, à chercher le point suivant qui s'éloigne le premier d'une région de tolérance délimitée par la tangente T avec une ouverture de ± θ (θ angle d'échantillonnage donné comme paramètre). Le choix du point se fait par test sur la position de chaque point par des approximations de Chebyschev. Cette technique est ensuite réitérée sur le reste du tracé en partant à chaque itération du dernier point échantillonné.

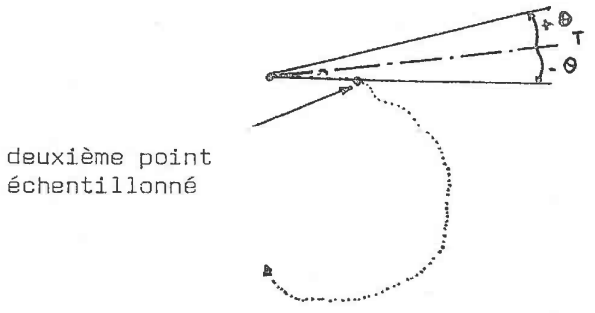


Figure 22 - Echantillonnage par une méthode d'approximation

Cette technique a été écartée dans notre cas à cause du calcul prohibitif des tests. Nous avons utilisé une technique beaucoup plus simple et très rapide inspirée des diverses techniques d'échantillonnage angulaire mises au point par BERTHOD (27) pour la même application.

Son principe est le suivant :
Examiner l'angle formé par deux traits successifs dans le tracé et supprimer leur point de jonction si la valeur de cet angle se trouve inférieure à un seuil d'angle SANGLE déterminé au départ par l'écrivain.

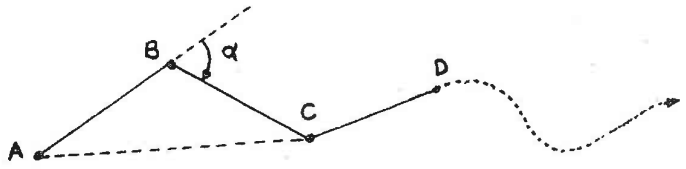


Figure 23 - Echantillonnage angulaire

(nous avons choisi, dans notre cas, SANGLE = 20°)

Le point B serait supprimé dans le cas de la figure 23, si α était inférieur à SANGLE et le processus d'échantillonnage serait itéré sur A, C et le point suivant et ainsi de suite jusqu'à la fin du tracé.

Cet échantillonnage angulaire est insuffisant pour corriger les déformations du type indiqué dans la figure 24 (pics introduits imprudemment suite à un tremblement de la main) où l'angle entre les traits peut dépasser dans ce cas le seuil d'échantillonnage.

Pour consolider cet échantillonnage, nous avons introduit des tests portant sur la longueur relative de chaque trait. Le point de jonction serait supprimé si l'une au moins de ces deux longueurs était inférieure à un seuil de longueur SLONG calculé en fonction de la longueur totale du tracé (nous avons choisi SLONG = $\frac{1}{20}$ de la longueur totale).

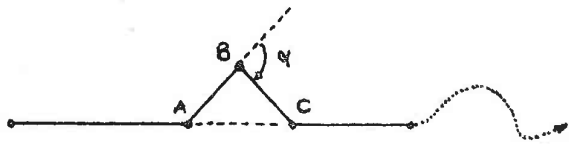


Figure 24 - Echantillonnage de pics

(B serait supprimé si |AB| était inférieur à SLONG).

D'autre part, il faut supprimer les bavures introduites en fin de tracé suite à un lever trop rapide du crayon. Ces bavures se traduisent le plus souvent par un rebroussement en fin de tracé donnant lieu à un trait supplémentaire de taille plus ou moins grande suivant la vitesse à laquelle a été levé le crayon (voir figure 25).

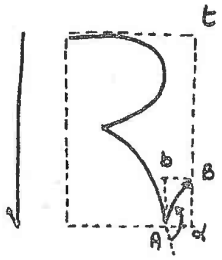


Figure 25 - Bavures

Le traitement des bavures se fait dans la procédure en fin d'échantillonnage en comparant la taille du dernier trait à l'enveloppe rectangulaire de tout le tracé qui le contient.

AB serait supprimé du tracé du R dans la figure 25 si $b \leqslant p\ t$

b = demi-périmètre de la bavure

t = demi-périmètre du tracé

p = coefficient de comparaison (appelé PBAV dans la procédure)

III.3 - QUELQUES EXEMPLES D'ECHANTILLONNAGE POUR DIFFERENTS PARAMETRES

Nous avons fait varier les valeurs des trois paramètres d'échantillonnage : SANGLE , SLONG et PBAV et nous les avons essayés sur tout le jeu de caractères dont on dispose (caractères latins majuscules, minuscules, chiffres arabes et quelques signes mathématiques). Les figures qui vont suivre donnent quelques résultats obtenus pour une écriture normale sur des tracés de taille suffisamment grande (2 cm au minimum) pour qu'on puisse voir les différents traits composants.

Les valeurs finalement retenues sont :

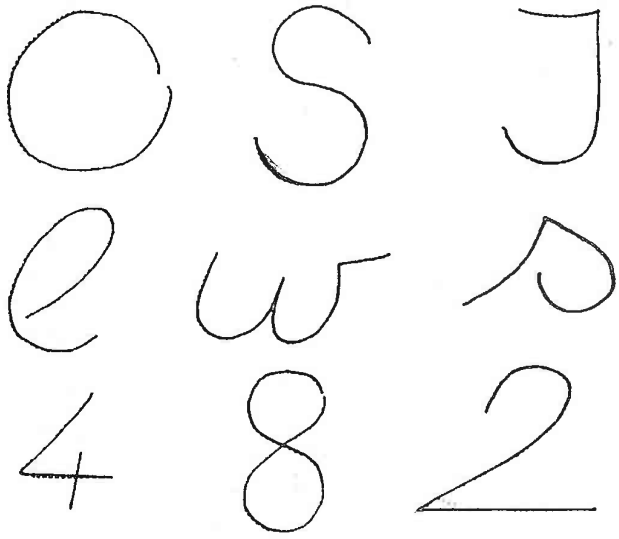
$$\text{SLONG} = \frac{1}{20} \text{ de la longueur totale du tracé}$$

$$\text{SANGLE} = 20°$$

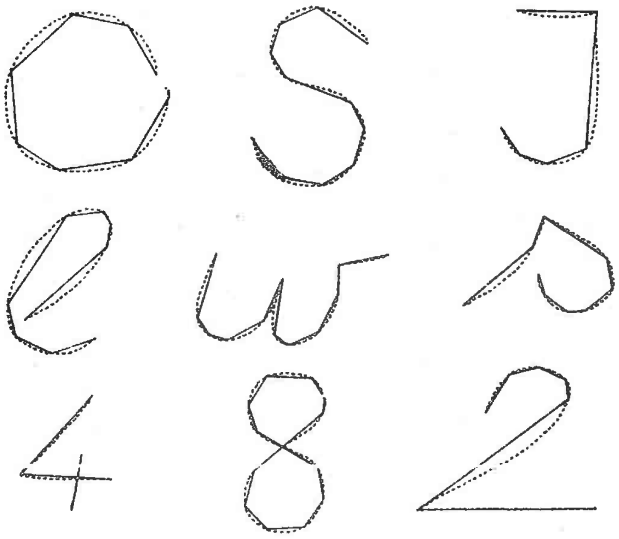
$$\text{PBAV} = \frac{1}{15} \text{ de la longueur totale du tracé}$$

On trouvera dans BERTHOD (26) des résultats plus complets que l'on obtient avec plusieurs paramètres.

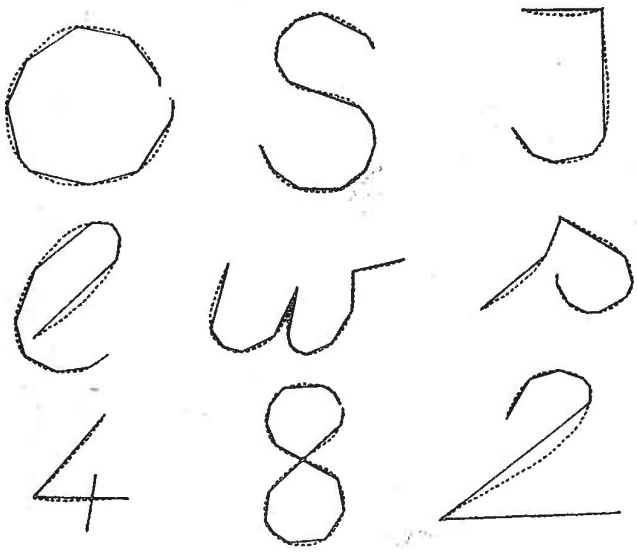
Dans les figures qui vont suivre, les tracés bruts sont en traits pointillés et les tracés échantillonnés sont en traits pleins.



SANGLE = 5°
SLONG = 0,025
SBAV = 0,01

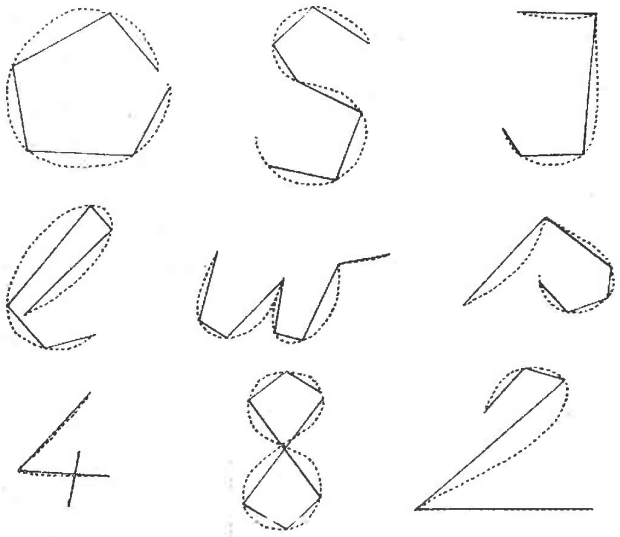


SANGLE = 15°
SLONG = 0,05
SBAV = 0,02



SANGLE = 20°
SLONG = 0.05
SBAV = 0.02

Paramètres retenus



SANGLE = 40°
SLONG = 0.2
SBAV = 0.01

III.4 - CONCLUSION

L'algorithme, mis au point dans ce chapitre, nous assure un échantillonnage cohérent pour une variété de tracés manuscrits. Les paramètres sélectionnés ont été testés sur tout le jeu de caractères dont on dispose et semblent répondre parfaitement à l'échantillonnage désiré. Notre technique d'échantillonnage reste néanmoins limitée à des caractères de taille importante. Son application à des petits caractères ne donne pas des résultats satisfaisants. Ceci vient du fait qu'on a cherché à mettre au point un algorithme général qui fonctionnerait aussi bien pour les caractères que pour les dessins (généralement de taille importante).

BERTHOD (27), qui a travaillé sur les petits caractères, procède lui à un lissage préliminaire du tracé en remplaçant chaque point par une combinaison linéaire de ce point et de ses deux voisins, ce qui lui assure une homogénéité dans les longueurs des traits, et par conséquent, un échantillonnage beaucoup plus fidèle.

CHAPITRE 4

DESCRIPTION STRUCTURELLE DE TRACES

IV.1 - DEFINITION DES PRIMITIVES

Les primitives qu'on cherche à extraire du tracé manuscrit sont des sous-formes géométriques naturelles qu'on peut distinguer visuellement par simple examen de la forme du dessin. Elles sont assez semblables à celles utilisées par d'autres chercheurs et se répartissent en deux grandes classes différentes :

- segment de droite ou trait (T)
- Arc de cercle ou courbe (C)

Les levers de crayon et la fin du caractère ne seront pas considérés ici comme primitives car ils n'interviennent pas directement dans la représentation structurelle du tracé. Nous en tenons compte dans la représentation finale du caractère pour délimiter d'une part les divers tracés dans les différentes listes assorties et pour marquer d'autre part la fin du caractère dans la chaine d'entrée. Nous donnerons un bref aperçu, en fin de paragraphe, sur leur mode de détection et leur utilité ; mais nous allons d'abord donner une définition exacte des deux classes de primitives citées ci-dessus à partir des échantillons sélectionnés (points) au niveau inférieur.

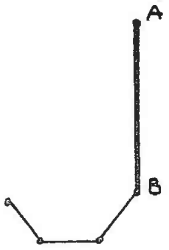
a) Segment de droite

Le segment de droite garde ici sa définition mathématique (géométrique). Il sera donné par la succession de deux points dans le tracé.

On le distingue, dans le tracé, de deux manières différentes :

- soit directement si le tracé est réduit à deux points (cas du I donné par la figure 26 a),
- soit indirectement dans une région aplatie du tracé comme le montrent les deux cas de figure 26 b. On assimile à vue d'oeil la portion AB à un segment de droite parce qu'elle se distingue dans le tracé par sa faible courbure et par sa taille.

Figure 26 a - Trait



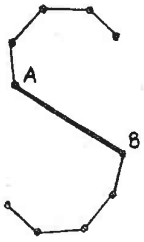


Figure 26 b - Régions de faibles courbures

Il arrive parfois de représenter le trait par une suite de trois ou quatre points comme le montre la figure 27. Des bavures de petites tailles peuvent être introduites aux bouts du trait suite à un mauvais départ ou à une déconcentration de l'écrivain à la fin du tracé. Nous assimilons dans ce cas le tracé à un trait d'origine : l'origine du premier trait et d'extrémité : l'extrémité du dernier trait :

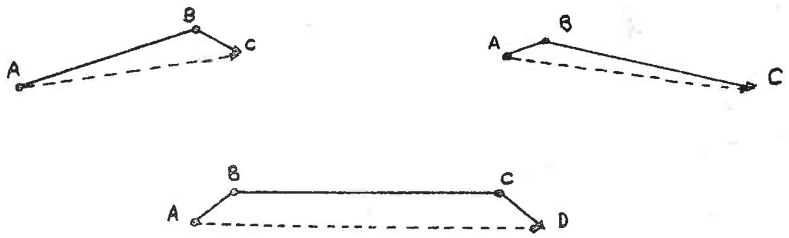


Figure 27 - "Segments de droite" de plus de deux points

b) Arc de cercle

Il est assez difficile de donner une définition précise de l'arc de cercle à partir de la représentation vectorielle faite à l'échantillonnage. Le nombre de points et leur disposition dans les régions courbées du tracé changent suivant le caractère et le mode d'écriture. Il est impossible de dessiner le même caractère deux fois de suite et d'avoir par conséquent deux représentations identiques.

En effet, selon que l'on écrit lentement ou rapidement, on obtient un nombre plus ou moins important de points. D'autre part, la vitesse d'écriture ne reste jamais

constante pendant la durée du tracé, ce qui peut influencer beaucoup sur la bonne répartition des points et leur nombre dans le dessin. Pour une même courbe (voir figure 28), on peut avoir une infinité de représentations vectorielles différentes.

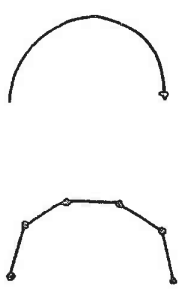


Figure 28 a - Courbe idéale

Figure 28 b - Différentes représentations obtenues

Il faudrait donc une définition un peu plus souple de la courbe qui tolère au maximum tous les types d'irrégularité que peut subir le tracé au cours de l'écriture, mais assez cohérente pour qu'elle soit significative d'un arc de cercle.

On définit une courbe, d'après les remarques faites précédemment, comme une suite d'au moins deux traits de tailles à peu près égales et formant entre eux des angles de même signe et de valeurs sensiblement égales.

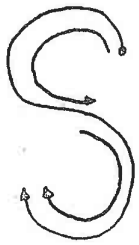
L'égalité entre les longueurs des traits et les angles sera faite avec une large marge de tolérance. Nous reviendrons par la suite sur les seuils de comparaison. Nous allons étudier à présent les trois notions importantes qui interviennent dans la définition de la courbe.

Angle de même signe

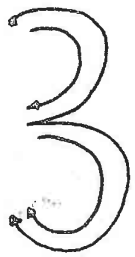
Le signe de l'angle est une information très importante sur la courbe. Elle nous permet d'extraire deux classes de courbes différentes qu'on peut distinguer à la simple vue du tracé, telles :

- courbe à courbure positive (directe) : C +
- Courbe à courbure négative (rétrograde) : C -

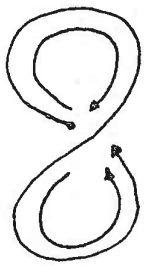
Avec le sens de la courbe, nous pouvons séparer plusieurs caractères comme le montre la figure 29 :



Courbe positive
+ courbe négative



Courbe négative
+ courbe négative



Courbe négative
+ courbe positive

Le changement de courbure peut être déterminé facilement par calcul des angles entre les vecteurs successifs dans le tracé et comparaison de leur signe.

Valeur des angles :

La valeur des angles des traits composants la courbe doit être petite (inférieure à un certain seuil) pour qu'on puisse distinguer les coins anguleux, les traits et les courbes dans le tracé. Le carré de la figure 30 est une succession de 4 traits. Sa codification par une courbe ne correspond pas à la réalité d'une personne qui écrit.

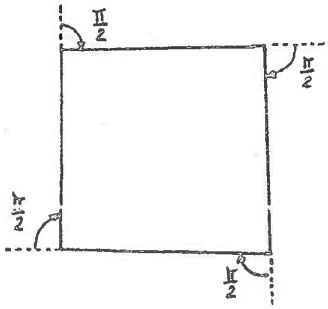


Figure 30 : Trait + Trait + Trait + Trait

Ce cas peut encore se poser sérieusement pour les tracés formés d'une suite de deux traits où il est assez difficile de décider comme dans le cas de la figure 30bis jusqu'à quelle limite d'angle on peut encore considérer que deux traits consécutifs forment une courbe ?

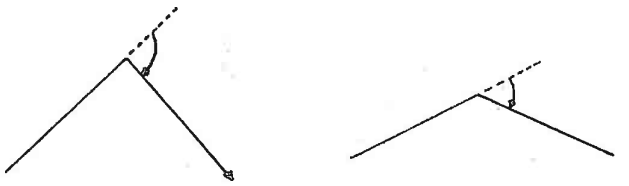


Figure 30 bis

Trait + Trait Courbe

Notre expérimentation (comparaison à vue d'oeil) nous a conduit à un seuil de $\frac{\pi}{3}$

Taille des traits :

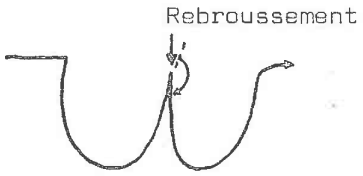
La taille des traits doit être sensiblement égale pour qu'on puisse séparer dans le même tracé le trait de la courbe.

Figure 31

Trait + Courbe Courbe + Trait

Les points de la courbe comme le montrent les deux cas de la figure 31 sont beaucoup plus rapprochés que pour ceux du trait.

Nous trouvons dans la littérature une autre classe de primitive appelée : Rebroussement. Elle désigne l'emplacement d'un point anguleux du tracé de valeur proche de 180°. Nous ne l'utiliserons dans notre système que pour découper le tracé en zones homogènes distinctes.



Lever de crayon :

Le lever de crayon est une information qu'on introduit dans la représentation finale du caractère pour délimiter les différents tracés composants. Il ne nécessite aucun calcul supplémentaire puisqu'il est donné par la tablette graphique chaque fois qu'on lève le stylo du plan d'écriture.

Fin de caractère :

Elle peut être donnée de deux manières différentes suivant qu'on est en mode d'apprentissage ou en mode de reconnaissance.

- A l'apprentissage, l'écrivain trace un caractère sur la tablette et pointe le stylo en fin de saisie dans une case spécifique correspondant au caractère entré (voir figure 32). Ceci arrête la segmentation des différents tracés du caractère.

- A la reconnaissance, la fin du caractère est observée directement dans l'arbre de décision , en trouvant une feuille correspondant au nom du caractère entré.

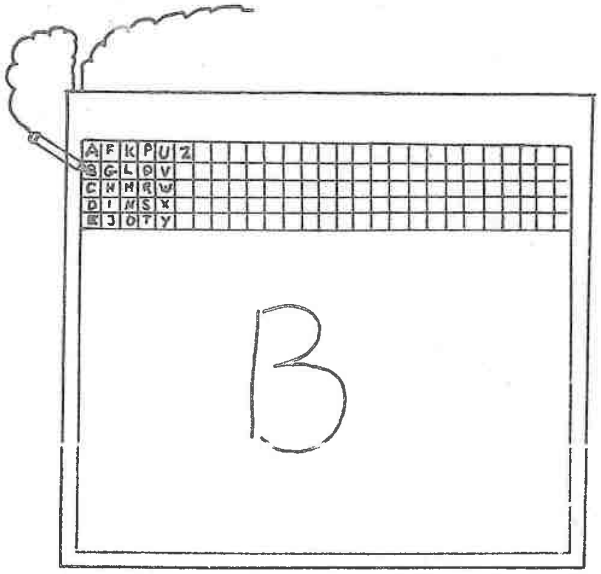


Figure 32 - Fin du caractère à la saisie

IV.2 - AUTOMATE D'EXTRACTION DES PRIMITIVES

L'automate que nous allons décrire est à réponses multiples et valuées. Il est chargé de localiser les deux classes de primitives (courbe - Trait comme définies précédemment) dans le tracé manuscrit et de le présenter au module supérieur comme une ou plusieurs juxtapositions possibles de primitives. Il oeuvre dans des zones homogènes du tracé sélectionnées au départ par un module spécial (zone à 1 trait, zone à deux traits ...) au sens de quelques paramétrages faits au préalable et adaptés de façon particulière à chacune de ces zones (45).

La segmentation ne se fait pas comme dans les travaux de BERTHOD de façon séquentielle par "suivi de courbe", mais "par morceaux" avec segmentation globale de chaque zone particulière du tracé.

En effet le problème fondamental auquel se heurtent tous les chercheurs dans la segmentation structurelle est la séparation dans le tracé des deux classes de primitives Trait et Courbe. Cela provient essentiellement de la définition de la courbe qui est trop difficile à saisir à cause des irrégularités incessantes de la forme du tracé. Les mesures d'angle et de taille de traits ne suffisent pas toujours à la localiser dans tous les cas de figure. Une façon normale de procéder est de localiser d'abord la portion du tracé qui serait susceptible de contenir une telle courbe et de dégager ensuite et en bloc les listes de primitives qui seraient capables de la représenter globalement.

La figure 33 donne un exemple de séparation en zones homogènes et de segmentations multiples réalisées dans ces zones.

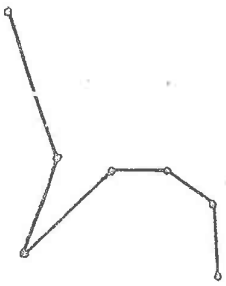


Figure 33 a - tracé entré

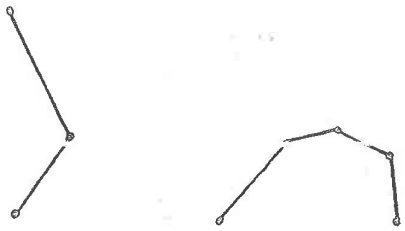


Figure 33 b - Décomposition du tracé en 2 zones et segmentation dans ces zones

1ère zone	2ème zone
Codes : T(2) + T(2)	Codes : T(2) + C(4)
ou C(3)	ou T(2) + C(3) + T(2)
	ou C(5)

T(n) : signifie trait à n points

C(n) : signifie courbe à n points

Une des chaines de segmentation possible de tout le tracé serait :

C(3) + T(2) + C(3) + T(2)

IV.2.1 - Découpage du tracé en zones homogènes

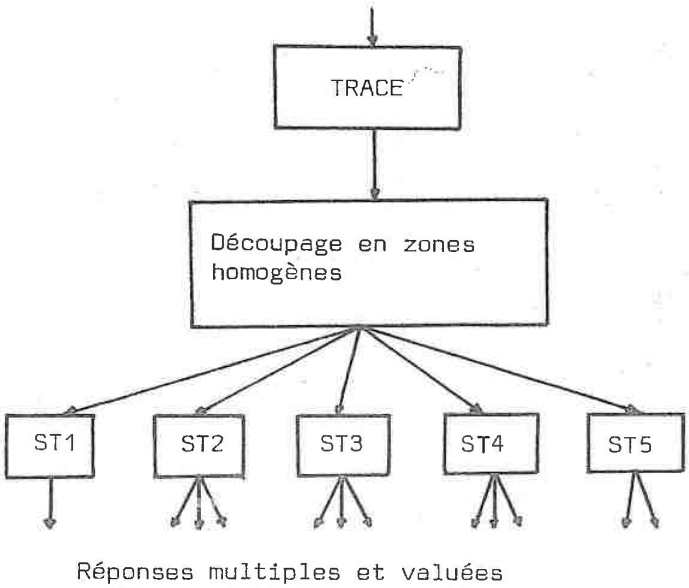


Figure 34 - Schéma général de l'automate d'extraction

L'automate d'extraction est composé d'un module principal qui procède à une segmentation angulaire du tracé pour délimiter les diverses zones particulières du tracé qu'il renvoie par la suite à des modules spécifiques de segmentation en primitives tels ST1 , ST2 , ST3 , ST4 et ST5 (voir figure 34). Cette segmentation angulaire se fait par examen des valeurs des angles formés par les traits consécutifs dans le tracé.

Le renvoi à ces modules de segmentation se fait de trois manières différentes :

a) à la rencontre d'un angle supérieur à un seuil de coin (SCOIN = 70°). En effet, au-delà de ce seuil, nous concevons que la portion du tracé n'est pas suscepti-

ble de représenter une courbe (d'après la définition de la courbe donnée dans le paragraphe précédent).

b) à la détection d'un changement de courbure(angle de signe contraire à celui qui le précède dans le tracé).

c) à la rencontre d'un cinquième point faisant donc partie d'une suite de quatre traits formant entre eux des angles de même signe et inférieurs en valeur absolue à un seuil de coin. Ceci répond parfaitement à la définition proposée de la courbe. Il est donc nécessaire d'aller examiner à ce niveau la structure de la zone détectée. La zone est ainsi renvoyée au module ST4 (sous-tracé à 4 traits) qui ressort toutes les listes de primitives qui composent la zone à l'aide de paramètres de segmentation donnés au départ par l'utilisateur et utilisant des mesures simples d'angles et de longueurs de traits.

Si le sous-module ST4 répond par une courbe, nous continuons à chercher la suite de la courbe dans le module supérieur ST5 par analyse successive des traits suivants (ceci revient à lire le trait suivant et voir s'il forme avec la courbe détectée une nouvelle courbe). Deux réponses sont donc possibles dans le module : Courbe ou Courbe + Trait. Dans le cas où on obtient une courbe, on réitère le même procédé de comparaison sur le trait suivant jusqu'à ce qu'on trouve la suite Courbe + Trait ou jusqu'à ce que le module principal détecte un changement de Courbure ou une fin de tracé.

Si le module ST4 répond par autre chose que courbe (exemple : Trait + Courbe), alors on retient la première primitive de cette sous-liste et on renvoie le reste de la zone au module correspondant.

Le fonctionnement détaillé de l'automate sera explicité plus loin, dans ce chapitre.

IV.2.2 - Segmentation dans les zones

Les zones homogènes du tracé sont répertoriées par l'automate d'extraction en fonction du nombre de traits qu'elles contiennent et sont renvoyées par la suite aux modules de segmentation correspondants.

Ces modules sont munis chacun d'une pile de tests pertinents déterminés au préalable par l'utilisateur et ajustés chacun pour extraire une liste particulière de primitives.

Ces tests sont appliqués séquentiellement sur la zone entrée et celui qui aboutit à un succès donne la liste de primitives correspondante.

Les listesde primitives à extraire de chaque zone sont connues à l'avance. Elles sont assez représentatives de leurs formes et faciles à extraire à l'aide de mesures d'angle et de longueur établies sur les traits composants.

Nous les classons par zone comme suit :

Zone à 1 trait

Liste :

- Trait

Zone à 2 traits

listes :

- Trait
- Courbe
- Trait + Trait

Zone à 3 traits

listes :

- Courbe
- Courbe + Trait
- Trait + Courbe
- Trait + Trait + Trait

Zone à 4 traits

Listes :

- Courbe
- Trait + Courbe
- Courbe + Trait
- Trait + Courbe + Trait

  Courbe + Courbe

- Trait + Trait + Courbe
- Courbe + Trait + Trait

Zone à 5 traits

L'accès à ce module se fait au cas où on obtient à l'analyse de la zone à 4 traits une courbe. Les tests introduits ici vérifient si le trait suivant dans le tracé forme une courbe avec la précédente.

Listes :

- Courbe
- Courbe + Trait

Les tableaux de la figure 38 donnent les schémas correspondants aux listes de primitives détectées dans ces zones.

Ces listes de primitives ont été cherchées à la main par étude de différentes variations de chaque zone. Les tests qu'on a introduits pour les extraire sont assez simples et s'appliquent sur la zone concernée de façon totalement indépendante du reste du tracé.

Nous allons voir comment s'appliquent ces tests sur une zone de 3 traits (voir figure 35).

- Courbe : C(4)
  si $l_1 \simeq l_2 \simeq l_3$
  $\alpha \simeq \beta$
  et α , $\beta \leqslant$ SANG

- Courbe + Trait : C(3) + T(2)
  si $l_1 \simeq l_2$
  $\alpha \leqslant$ SANG
  et $l_3 >> (l_1 + l_2)/2$

- Trait + Courbe : T(2) + C(3)
  si $l_2 \simeq l_3$
  $\beta \leqslant$ SANG
  et $l_1 >> (l_2 + l_3)/2$

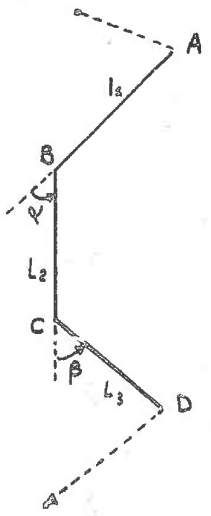


Figure 35 - Zone à 3 traits

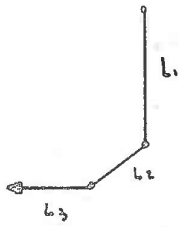
- Trait + Trait + Trait : T(2) + T(2) + T(2)
  si les tests ci-dessus sont mis en échec.

- les l_1 , l_2 , l_3 sont les longueurs absolues des traits composants AB, BC et CD

- α , β sont les angles entre les couples de traits consécutifs dans la zone (AB, BC) et (BC, CD)

- SANG est le seuil maximum d'angle que ne doit pas dépasser l'angle de deux traits composant la courbe.

Comme nous opérons sur des tracés déformés, les comparaisons entre les angles et les longueurs seront faites dans les tests avec une large marge de tolérance car il est presque impossible de trouver deux traits de même taille (de même pour les angles). D'autre part, et de peur que la segmentation ne soit trop restrictive à ce niveau, nous avons élargi les marges de tolérance de sorte que plusieurs tests soient vérifiés en même temps et par conséquent, plusieurs segmentations soient données pour la même zone (voir fig. 36).



trait + courbe
ou courbe

Figure 36 - Deux segmentations possibles

Cette segmentation multiple est très utile dans notre cas. Elle renforce l'interaction entre le module d'extraction et le module de reconnaissance en lui proposant les différentes segmentations possibles du caractère. Le choix est donc plus large, ce qui permet une reconnaissance beaucoup plus sure.

Le module de reconnaissance ne retient parmi les listes de primitives que celles qui ont un cheminement possible dans l'arbre de décision.

Nous pensons que ce procédé de segmentation est assez efficace. Son extension à la segmentation d'autres formes graphiques ne pose à priori aucune difficulté. Il suffit d'introduire dans l'automate les listes de primitives à extraire et

les tests adéquats. D'autre part, les seuils de comparaison (longueur et angle) sont tous paramétrés. Il suffit de les ajuster en fonction des listes des primitives à ressortir et de leur nombre.

IV.2.3 - Représentation structurée du tracé en primitives

Suite à la segmentation multiple opérée dans ses différentes zones, le tracé manuscrit sera donné à la sortie de l'automate d'extraction par un arbre dont chaque chemin représentera une chaine de primitives (voir figure 37). Le noeud de l'arbre a le format suivant :

Type	INFO	LFR

où :

Type : est un code précisant la nature de la primitive. On distingue trois types différents :

- Trait
- Courbe positive
- Courbe négative

INFO : est un pointeur dans un tableau où seront rangées diverses informations nécessaires à la classification de la primitive dans une étape supérieure, telles les coordonnées de ses extrémités, la longueur absolue et la variation angulaire s'il s'agit d'une courbe.

LFR : est le lien horizontal avec la primitive "frère"
(autres primitives possibles en ce noeud).

Exemples

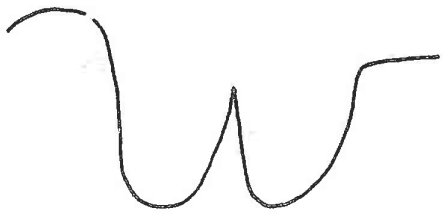


Figure 37 a - Tracé entré

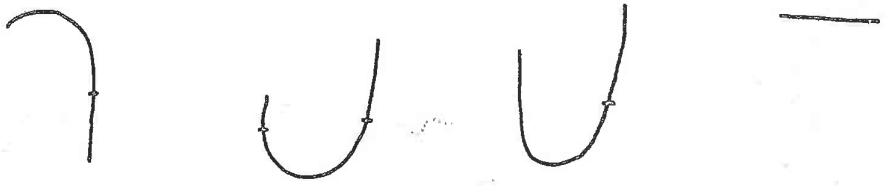


Figure 37 b - différentes zones homogènes du tracé

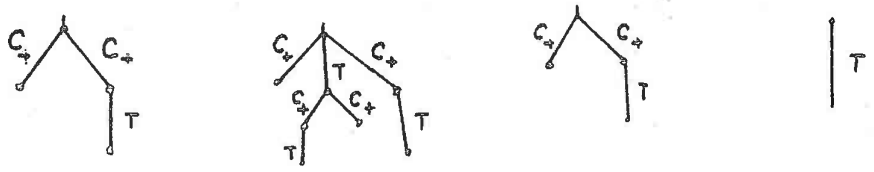


Figure 37 c - Segmentations multiples dans les zones

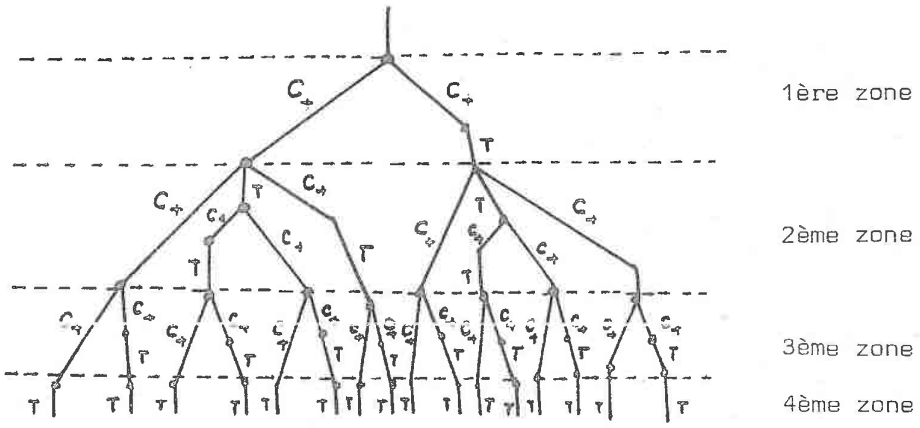


Figure 37 d - Représentation structurée du tracé en plusieurs listes de primitives

IV.2.4 - Calcul du score de segmentation

Un score de segmentation est calculé pour chaque liste de primitives extraites du tracé. Ce score accompagne chaque chemin de l'arbre du tracé et indique au module de reconnaissance le taux de plausibilité (0 à 100) de chacune de ces segmentations possibles. En imposant un seuil assez élevé pour le score, nous pouvons limiter le nombre de listes trouvées. Le module de reconnaissance ne prend en compte que les listes de primitives qui ont un score de segmentation élevé. Ce score est par ailleurs très utile à l'apprentissage. Il peut fixer nos idées sur le bon choix des tests de segmentation et surtout sur la bonne représentativité de ces listes.

Le score de segmentation final (Ss) qu'on calcule pour chaque chemin est une moyenne arithmétique de tous les scores (Si) calculés localement dans les diverses zones du tracé.

$$Ss = \frac{\sum_{i=1}^{n} Si}{n}$$

n est le nombre de zones découpées du tracé.

Les Si sont eux-mêmes des moyennes des différents scores calculés lors des diverses mesures faites dans chaque test.

$$Si = \frac{\sum_{j=1}^{P} Sij}{P}$$

P est le nombre de ces mesures. P varie d'un test à l'autre.

Reprenons le test de la figure 35 qui est donné par l'automate pour voir si la zone de trois traits forme une courbe (le score est établi chaque fois qu'il y a succès).

Courbe : C(4)

Si 1/ $l_1 \simeq l_2 \simeq l_3$ → Si_1

2/ $\alpha \simeq \beta$ → Si_2

3/ $\alpha \leqslant$ SANG → Si_3

4/ $\beta \leqslant$ SANG → Si_4

Récrivons ces mesures de façon plus formelle :

1/ (MAX (l_1 , l_2 , l_3) - MIN (l_1 , l_2 , l_3)) $\leq$ SLON

2/ $| \alpha - \beta | \leq$ SANG1

3/ $\alpha \leq$ SANG

4/ $\beta \leq$ SANG

ce qui donne :

$$Si_1 = \left(1 - \frac{MAX (l_1 , l_2 , l_3) - MIN (l_1 , l_2 , l_3)}{SLON} \right) * 100$$

$$Si_2 = \left(1 - \frac{| \alpha - \beta |}{SANG1} \right) * 100$$

$$Si_3 = \frac{\alpha}{SANG} * 100$$

$$Si_4 = \frac{\beta}{SANG} * 100$$

$$\rightarrow \quad Si = \frac{Si_1 + Si_2 + Si_3 + Si_4}{4}$$

SLON : seuil de longueur

SANG1 : seuil d'angle

Les scores reflètent bien l'exactitude de l'extraction des différentes primitives dans le tracé puisqu'ils sont calculés directement dans les tests de comparaison.

	Sous-tracé à 1 trait	Primitives
ST1		T(2)

	Sous-tracé à 2 traits	Primitives
ST2		C(3)
		T(2) + T(2)
		T(3)

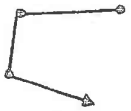
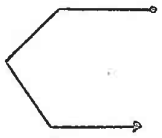
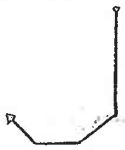
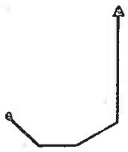
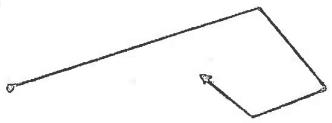
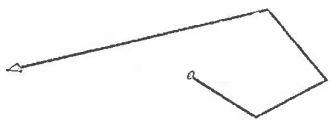
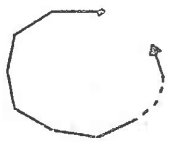
	Sous-tracé à 3 traits	Primitives
ST3		C(4)
		T(2) + C(3)
		C(3) + T(2)
		T(2) + T(2) + T(2)

Figure 38 - Modules d'extraction

./...

	Sous-tracé à 4 traits	Primitives
ST4		C(5)
		C(3) + C(3)
		T(2) + C(4)
		C(4) + T(2)
		T(2) + C(3) + T(2)
		T(2) + T(2) + C(3)
		C(3) + T(2) + T(2)

	Sous-tracé à n points	Primitives
ST5		C(n)
		C(n - 1) + T(2)

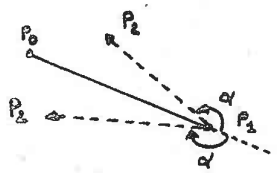
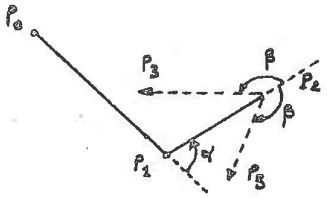
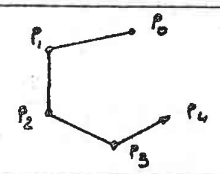
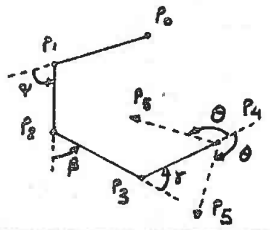
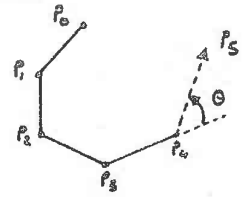
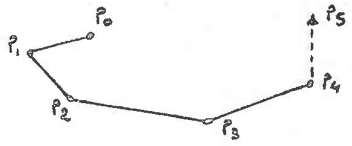
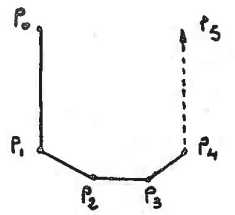
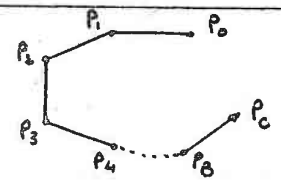
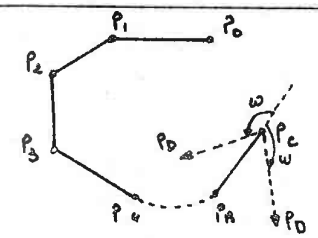
Etape	Fonction	Renvoi à l'étape	Schéma
1	Lire un point . Si lever .... STOP	ST1	
1	. Si Point : P2 $\alpha = (\vec{P0P1}, \vec{P1P2})$. Si $\lvert\alpha\rvert$ > SCOIN .... P0 ← P1 ; P1 ← P2	ST1 1	
1	. Si α < 0	2-	
1	. Si α > 0	2^+	
2^+	Lire un point . Si lever .... STOP	$ST2^+$	
2^+	. Si Point : P3 $\beta = (\vec{P1P2}, \vec{P2P3})$. Si $\lvert\beta\rvert$ > SCOIN .... P0 ← P2 ; P1 ← P3	$ST2^+$ 1	
2^+	. Si β < 0 P1 ← (P2 + P3)/2 : point d'inflexion .... * P1 retenu P0 ← P1 P1 ← P2 P2 ← P3 $\alpha \leftarrow \beta$	$ST2^+$ 2-	Cas : . C(3) . T(3)
	* P1 non retenu P0 ← P1 P1 ← P2 .... P2 ← P3 $\alpha \leftarrow \beta$	2-	Cas : . T(2) + T(2)
	. Si β > 0	3^+	
3^+	... identique à 4^+		

Figure 9 - Automate d'extraction

./...

4^+	Lire un point Si lever .... STOP	$ST4^+$	
4^+	Si Point : P5 $\theta = (\overrightarrow{P3P4}, \overrightarrow{P4P5})$ Si $\lvert\theta\rvert >$ SCOIN .... P0 ← P4 , P1 ← P5	$ST4^+$ 1	
4^+	Si $\theta < 0$ P1 ← (P3 + P4)/2 : .... point d'inflexion * P1retenu P0 ← P1 P1 ← P4 .... P2 ← P5 $\alpha \leftarrow \theta$	$ST4^+$ 2^-	Cas : . C(5) . C(3) + C(3) . T(2) + C(4) . T(2) + T(2) + C(3)
4^+	* P1 non retenu P0 ← P3 P1 ← P4 .... P2 ← P5 $\alpha \leftarrow \theta$	2^-	Cas : . C(4) + T(2) . T(2) + C(3) + T(2) . C(3) + T(2) + T(2)
4^+	Si $\theta > 0$ Alors .... IND = 3 : tous les points ont été retenus PA ← P3 PB ← P4 PC ← PC .... $\eta \leftarrow \theta$	$ST4^+$ 5^+	Cas : C(5) 
4^+	IND = 2 : 4 points ont été retenus P0 ← P3 P1 ← P4 .... P2 ← P5 $\alpha \leftarrow 0$	2^+	Cas : C(4) + T(2) 
4^+	IND = 1 : 3 points ont été retenus P0 ← P2 P1 ← P3 P2 ← P4 .... P3 ← P5 $\alpha \leftarrow \gamma$ $\beta \leftarrow \theta$	3^+	Cas : C(3) + C(3) C(3) + T(2) + T(2) 

4^+	*IND = 0 : 2 points ont été retenus P0 ← P1 P1 ← P2 P2 ← P3 P3 ← P4 .... P4 ← P5 α ← β β ← γ γ ← θ	4^+	Cas : T(2) + C(4) T(2) + C(3) + T(2) T(2) + T(2) + C(3) 
5^+	Lire un point Si Lever .... STOP	$ST5^+$	
5^+	. Si point : PD ω = (PBPC , PCPD) . Si |ω| > SCOIN .... P0 ← PC P1 ← PD	$ST5^+$ 1	
5^+	. Si ω < 0 P1 ← (PB + PC)/2 .... point d'inflexion * P1 retenu P0 ← P1 P1 ← PC .... P2 ← PD α ← ω	$ST5^+$ 2^-	Cas : C(n-2-
5^+	* P1 non retenu P0 ← PB P1 ← PC P2 ← PD α ← ω	2^-	Cas : C(n - 3) + T(2)
5^+	. Si ω > 0 Alors .... IND = 1 tous les points ont été retenus PA ← PB PB ← PC .... PC ← PD η ← ω	$ST5^+$ 5^+	Cas : C(n-2)
5^+	IND = 0 (n-3) points ont été retenus P0 ← PC P1 ← PD	1	Cas : C(n -3) + T(2)

Quelques remarques :

IND *est un indicateur de travail permettant à l'automate de passer d'un état à l'autre*

ST_i^+ *module de segmentation d'une zone à i traits formant entre eux des angles positifs*

ST_i^- *module de segmentation d'une zone à i traits formant entre eux des angles négatifs*

En réalité, c'est le même module, sauf qu'on lui donne à l'entrée le signe des angles (+ ; -) pour qu'il puisse affecter le bon code à la courbe détectée (courbe positive C+ , courbe négative C-).

Point d'inflexion :

Le point d'inflexion est la limite géométrique entre deux courbes de sens différents dans le même tracé.

Il n'existe pas réellement dans le vecteur des points (zone aplatie du tracé : point supprimé par l'ECHANTILLONNEUR). On peut le rajouter au tracé dans certains cas de la manière suivante :

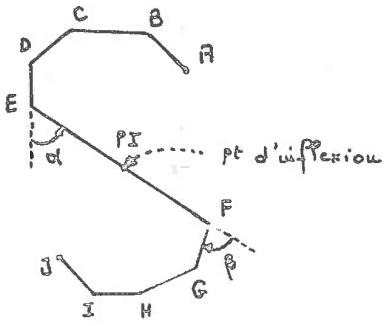
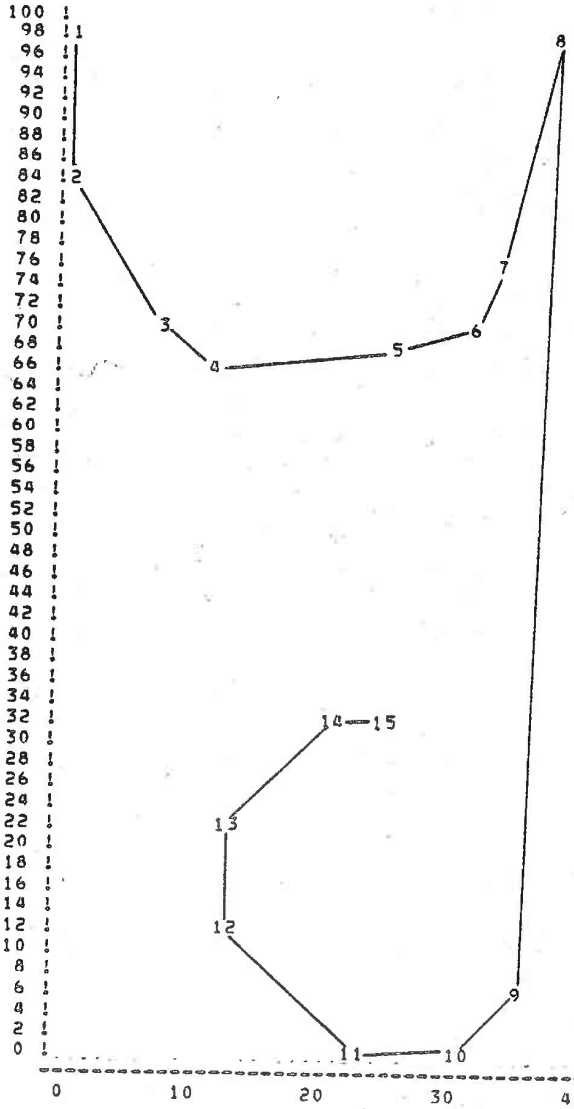


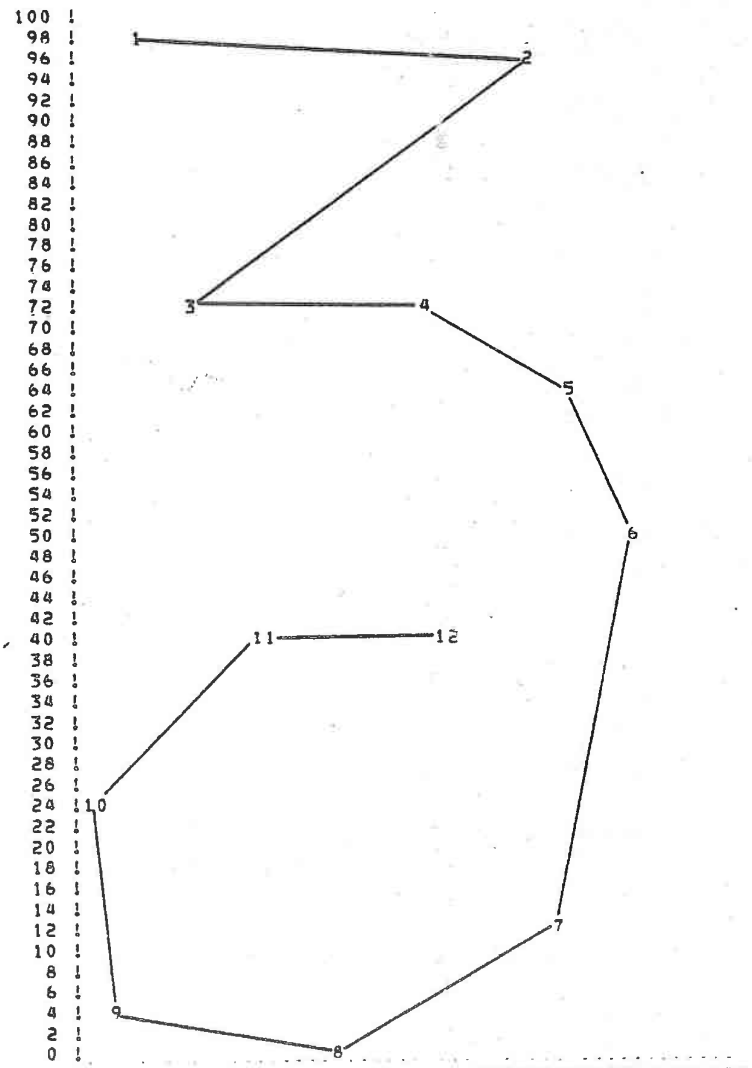
Figure 40

Lors du découpage en zones homogènes, le changement de courbure n'est observé par le module principal qu'au point F de la figure 40, car α et β sont de signes contraires. Nous estimons que la deuxième courbe du tracé (F G H I G) a commencé bien avant (c'est-à-dire au point d'inflexion PI) . Dans ce cas, on découpe le tracé de

façon provisoire au point PI qu'on rajoute au tracé (milieu du segment de droite E F), et on observe le résultat de la segmentation dans la première zone.
PI est retenu s'il fait partie d'une primitive : courbe, dans ce cas, on recommence l'analyse de la deuxième zone à partir de PI (PI , F , G , H , I , J).
Si le point PI n'est pas retenu, le découpage se fait au point E et le point d'inflexion est complètement rejeté.



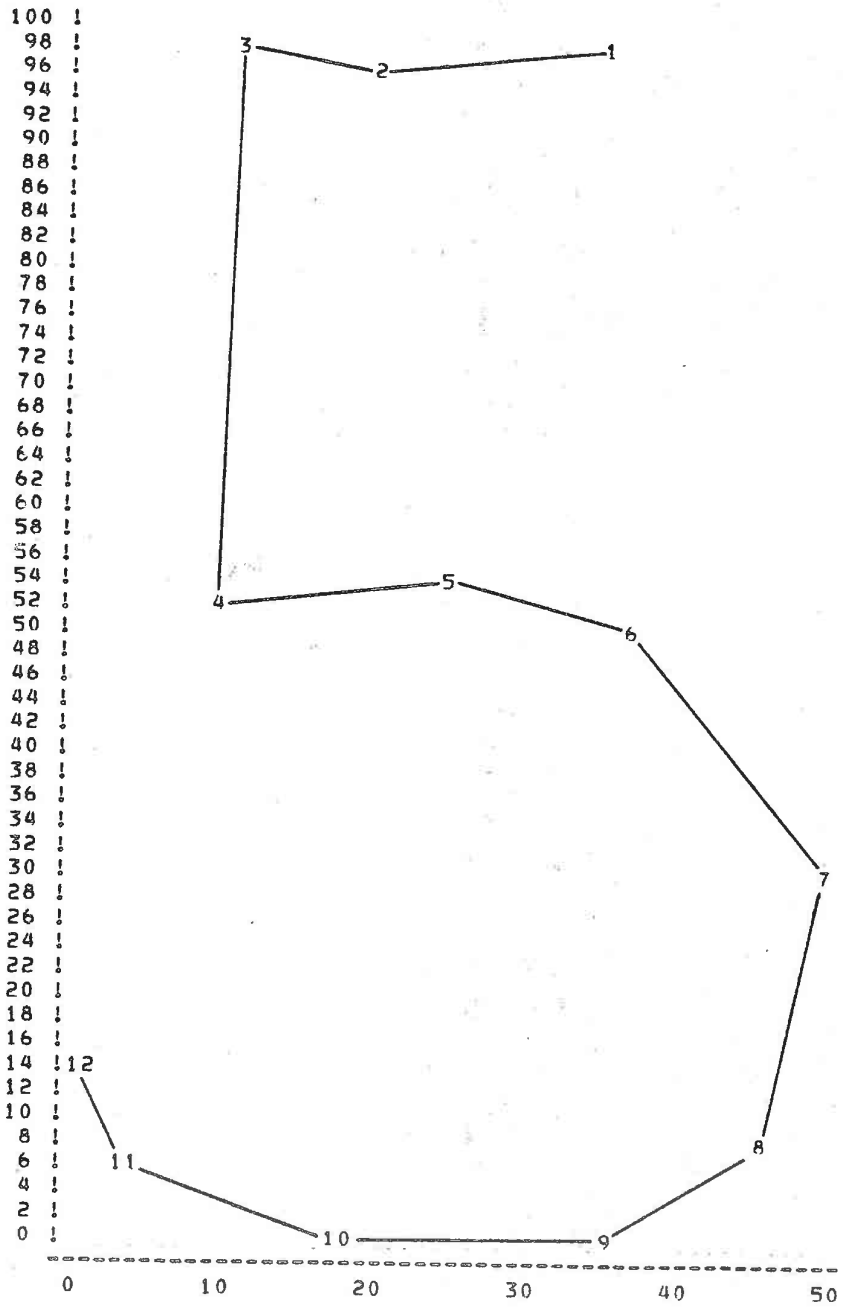
Exemple de segmentation double établie dans la première zone homogène du tracé



RESULTATS DE LA SEGMENTATION

LISTE 1 :	T	T	C	C	...	85
NPOINT :	2	2	4	7		
LISTE 2 :	T	T	C	T	C	... 77
NPOINT :	2	2	4	2	6	

Exemple de segmentation double établie dans la deuxième zone du tracé



RESULTATS DE LA SEGMENTATION

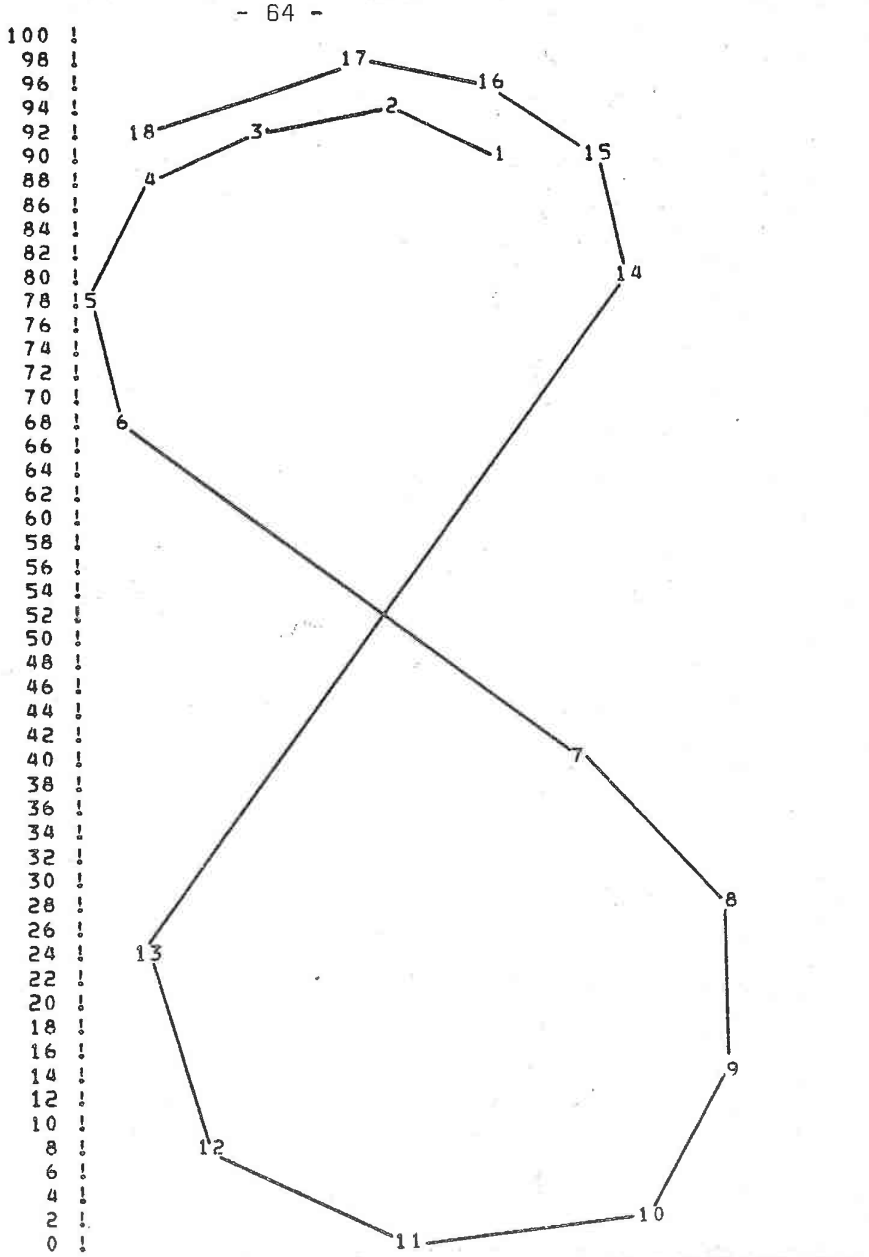
LISTE 1 : C T C ... 70

NPOINT : 3 2 9

LISTE 2 : T T T C ... 70

NPOINT : 2 2 2 9

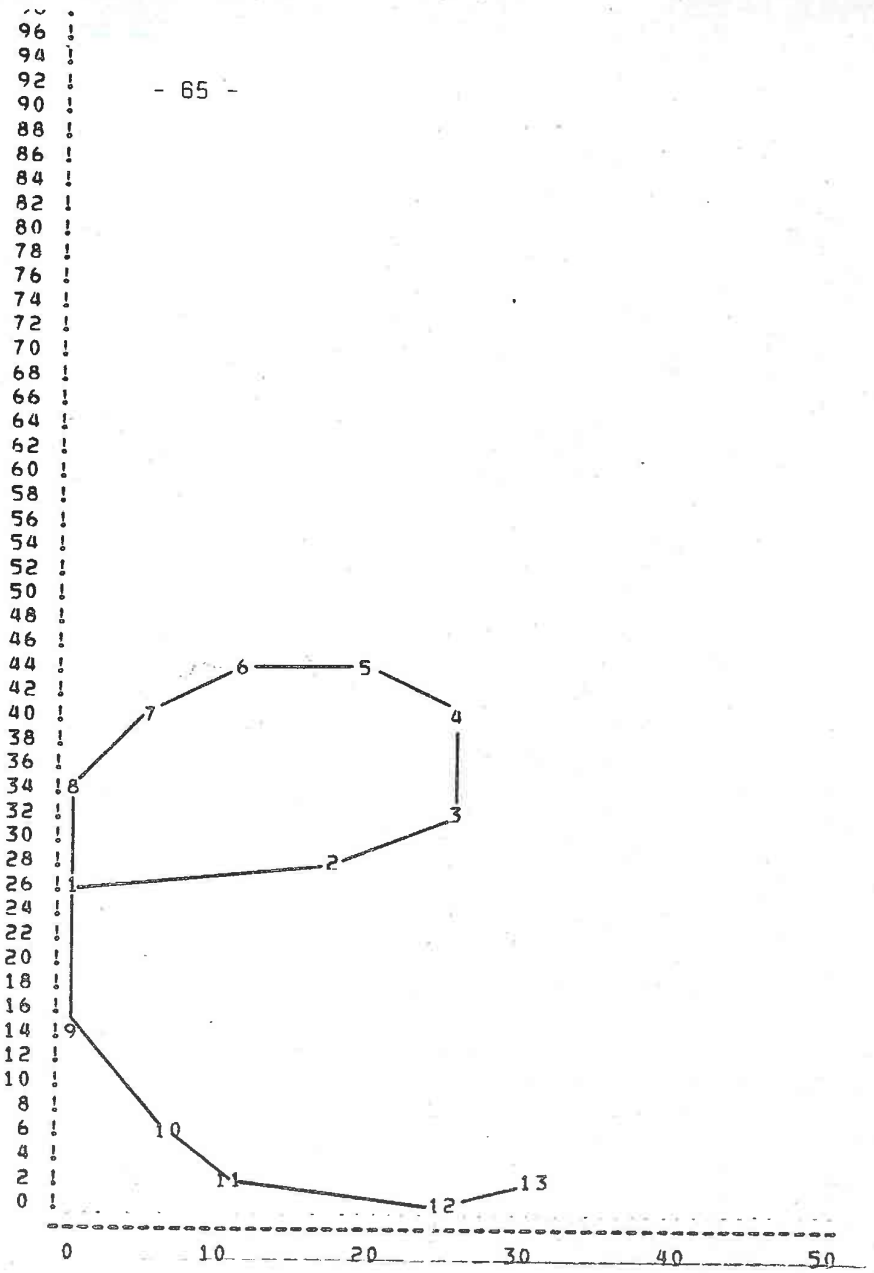
L'angle entre les deux premiers traits se trouve à la limite d'un seuil de comparaison ; ce qui explique les deux segmentations données et l'égalité entre les scores



RESULTATS DE LA SEGMENTATION

LISTE 1 :	C	C	T	C	... 50		
NPOINT :	6.5	8.5	2	5			
LISTE 2 :	C	C	T	C	... 43		
NPOINT :	6.5	8	2	5			
LISTE 3 :	C	T	C	T	C	... 48	
NPOINT :	6	2	7.5	2	5		
LISTE 4 :	C	T	C	T	C	... 47	
NPOINT :	6	2	7	2	5		

Exemple de segmentation où deux points d'inflexion ont été retenus (6.5 et 8.5)



RESULTATS DE LA SEGMENTATION

LISTE 1 : C ... 30

NPOINT : 13

LISTE 2 : C T C ... 70

NPOINT : 8 2 5

LISTE 3 : T C ... 55

NPOINT : 2 12

LISTE 4 : T C T C ... 85

NPOINT : 2 7 2 5

Exemple de segmentation d'une zone homogène en 4 listes différentes de primitives

IV.3 - ATTRIBUTS DES PRIMITIVES

IV.3.1 - But

La description structurelle par les listes de type de primitives (Trait - Courbe) exposée au paragraphe précédent s'avère insuffisante pour séparer tous les tracés manuscrits (pour les caractères latins). Plusieurs d'entre eux se trouvent segmentés de façon identique par la même chaine de primitives (voir le tableau de la figure 41). Il est évident qu'une élaboration beaucoup plus fine des codes de ces primitives peut lever cette ambiguïté entre les segmentations identiques et améliorer le processus de reconnaissance des caractères.

Le CLASSIFIEUR sera chargé de compléter la description de chaque primitive par des attributs simples relatifs à la taille et direction dans le tracé. Comme nous opérons sur des tracés déformés, les réponses du CLASSIFIEUR seront là-aussi multiples et valuées, ce qui élargira le nombre de segmentations finales du tracé et fixera le choix de l'analyseur à la reconnaissance sur les meilleures représentations.

Les attributs qu'on a choisis pour élargir le nombre de classes de chaque primitive sont de type topographique. On les déterminera à partir de quelques mesures d'angles et de longueurs déjà faites lors de la segmentation brute sur les traits composants. Ils seront assez significatifs des variations des caractères manuscrits de taille importante (plus d'un centimètre).

On note :

. pour le segment de droite :
- direction dans le plan (8 classes)
- longueur relative (3 classes)

. pour l'arc de cercle :
- sens d'écriture (2 classes)
- direction dans le plan (4 classes)
- longueur relative (3 classes)
- variation angulaire (4 classes)

Ces classes d'attributs seront déterminées pour chaque primitive extraite du tracé, de façon totalement indépendante du caractère qui la contient.

Codes	Tracés ambigus
T	/ — |
C	C O U
T + T	V L 7 1 > <
T + C	J e D h n b U P
C + T	G q a 4 R
C + C	ω 3 m S 2 y
T + T + T	N ⅃ ⊏ Z ⌐ ⊔
T + T + C	J 5 3
C + T + T	G 2
T + C + T	∩ U V R
C + T + C	S 2 y 9
C + C + C	8 B
T + T + L + T	A 4 1 F K N

Figure 41 - Quelques codages ambigus

IV.3.2 - Calcul des classes d'attributs

IV.3.2.1 - Pour le segment de droite

On distingue huit classes de directions différentes pour le segment de droite correspondant chacune à une zone du plan (a , b , c , d , e , f , g , h . Voir figure 42 b)

θ_1 et θ_2 sont des paramètres de cette répartition. Nous les avons choisis assez petits ($\theta_1 = 5°$, $\theta_2 = 10°$) pour laisser une large marge de tolérance aux obliques qui varient nettement plus que les horizontales et les verticales dans le tracé manuscrit.

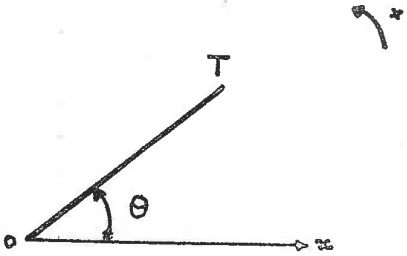


Figure 42 a
Direction du trait

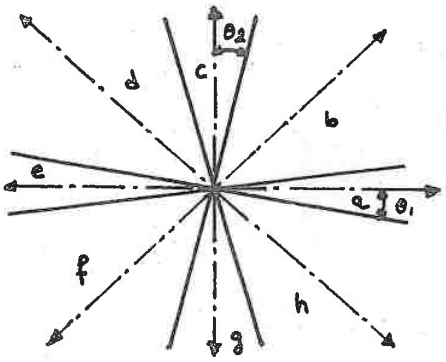


Figure 42 b
Directions principales dans le plan

Connaissant l'origine et l'extrémité du segment de droite T (voir figure 42 a) le CLASSIFIEUR détermine l'angle θ que forme le trait avec l'axe OX et lui affecte une classe de direction par comparaison de la valeur de θ aux limites des zones du plan.

T sera de classe :

- a si $-\theta_1 \leqslant \theta < \theta_1$
- b si $\theta_1 \leqslant \theta < \frac{\pi}{2} - \theta_2$

  etc ...

Deux réponses peuvent être données par le CLASSIFIEUR quand le trait se trouve à la limite de deux zones voisines. Une plage de recouvrement d'une ouverture arbitraire (ε) sépare les zones. Deux classes de direction sont donc affectées au trait si θ se trouve dans cette plage. Un score d'appartenance est affecté à chaque classe de direction détectée par le CLASSIFIEUR. Ce score sera très élevé si la réponse est unique et nettement plus faible au cas d'une réponse double.

La figure 43 montre la variation des scores en fonction de l'angle θ .

La direction, pour le trait, est une information très importante pour écarter l'ambiguïté entre les tracés des caractères (la figure 44 en donne quelques exemples). Elle a été retenue dans les divers systèmes de classification structurelle. Le nombre de classes et l'ouverture des zones varient d'un système à l'autre.

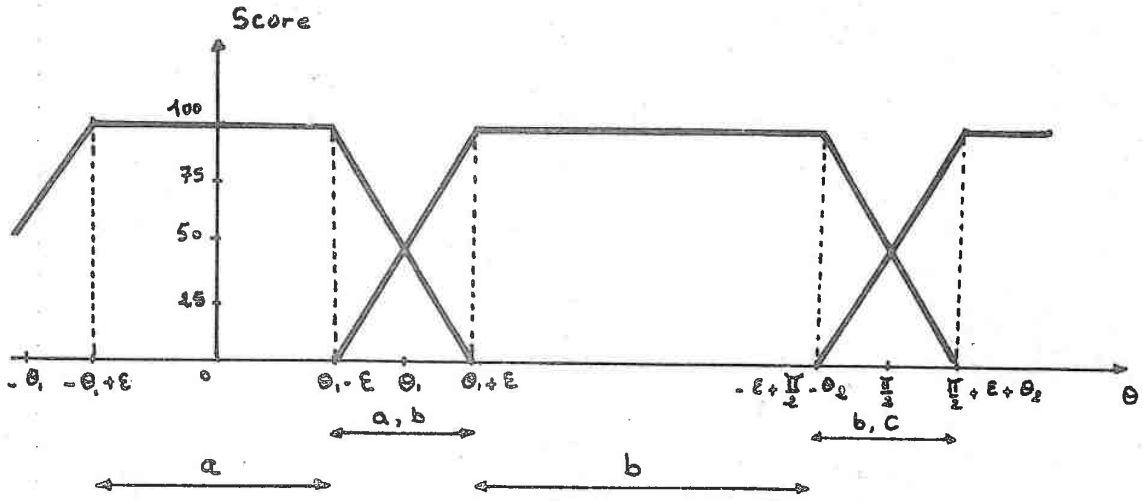


Figure 43 - Déterminations des classes de direction et des scores d'appartenance

Tg + Th + Tc Ta + Tg + Te Te + Tg + Ta Tg + Tc + Ta

Figure 44 - Classification par les directions

Longueur :

Les classes de direction ne suffisent pas toujours (voir figure 45) à séparer certaines représentations identiques. La taille relative du ou des traits composants par rapport à la taille du tracé peut compléter cette description et séparer ces tracés.

Les classes de longueur choisies ici sont au nombre de trois :

- 1. petite
- 2. moyenne
- 3. grande

Le CLASSIFIEUR calcule la longueur absolue l du trait et la compare au demi-périmètre P du cadre du tracé.

T sera de classe :

1 si $l \leqslant \frac{P}{3}$

2 si $\frac{P}{3} < l \leqslant 2\frac{P}{3}$

3 si $l > 2\frac{P}{3}$

Des plages de recouvrement de longueur (d) séparentlà-aussi les trois partitions de la longueur du tracé et deux réponses sont encore possibles. Un score de longueur est calculé pour chaque classe comme précédemment.

Figure 45

Tb + Th

Tb_2 + Th_2

Tb + Th ← insuffisance de la direction

Tb_2 + Th_3 ← discrimination par les longueurs

Représentation finale du segment de droite

Le segment de droite est donné par un code regroupant :

- son type (Trait)
- sa classe de direction
- sa classe de longueur

Vues les doubles réponses qu'on peut avoir à la classification, le trait peut être donné par le CLASSIFIEUR par quatre combinaisons de classesde longueur et de direction (voir figure 46).

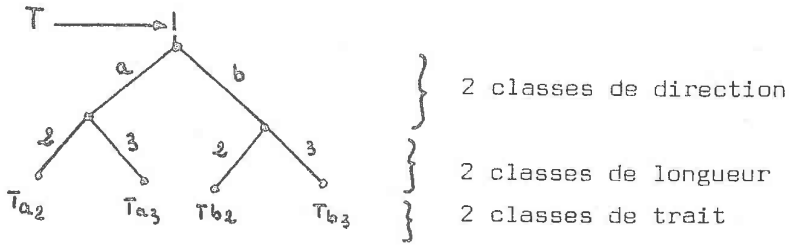


Figure 46 - Exemple de classification de la primitive TRAIT

Le score de la classe finale (feuille) est la moyenne des deux scores de classes de direction et de longueur qui la composent.

exemple :

$$S_c(Ta_2) = \frac{Sc(a) + Sc(2)}{2}$$

IV.3.2.2 - Pour l'arc de cercle

a) sens d'écriture

Le sens de la courbe est observé lors de la segmentation par l'automate d'extraction. On distingue deux classes de sens :

- Sens positif ou direct :  : C+

  Quand les angles formés par les traits consécutifs de la courbe sont positifs.

- Sens négatif ou rétrograde :  : C-

  Quand les angles formés par les traits consécutifs de la courbe sont négatifs.

On distingue ainsi les caractères différents de la figure ci-dessous uniquement par cette information.

C+ + C+ C- + C- C+ + C- C- + C+ C+ + C- + C+ C- + C- + C-

Figure 47

b) direction

Le nombre de classes de direction de la courbe est beaucoup plus réduit que celui déterminé pour le trait. La courbe varie énormément dans le tracé et une classe de direction trop précise risque d'être trop stricte et par conséquent peu suffisante pour lever l'ambiguité. Seulement, quatre classes de direction sont retenues dans notre cas. Ces classes sont déterminées par comparaison de l'angle θ du seg-

ment de droite T joignant l'origine et l'extrémité de la courbe (figure 48 a) aux limites des 4 zones a, b, c et d du plan. Ces zones sont de même ouverture ($\pi/2$) et séparées par de très larges plages de recouvrement ($\varepsilon = 30°$), de peur que la direction ne soit trop restrictive.
Deux réponses valuées sont là*aussi possibles.

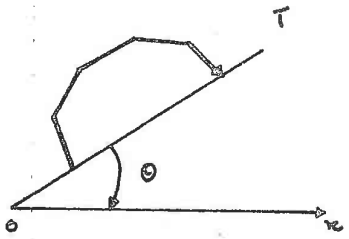


Figure 48 a - Direction de la courbe

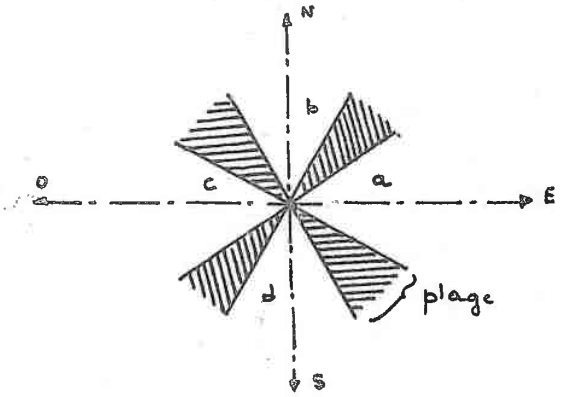


Figure 48 b -
Directions principales du plan

Exemple :

C- + C- C- + C- ← insuffisance du sens
C-d + C-d C-a + C-a ← discrimination par les directions

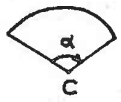
c) Longueur relative :

Le CLASSIFIEUR calcule la longueur absolue de la courbe par sommation des longueurs des traits composants et détermine comme pour le segment de droite la classe (1, 2 ou 3) de longueur relative. Deux réponses sont là*aussi possibles.

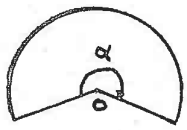
d) Variation angulaire :

La variation angulaire indique l'ouverture de la courbe. Elle peut être très importante pour séparer des tracés difficiles comme celui du C et du O . La courbe du C est ouverte alors que celle du O est fermée.
Quatre classes de variation angulaire sont observées :

- O : ouverte (O)
- O' : moyennement ouverte (MO)
- f' : moyennement fermée (MF)
- f : fermée (F)



courbe ouverte



courbe fermée

Ces classes sont déterminées par comparaison de l'angle au centre C du cercle circonscrit aux limites des 4 zones du plan $(0 - \frac{\pi}{2} , \frac{\pi}{2} - \pi , \pi - \frac{2\pi}{3} , \frac{2\pi}{3} - 2\pi)$

$$\alpha = \sum_i \alpha i + \beta_1 + \beta_2$$

α_i = angle formé par deux traits consécutifs

β_1 et β_2 : angles formés par les tangentes T_1 et T_2 au cercle à l'origine et à l'extrémité de la courbe. Ils sont assez difficiles à déterminer. On les évalue à :

$$\beta_1 + \beta_2 \simeq 2 \times SAECH$$

SAECH = angle d'échantillonnage

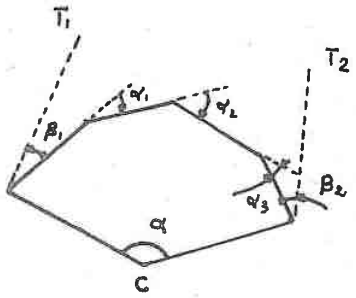


Figure 49 - Calcul de la variation angulaire

Deux classes de variation angulaire peuvent être données par le CLASSIFIEUR à la limite des zones.
Outre le C et le O, la variation angulaire peut être utile pour séparer le U du V (voir figure 50).

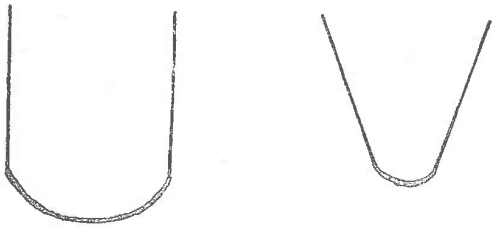


Figure 50 - Discrimination par la variation angulaire

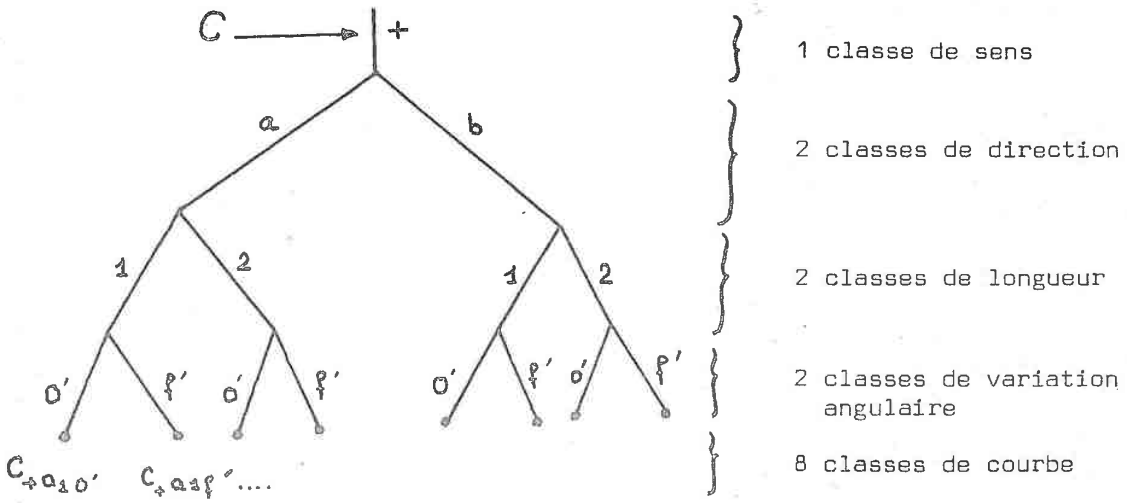
La courbe du V est généralement plus fermée que celle du U.

Représentation finale de la courbe :

La courbe est donnée par un code contenant :

- type
  - . courbe positive
  - . courbe négative
- classe de direction
- classe de longueur
- classe de variation angulaire

Le nombre de courbes obtenu après classification est nettement plus important que celui du trait. On peut avoir 8 classes de courbes différentes.



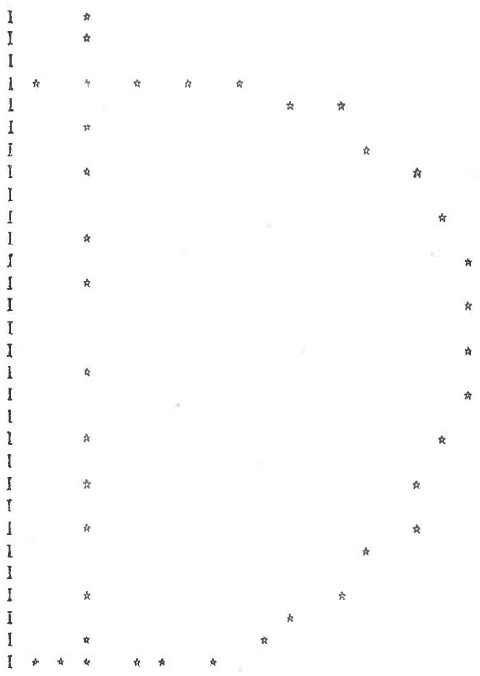
Un score de classification est déterminé pour chaque feuille. Il est calculé par la moyenne des 3 scores de classes

$$S_c\ (C_{+a+f}) = \frac{Sc(a) + Cs(1) + Sc(f)}{3}$$

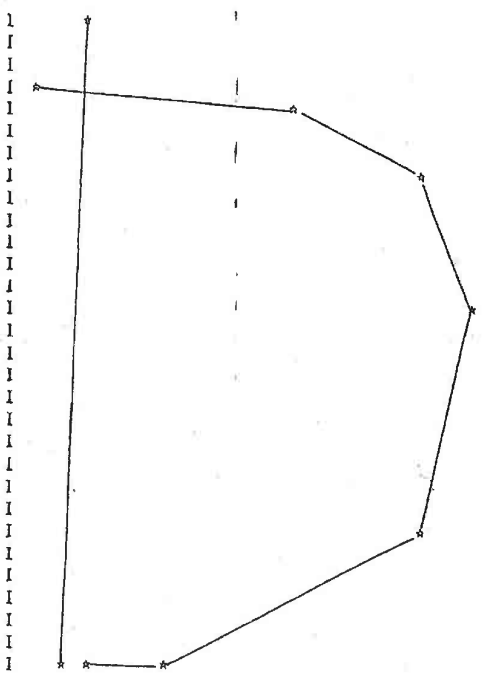
Les figures 52 - 54 donnent trois exemples de segmentations élaborées.

TRACES DU CARACTERE: D

TRACE BRUT



TRACE ECHANTILLONNE



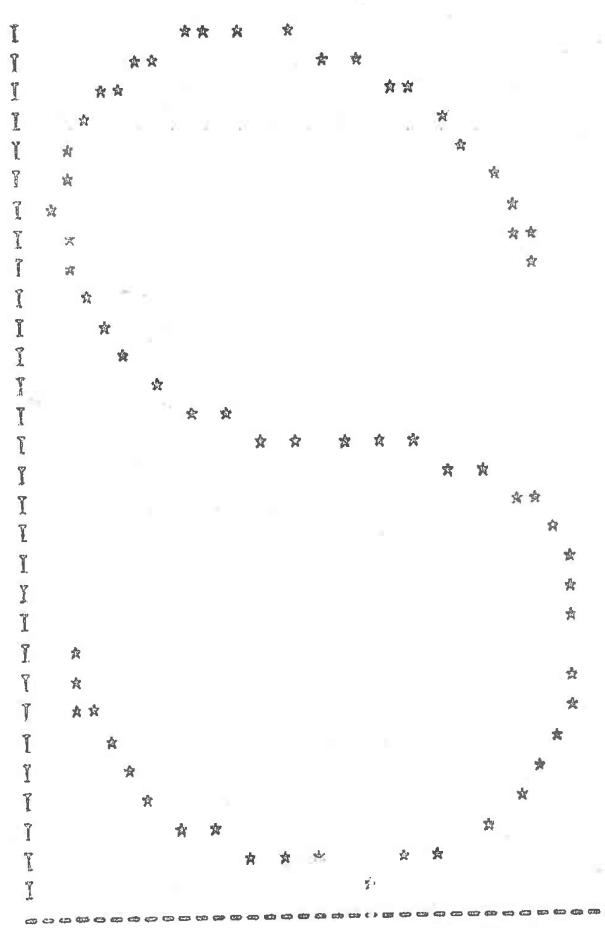
RESULTATS DE LA SEGMENTATION

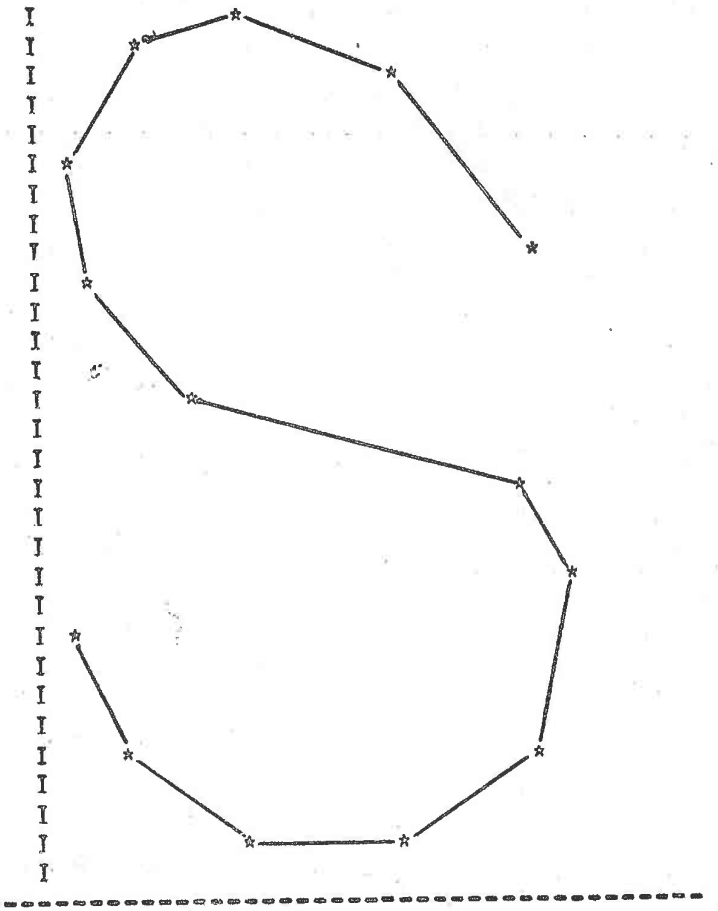
		DIRECTION	LONGUEUR	V.ANGULAIRE
TRACE NO: 1	SCORE=99			
	TRAIT	S	G	
TRACE NO: 2	SCORE=55			
	COURBE	S	G	MO

Figure 52

Exemple de segmentation de caractère à deux tracés

TRACE BRUT





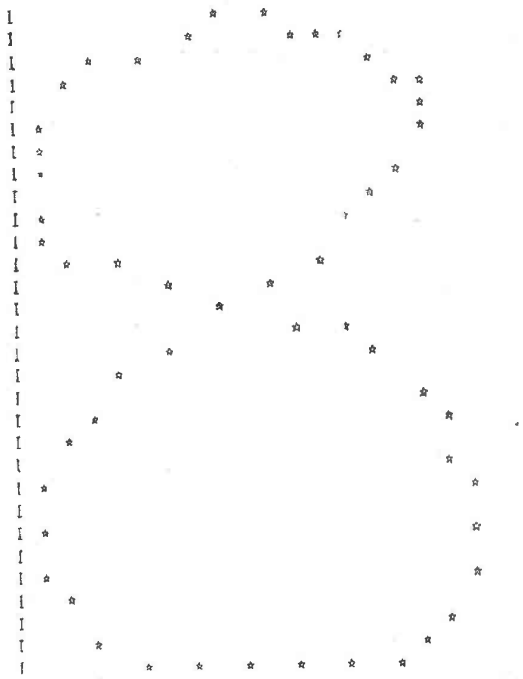
RESULTATS DE LA SEGMENTATION

	DIRECTION	LONGUEUR	V.ANGULAIRE
COURBE	S	G	MO
COURBE	O	G	MO
LEVER	SCORE= 73		
OU BIEN			
COURBE	O	G	MO
COURBE	O	G	MO
LEVER	SCORE= 73		

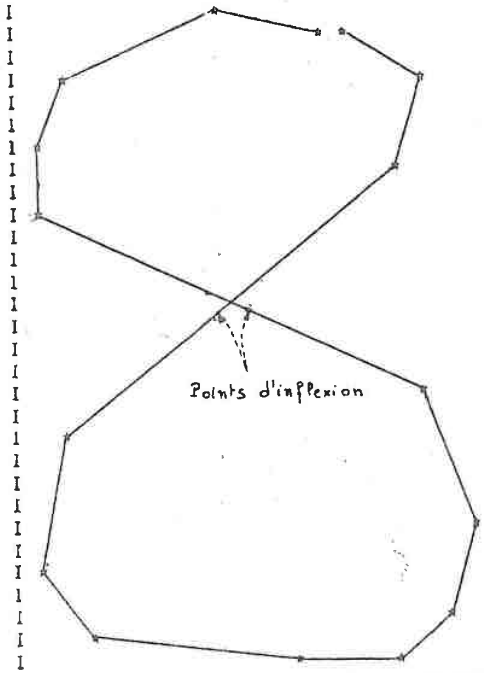
Figure 53

TRACES DU CARACTERE: 8

TRACE BRUT



TRACE ECHANTILLONNE

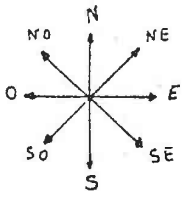


RESULTATS DE LA SEGMENTATION

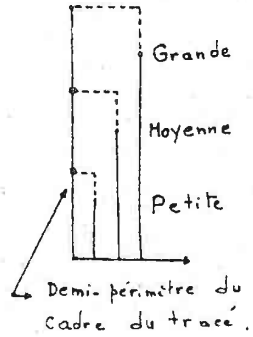
	DIRECTION	LONGUEUR	V.ANGULAIRE
TRACE NO: 1 SCORE=55			
COURBE	S	M	MO
COURBE	E	G	O
COURBE	N	M	MF

Légende:

DIRECTION



LONGUEUR



V. ANGULAIRE

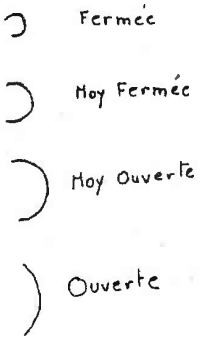
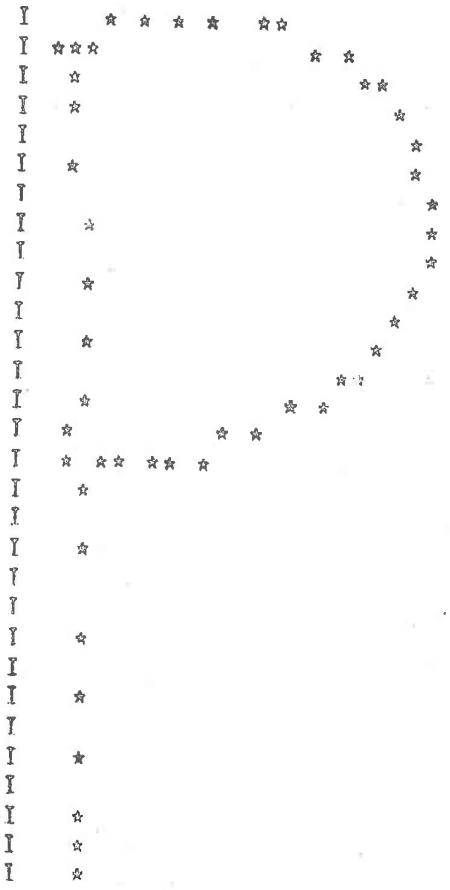


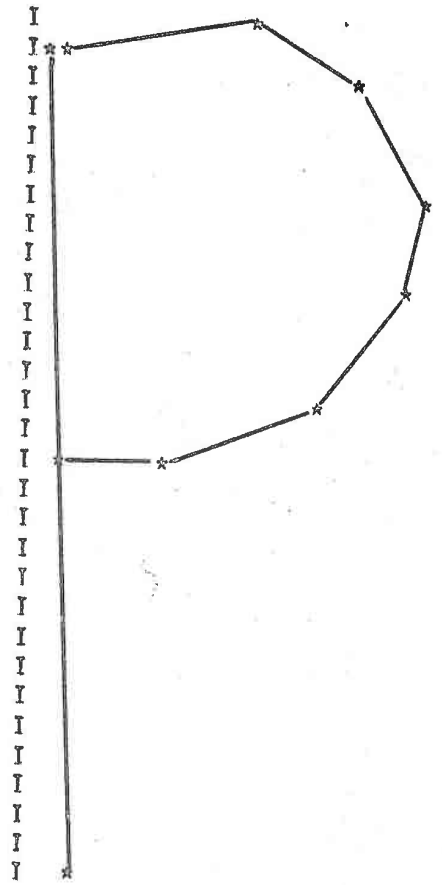
Figure 55

. Exemple de segmentation où deux points d'inflexion ont été retenus

TRACE BRUT



TRACE ECHANTILLONNE



RESULTATS DE LA SEGMENTATION

		DIRECTION	LONGUEUR	V.ANGULAIRE
TRAIT		S	G	
LEVER	SCORE= 99			
COURBE		S	G	
LEVER	SCORE= 68			

OU BIEN

TRAIT		S	G	
LEVER	SCORE= 99			
COURBE		S	G	
LEVER	SCORE= 63			

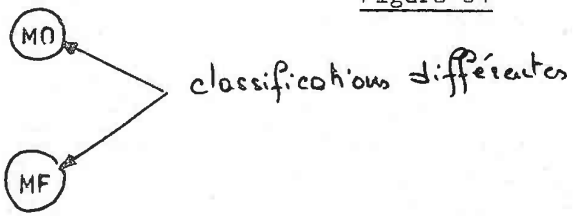


Figure 54

IV.3.3 - Représentation structurée du tracé en classes de primitives

A la suite de la classification multiple et valuée appliquée sur chaque primitive, le tracé manuscrit se trouve représenté par le CLASSIFIEUR par un arbre important où chaque chemin décrit une suite de classes de primitives. Chaque arc contient une classe donnée par un code spécifiant le type de la primitive et les classes de ses attributs comme suit :

Pour le segment de droite :

Type	DIR	LG

Type = 1 DIR = direction

$1 \leqslant DIR \leqslant 8$ LG = longueur relative

$1 \leqslant LG \leqslant 3$

Exemple : le code 1 4 3 décrit un trait de grande taille et de direction Nord-Ouest (voir figure 51)

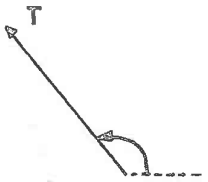


Figure 51 - Codage du trait

Pour l'arc de cercle

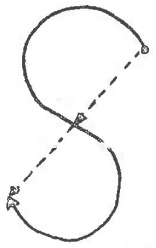
Type	DIR	LG	VAR

Type = { 1 si courbe positive
{ 2 si courbe négative

$1 \leqslant DIR \leqslant 4$ VAR = variation angulaire

$1 \leqslant LG \leqslant 3$

$1 \leqslant VAR \leqslant 4$



code : 1223 + 2232

Une primitive lever de crayon sépare les tracés consécutifs dans le chemin de l'arbre. On la représente par un code arbitraire (= 100).

Structure de l'arbre du tracé :

L'arbre du tracé est donné par un tableau à deux dimensions (TRAC). Chaque ligne correspond à un arc dont le format est le suivant :

CODE	LHOR

où, le code est, soit :

- une classe de primitives
- un lever de crayon

LHOR est, soit le lien horizontal avec la primitive de droite dans l'arbre, soit un score de segmentation (SG) calculé pour tout le chemin si le code indique un lever de crayon.

$$SG = \frac{n\ Ss + Sc}{n + 1}$$

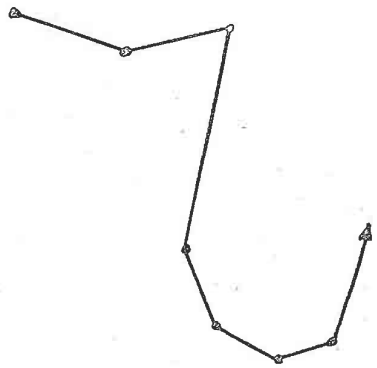
où

Ss = moyenne des scorès de segmentation calculés dans les différentes zones homogènes du tracé,

Sc = moyenne des scores de classification calculés par le CLASSIFIEUR pour chaque primitive du chemin.

n est un poids qu'on affecte au score de segmentation pour mieux le valoriser.

Exemple



Codes : $Te_1 + Tg_1 + C_{+d_{33}}$

$Te_1 + Tg_1 + C_{+C_{33}}$

$Te_1 + Tg_1 + C_{+d_{22}} + Td_1$

$C_{+C_{11}} + Tg_1 + C_{+d_{33}}$

$C_{+C_{11}} + Tg_1 + C_{+C_{33}}$

$C_{+C_{11}} + Tg_1 + C_{+d_{22}} + Td_1$

arbre du tracé

1	151	10
2	171	0
3	2433	5
4	100	80
5	2333	7
6	100	75
7	2422	0
8	141	0
9	100	50
10	2311	0
11	171	0
12	2433	14
13	100	50
14	2333	16
15	100	76
16	2422	0
17	141	0
18	100	15
19		

IV.4 - REPRESENTATION FINALE DU CARACTERE

Le caractère manuscrit regroupe un ou plusieurs tracés. Il sera représenté par le CLASSIFIEUR par un tableau à deux dimensions (CAR) regroupant la succession des divers tracés dans l'ordre de leur entrée graphique.

Ce tableau sera accompagné par les informations suivantes :

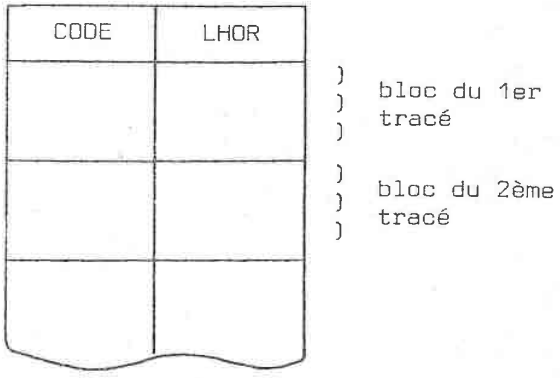
- Nom (nom du caractère : A, B, ...) appelé NONCAR
- Nombre de tracés qui composent le caractère : NTRAC
- et un tableau CADRE contenant des informations se rapportant à chaque tracé, telles :
  - . le cadre du tracé :
    MINX , MAXX , MINY , et MAXY
  - . et les coordonnées de ses extrémités
    OX , OY , EX et EY

L'arbre du tracé n'est pas rangé dans sa totalité dans le tableau CAR. Le CLASSIFIEUR sélectionne parmi les chemins de l'arbre un nombre NSEL réduit généralement à 3 ou 4 et dont le score général de segmentation est supérieur à un certain seuil (SSEL).

NSEL et SSEL sont des paramètres qu'on peut changer directement par programme. Leur valeur n'est pas la même suivant qu'on se trouve en phase d'apprentissage ou de reconnaissance.

Structure de CAR

La structure de CAR est une structure de bloc (TRAC) où chacun contient l'arbre d'un tracé (voir figure 52).
La taille de chaque bloc est fixe. Nous verrons l'intérêt d'une telle représentation dans la phase de reconnaissance.



IV.5 - CONCLUSION

Nous avons développé dans ce chapitre un procédé efficace de segmentation de tracés manuscrits en primitives très élaborées. Ces primitives sont des sous-formes naturelles qu'on peut distinguer à vue d'oeil dans le tracé. Ce sont des segments de droite de différentes tailles et directions et des arcs de cercles de différentes orientations et courbures. Elles sont extraites du tracé à l'aide d'un automate de segmentation dans des zones homogènes localisées dans le tracé. L'automate cherche ces primitives de façon globale à l'aide de tests pertinents utilisant des mesures simples de taille et d'angle des traits composants.

De peur que la segmentation ne soit trop restrictive à ce niveau et pour favoriser surtout l'interaction entre ce niveau d'extraction et le niveau supérieur d'analyse, l'automate propose pour chaque tracé plusieurs listes de classes de primitives. Le niveau d'analyse fera son choix en ne retenant que les listes qui ont d'une part un score de segmentation élevé et celles qui auront un cheminement possible dans un arbre de décision construit par apprentissage.

Cette segmentation a été utilisée au départ pour la description structurelle de différents modèles de dessins dans les travaux de MASINI (42). Les classes de primitives ont été réduites.

Un analyseur syntaxique aidé d'une grammaire de description spécifique à chaque modèle reconstruit le dessin par assemblage de ses différentes primitives.

CHAPITRE 5

APPRENTISSAGE ET RECONNAISSANCE DES CARACTERES

V.1 - INTRODUCTION

Après prétraitement et segmentation par l'automate d'extraction, le caractère inconnu est représenté par un arbre de chemins multiples de classes de primitives. Pour le reconnaître, il faut comparer ses listes de primitives à celles de différents modèles de caractères rangés dans un dictionnaire construit dans une phase préalable d'apprentissage, et fournir une réponse correcte, non ambiguë avec un taux de reconnaissance satisfaisant.

Avant de passer à l'exposé de la technique de comparaison adoptée dans notre système, nous allons voir comment se fait l'apprentissage des caractères à partir de la description structurelle qu'on a leur a attribué et quelles sont les améliorations qu'on peut déjà faire à ce niveau pour réduire d'une part les ambiguïtés entre les représentations identiques et éviter d'autre part la mauvaise reconnaissance.

V.2 - APPRENTISSAGE DES CARACTERES

Il s'agit de faire apprendre au système la configuration structurelle de chaque modèle de caractère en lui fournissant son nom et toutes les listes de primitives nécessaires pour le décrire.

D'autre part, comme nous voulons utiliser notre système pour reconnaître un jeu important de caractères avec des écritures très variées, nous devons étendre l'apprentissage à toutes les formes d'écriture (apprentissage collectif) pour que son utilisation soit universelle.

Le jeu de caractères à ranger dans le dictionnaire est composé de :

- caractères alphabétiques majuscules, minuscules
- chiffres arabes
- signes mathématiques

Nous comptons, pour chacun de ces caractères, cinq à dix écritures différentes (voir la variation du A dans la figure 56). Ce qui fait environ 400 à 800 caractères différents à apprendre.

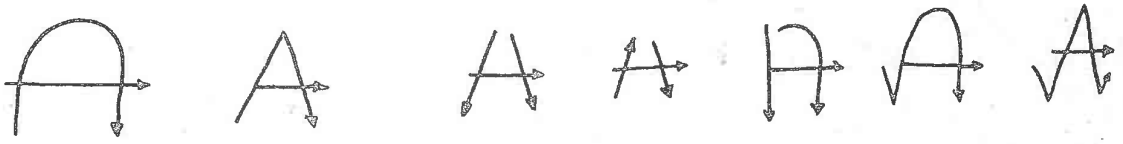


Figure 56 - Différentes écritures du A

Qui dit écritures différentes, dit aussi segmentations différentes. Nous comptons en moyenne cinq segmentations différentes par écriture d'un caractère, ce qui fait environ 2000 à 3000 listes différentes de primitives à ranger dans cette phase d'apprentissage. Il faut donc une organisation du dictionnaire permettant de ranger toutes ces segmentations possibles en peu de place mémoire.

La recherche dans le dictionnaire doit être, d'autre part, assez rapide, presque instantanée, pour qu'on puisse concevoir une reconnaissance en temps réel pendant l'écriture.

L'organisation du dictionnaire consiste en un arbre de décision dont les chemins représentent les listes de primitives des caractères et les feuilles, leur nom. Un arc peut être commun à plusieurs caractères, par contre, une feuille ne doit correspondre qu'à un seul caractère.

Cet arbre a donc la même structure que les arbres de segmentation construits par l'automate d'extraction, ce qui facilite les insertions des nouveaux chemins au cours de l'apprentissage et la suppression de certains d'entre eux suite à un apprentissage erroné.

Vu le nombre important de caractères à entrer, nous avons préféré faire cet apprentissage en deux temps dans le système, de la manière suivante :

- avant la reconnaissance

L'utilisateur introduit, au départ, un noyau réduit de caractères "parfaits" et suffisants pour débuter la reconnaissance.
Ces caractères sont introduits un à un et accompagnés d'un nom. Ils sont segmentés par l'automate et rangés directement dans l'arbre de décision.

- pendant la reconnaissance

Ce noyau de caractères est complété par la suite de façon continue pendant la phase de reconnaissance par d'autres écrivains.

En effet, l'utilisateur ne peut pas prévoir toutes les variations possibles de l'écriture de chaque caractère. On charge quelques écrivains d'introduire ces écritures de façon naturelle pendant la reconnaissance (sans leur imposer le mode d'écriture).

Chaque écrivain s'installe devant la tablette et écrit une phrase ou une formule mathématique. Le système isole chaque caractère dans la chaine entrée et essaie de le reconnaître. En cas d'échec, l'écrivain peut ordonner au système d'apprendre le caractère mal reconnu en lui affectant un nom. L'apprentissage ne se fait pas de façon automatique sans l'intervention de l'écrivain , car un caractère peut

être très entaché de bruits (parfois de parasites), et son apprentissage serait hasardeux. Le schéma du module d'apprentissage et de reconnaissance est illustré par la figure 57.

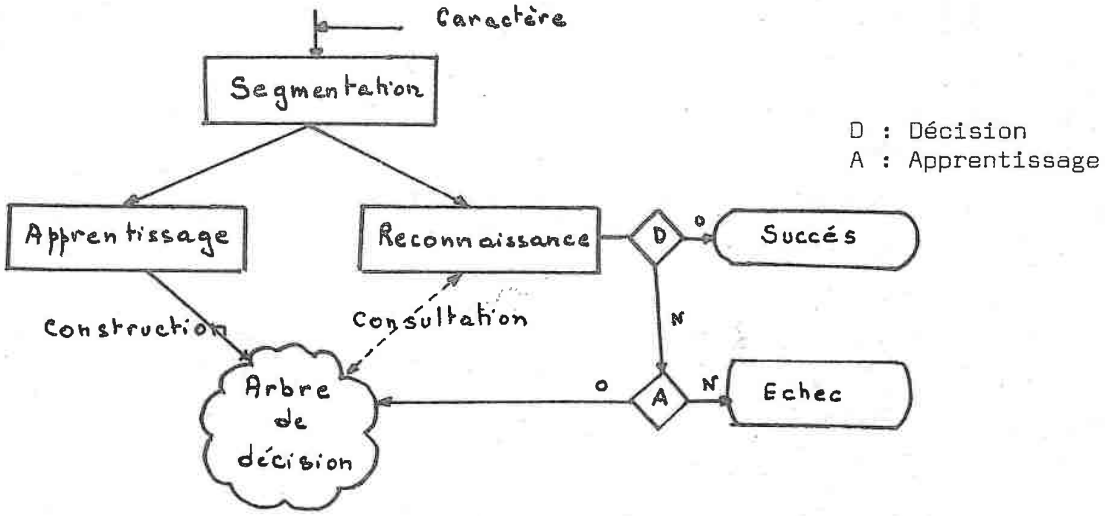


Figure 57 - Schéma du système d'analyse et de décision

Lever d'ambiguïté :

La classification fine faite à l'étape précédente par l'automate et le CLASSIFIEUR n'est pas suffisante pour séparer tous les caractères. Certains d'entre eux peuvent être donnés par la même séquence de classes de primitives (voir figure 58) et leur introduction dans l'arbre de décisions peut créer des ambiguïtés à la reconnaissance

Pour lever cette ambiguïté entre les représentations identiques, nous introduisons au niveau des feuilles de l'arbre des tests de séparation portant sur les attributs de position et de raccordement entre les différents tracés du caractère, ce qui éclatera le chemin ambigu en plusieurs chemins correspondant chacun à un seul caractère. Le cheminement dans cette partie de l'arbre sera guidé par la réponse de ces tests.

Nous reviendrons par la suite sur l'introduction de ces tests pour écarter automatiquement ces ambiguïtés et nous donnerons la liste des différentes représentations identiques rencontrées au cours de cette reconnaissance.

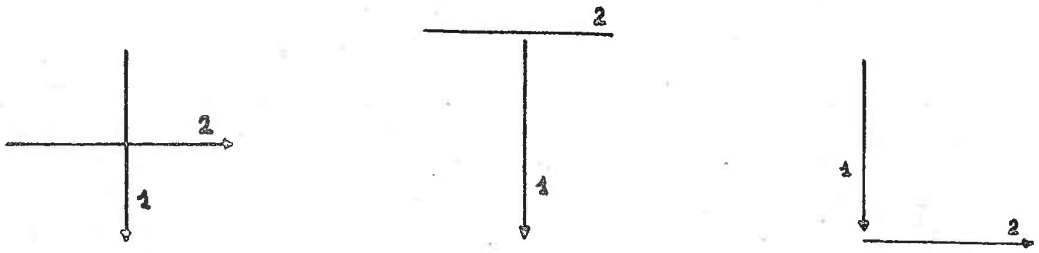


Figure 58 - Code commun : Tg_3 + L + Ta_3

L = lever de crayon

V.2.1 - Construction de l'arbre de décision

La construction de l'arbre de décision se fait par rangement des divers chemins de primitives du caractère à apprendre et introduction de tests discriminants au niveau des feuilles en cas d'ambiguïté.

V.2.1.1 - Rangement des caractères

Structure de l'arbre :

L'arbre de décision est représenté dans le système par un tableau à deux dimensions (TARB) donc chaque ligne décrit un arc de l'arbre et a la structure suivante :

TRANS	LFILS	LHOR	TEST	ACTION

où :

TRANS est une transition qui peut être soit :

- un code de primitives
  - . Trait
  - . Courbe
- un lever de crayon
- une fin de liste de primitives
- le nom d'un caractère

LFILS : lien avec le premier fils

LHOR : lien horizontal avec la primitive "frère"

TEST : est donné par un nombre n

si n = 0 : il n'existe pas de test en cette feuille

si n ≠ 0 : il existe un test en ce noeud ; n est dans ce cas le numéro d'une case d'un fichier TEST où se trouvent rangées des informations concernant le test et son mode d'emploi.

Nous reviendrons par la suite sur l'introduction de ces tests

Action :

L'action permet de passer d'une transition à une autre dans l'arbre de décision . L'action est spécifique de la nature de la transition. On distingue trois types d'actions.

Action 1 : La transition en cours est :
- une primitive trait, courbe
- un arc vide (TRANS = 0)

Fonction :
Lire la primitive suivante dans l'arbre du caractère et la ranger à partir de ce noeud.
Le rangement est effectué par une procédure spéciale appelée RANGE.

Action 2 : La transition en cours est :
- un lever de crayon

Fonction :
Ranger les informations concernant le chemin du tracé déjà rangé dans un buffer spécial pour le traitement ultérieur.
Si le lever est le dernier dans la liste de primitives (valeur du pointeur des TCAR).

alors :
Chercher ou construire dans l'arbre et à partir de ce noeud une primitive FIN (F) : marqueur de fin de chemin.

Sinon :
Calculer l'adresse de la primitive suivante : adresse de la première primitive du tracé suivant dans TCAR et appeler RANGE.

Action 3 : La transition en cours est :
- une fin de chemin

Fonction :
Si : existe un test en ce noeud,

alors :
l'appliquer sur le chemin entré. La réponse du test indique dans quelle branche gauche ou droite il faut acheminer le caractère (la structure de l'arbre à ce niveau est binaire)

sinon :
Si : existe une feuille en ce noeud

alors :
ranger le nom du caractère entré et remonter dans les deux arbres (arbre du caractère et arbre de décision) pour le rangement d'un autre chemin.

sinon :
comparer le nom de la feuille à celui du caractère entré

Si les noms sont les mêmes :

alors :
signaler la présence d'un nouvel échantillon du même caractère en ce noeud et remonter dans l'arbre pour le rangement d'un nouveau chemin (une pile de travail : PILARC, mise à jour par la procédure RANGE, indique les adresses relatives dans chaque arbre où il faut remonter. Le rangement est considéré comme terminé quand la pile est vide).

sinon :
calculer un test discriminant entre le caractère entré et ceux existants et créer deux nouvelles feuilles correspondant chacune à un caractère (voir figure 59).

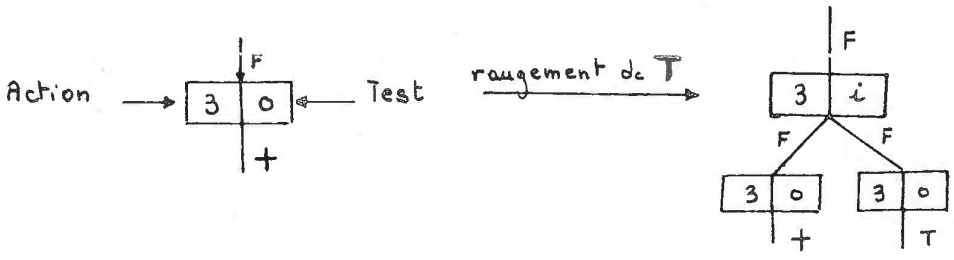


Figure 59 - Séparation des caractères par les tests

Procédure RANGE :

- Lit la primitive la plus à gauche dans l'arbre du caractère à partir d'une adresse indiquée et retient dans la pile PILARC l'adresse de la primitive "frère" (si elle existe : LHOR ≠ 0) et l'adresse de son implantation dans l'arbre de décision.
- Range cette primitive dans l'arbre de décision à partir de l'adresse indiquée par l'action.

Le rangement consiste à comparer son code à ceux des arcs descendants à partir de ce noeud et parcourir l'un des arcs en cas d'égalité. Si aucun code ne correspond au code de la primitive entrée, la procédure RANGE crée un nouvel arc (arc frère) qui contiendra cette primitive.

Exemple :

La figure 60a donne un arbre contenant trois caractères différents I, +, B ; chacun d'entre eux est à une feuille différente. Les numéros 1, 2, 3 sont les numéros des trois actions précédemment définies.

La figure 60b donne l'état de l'arbre de décision après rangement du caractère L donné par l'automate par les deux listes suivantes :

$Tg_3 + L + Ta_3$

$Tg_3 + L + C_{+a_{31}}$

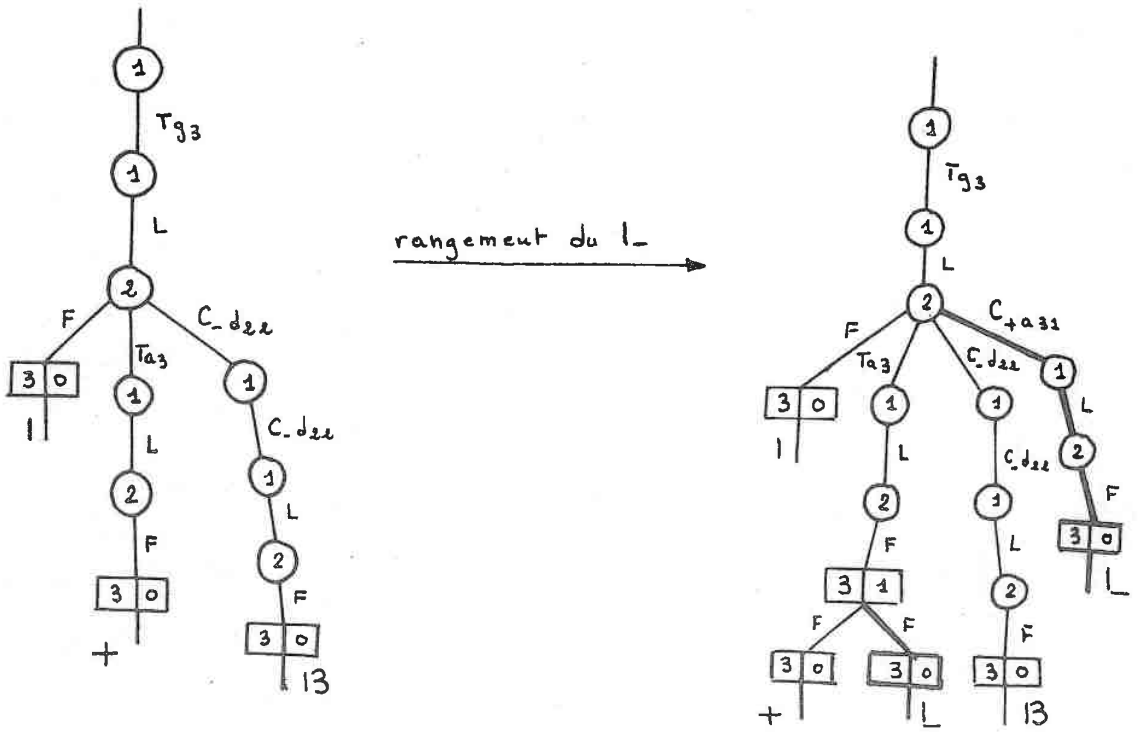


Figure 60 a - Etat de l'arbre avant rangement du L

Figure 60 b - Etat de l'arbre après rangement du L

V.2.1.2 - Introduction des tests

L'introduction des tests se fait automatiquement dans l'arbre de décision au niveau des feuilles et ceci chaque fois qu'une ambiguïté s'y présente. Ces tests consistent à faire un certain nombre de mesures pertinentes sur le couple de caractères ambigus (ou couple d'ensembles d'échantillons) et retenir celles qui les séparent le mieux. Ces caractères sont ainsi séparés et rangés dans deux nouvelles feuilles. La réponse du test calculée à partir de cette mesure guide le cheminement dans l'une ou l'autre feuille.

Le tableau ci-dessous (cf. figure 61) donne la liste des principales segmentations ambiguës qu'il s'agit d'écarter dans l'arbre de décision.

Codes	Caractères ambigus
$Ta_3 + L + Tg_3$	
$Tg_3 + L + Ta_3 + L + Ta_3$	
$Tg_3 + L + C_{+g_{33}}$	
$Tg_3 + L + Th_3$	
$Tg_3 + L + C_{\bullet g_{23}} + Th_3$	
$Tg_3 + L + Ta_3 + L + Tg_3$	
$Tg_3 + L + Th_3 + L + Ta_3$	
$C_{-d_{33}}$	
$Ta_2 + C_{\bullet a_{33}}$	

Figure 61 - Tableau des listes ambiguës

A l'exception des deux derniers cas, nous pouvons constater à vue d'oeil que ces divers caractères peuvent être séparés dans leur classe par la position relative des points d'intersection des différents tracés par rapport à leurs extrémités.

En effet, les caractères + , T , L de la figure 62 peuvent être séparés dans cette classe car la position du point d'intersection I des deux tracés T_1 et T_2 n'est pas la même d'un caractère à l'autre.

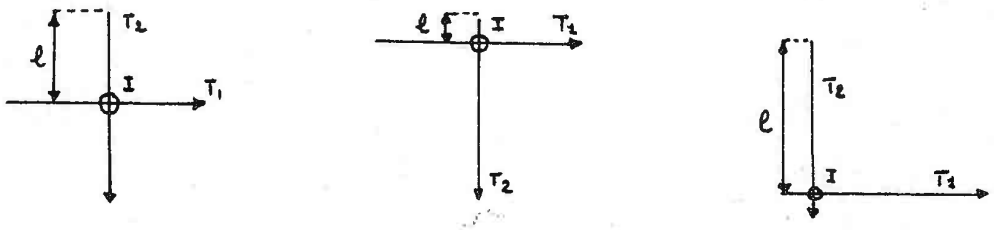


Figure 62 -

I est :	I est :	I est :
- au milieu de T1	- au milieu de T1	- à gauche de T1
- au milieu de T2	- au dessus de T2	- en dessous de T2

Ceci revient à déterminer la position exacte des points d'intersection dans chacun des deux caractères ambigus et comparer les distances correspondantes des points aux extrémités des différents tracés correspondants. La mesure qui donne l'écart le plus grand entre les deux réponses sépare le couple de caractères. La distance retenue pour séparer deux à deux les caractères + , T , L de la figure ci-dessus est la distance l entre le point d'intersection I et l'origine du deuxième tracé T2.

l est :
- moyenne par rapport à la taille du +
- petite par rapport à la taille du T
- grande par rapport à la taille de L

Cependant, la position exacte du point d'intersection des tracés n'est pas toujours facile à déterminer surtout quand ces tracés sont complexes (cas du R et k) Pour éviter le calcul prohibitif de ces positions et pour pouvoir résoudre tous les cas d'ambiguïté (absence de point d'intersection dans les deux dernières classes du tableau), nous remplaçons ces mesures par d'autres beaucoup plus

simples telles les différences en X et Y entre les coordonnées des extrémités des différents tracés.

Ces mesures sont évidemment beaucoup moins précises, mais permettent de séparer totalement les différents caractères ambigus.

N mesures sont à calculer pour chaque couple de caractères à distinguer.

$$N = 2 \times C^{2}_{2xP}$$

↑ Nombre de tracés

P est le nombre de tracés composant le caractère.

Calcul des distances :

Chaque tracé est donné dans le caractère par les trois informations suivantes :

- coordonnées de l'origine : OX_i , OY_i
- coordonnées de l'extrémité : EX_i , EY_i
- cadre ou enveloppe rectangulaire : $MAXX_i$, $MAXY_i$, $MINY_i$, $MINX_i$

i est le numéro d'entrée du tracé dans le caractère.

Pour un caractère formé de deux tracés (1, 2 dans la figure 63), lesdistances à calculer sont :

$d_1 = | OX_1 - EX_1 |$)
$d_2 = | OY_1 - EY_1 |$) 1er tracé
$d_3 = | OX_2 - EX_2 |$)
$d_4 = | OY_2 - EY_2 |$) 2ème tracé

$d_5 = | OX_1 - OX_2 |$)
$d_6 = | OX_1 - EX_2 |$)
$d_7 = | OY_1 - OY_2 |$) 1er + 2ème tracé
$d_8 = | OY_1 - EY_2 |$)

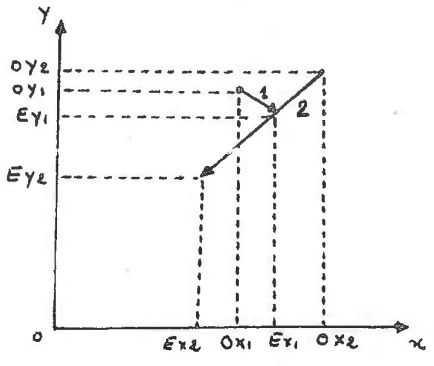


Figure 63

$d_9 = | EX_1 - OX_2 |$)
$d_{10} = | EX_1 - EX_2 |$) 1er et 2ème tracé
$d_{11} = | EY_1 - OY_2 |$) tracé
$d_{12} = | EY_1 - EY_2 |$)

En rangeant chaque tracé dans un vecteur de 8 composantes comme suit :

Ti :

OX_i	OY_i	EX_i	EY_i	$MAXX_i$	$MINX_i$	$MAXY_i$	$MINY_i$

Chaque distance d_k ,k = 1,12 est donnée par les 4 informations suivantes :

d_k =

i	C_i	j	C_j

La distance étant établie entre deux tracés
i , j : numéros des deux tracés
C_i : numéro de la 1ère coordonnée dans le tracé T_i
C_j : numéro de la 2ème coordonnée dans le tracé T_j

Séparation des caractères par le test :

Ces distances sont calculées pour chacun des deux caractères ambigus et comparées chacune au demi-périmètre P de l'enveloppe rectangulaire du caractère qui la contient. Une réponse r_k ($0 \leqslant r_k \leqslant 100$) est affectée à chaque distance d_k comme suit :

$$r_k = \frac{d_k}{P} * 100$$

Le test retenu par le système dans l'arbre de décision est celui qui donne l'écart le plus grand entre deux réponses (r_k , r'_k) obtenues par la même mesure sur le couple de caractères ambigus.

Format du test :

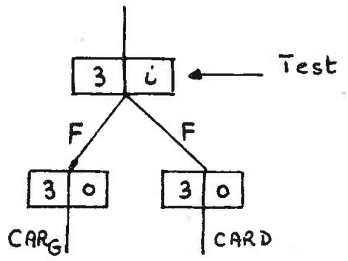
Le test retenu est donné par un vecteur de 6 composantes.

Test i :

ARC		dk		r_k	r'_k	CAR_G	CAR_D

où :

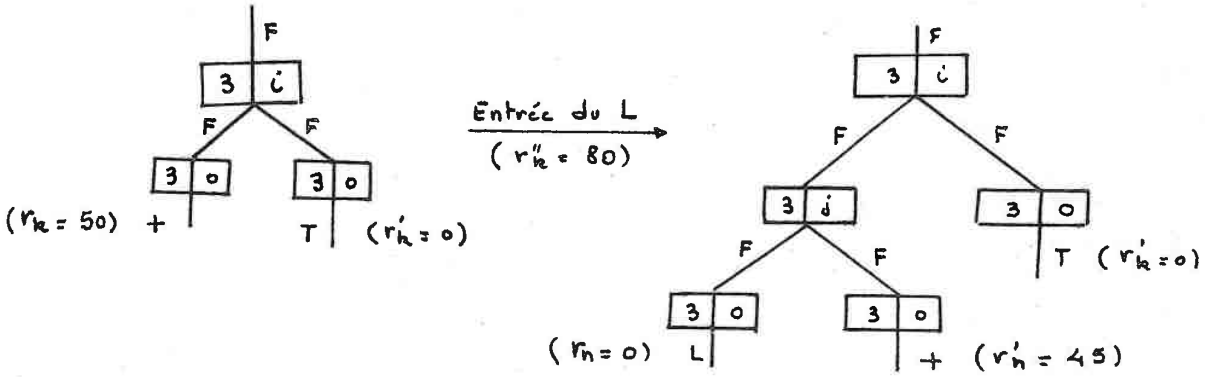
ARC : adresse du test dans l'arbre de décision
d_k : distance discriminante retenue
r_k : réponse obtenue sur le caractère CAR_G
r'_k : réponse obtenue sur le caractère CAR_D
CAR_G : caractère séparé et rangé dans l'arc gauche à partir du noeud
CAR_D : caractère séparé et rangé dans l'arc droit à partir du noeud



Cheminement par le test :

Le cheminement d'un nouveau caractère à partir de ce noeud se fait par consultation du test i et calcul de la distance indiquée (d_k). On détermine la réponse r''_k correspondante et on achemine le caractère dans l'arc de gauche si la réponse r''_k est plus près de r_k et dans l'arc de droite dans le cas contraire.
Un nouveau test peut être cherché au niveau d'une nouvelle feuille entre le caractère entré et le caractère existant s'ils sont différents (voir figure 64).

Un test est dit discriminant si l'écart entre les couples de réponses obtenues à partir de la distance retenue est supérieur à un certain seuil (STEST - 30). Dans le cas où l'écart maximum qu'on trouve est inférieur à ce seuil, on considère que le test n'est pas discriminant et on achemine le nouveau caractère dans les deux arcs quelque soit la réponse qu'il donne.



r_n et r'_n sont les réponses données par le test j sur la distance dn

Figure 64 - Cheminement par le test

Revalorisation du test :

Le test peut être remis en cause chaque fois qu'un nouvel échantillon d'un des deux caractères séparés est introduit en ce noeud. Un nouveau test est calculé pour écarter d'une façon globale deux ensembles d'échantillons différents. L'introduction de nouveaux échantillons peut être faite volontairement par l'écrivain suite à une mauvaise séparation et ceci dans le but de corriger la position du point d'intersection dans lescaractères. La figure 65 donne un exemple de couples d'échantillons descaractères + et T.

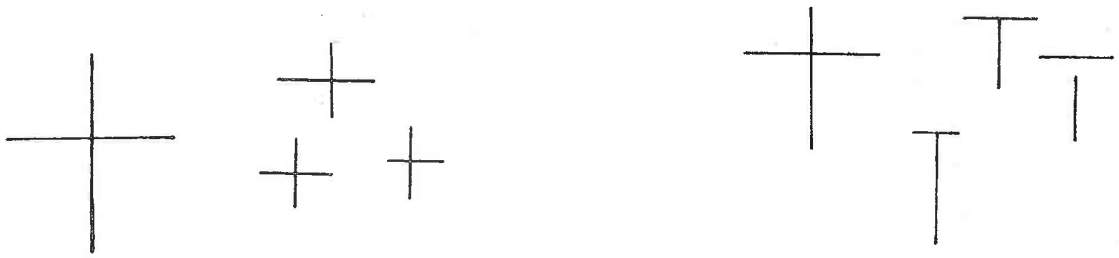
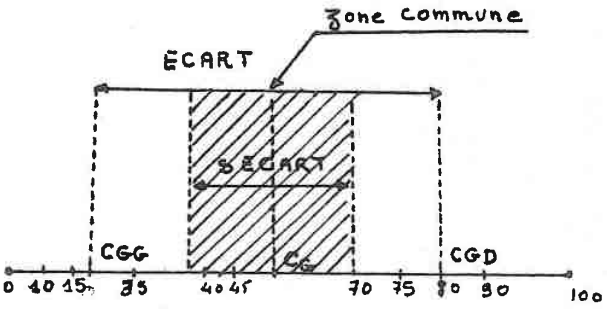


Figure 65 - Couples d'échantillons

Calcul du test global :

Pour chacune des distances d_k on obtient deux ensembles de réponses correspondant chacun à un ensemble d'échantillons.
Pour chaque ensemble, on calcule :
- le centre de gravité de ses réponses (CG_G ou CG_D)
- l'écart entre ses réponses (CR_G ou CR_D)



Le test retenu entre les deux ensembles est celui qui donne :
- l'ECART le plus grand entre les deux centres de gravité (ECART = CG_D - CG_G)
- et un écart (espacement) minimum entre les réponses de chaque ensemble.

Figure 66

Séparation des échantillons :

Les échantillons des deux ensembles de caractères ambigus sont acheminés dans les arcs suivant le même principe que précédemment :

- si le test est discriminant : ECART > SECART (seuil miniumum d'écart)
L'échantillon suit l'arc de gauche si la réponse des tests est à gauche de la zone commune (voir figure 66), et à droite si la réponse du test est à droite de la zone commune. Si la réponse est dans la zone commune (40, 45, 70 : dans la figure 66). L'échantillon suit les deux arcs en même temps et d'autres tests peuvent être calculés dans un niveau plus bas.

- si le test n'est pas discriminant : ECART ≤ SECART
Les deux ensembles d'échantillons sont placés dans deux arcs différents, mais on considère que le test n'est pas suffisant pour les séparer. On attend l'introduction d'un autre échantillon pour revaloriser le test. Si l'échantillon n'a pas été introduit, des réponses multiples sont données par le système à chaque caractère qui passe par ce test.

V.3 - RECONNAISSANCE DES CARACTERES

Le processus de reconnaissance, déclenché par l'utilisateur après un apprentissage suffisant, utilise un arbre de décision pour l'identification des différents caractères entrés. Ces caractères sont introduits depuis la tablette graphique sous forme d'une suite de tracés sans marque de fin pour aucun d'entre eux.

Reconnaître un caractère revient à :
- l'isoler dans la chaine en fixant son début et sa fin
- le segmenter en primitives élaborées
- trouver au moins un chemin qui le décrit par les primitives dans l'arbre de décision ,
- et enfin déterminer un score de reconnaissance en cas de succès.

V.3.1 - Séparation des caractères dans la chaine d'entrée

La séparation des caractères se fait par regroupement des divers tracés connexes dans la chaine d'entrée.

Définition de la connexité :

Une suite de tracés est dite connexe si :

- la suite est réduite à un seul tracé comme dans l'exemple suivant :

. les cadres des tracés s'emboitent, ou bien si
. la distance entre ses cadres est inférieure à un seuil donné (SCON)

Les trois tracés T_1 , T_2 , T_3 représentés dans la figure 67 ci-après sont connexes si :

$$| MX_2 - MMX_1 | < SCON$$
$$| MX_3 - MMX_2 | < SCON$$
$$| MY_2 \quad MMY_3 | < SCON$$

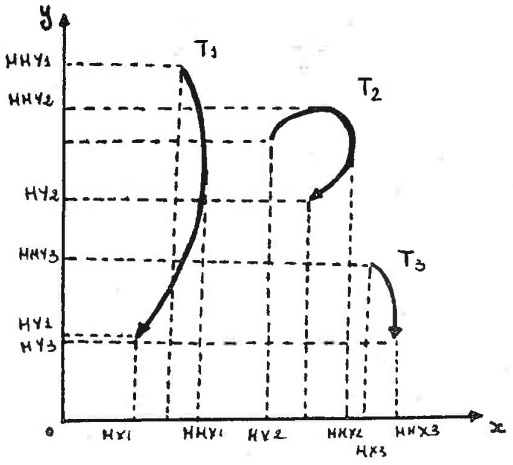


Figure 67 - Tracés du R

Regroupement des tracés connexes :

Chaque caractère de la chaine écrite est censé être formé d'une suite de tracés connexes séparés des autres caractères par un espace assez grand.
Une suite de caractères parfaitement isolables ressemblerait à cet exemple.

avec di ⩾ SCON

Or, la plupart des textes écrits ne remplissent pas ces conditions, soit parce que l'écrivain, en écrivant rapidement colle les caractères dans la chaine, soit parce qu'on se trouve des fois obligé de le faire en écrivant certaines formules mathématiques.

Exemples :

Les cadres de A , + et B sont à l'intérieur du cadre du symbole racine carrée. Toute la formule est prise pour un seul ensemble connexe. Il s'agit de séparer ces divers caractères par la suite à la reconnaissance. Nous reviendrons par la suite sur le processus de séparation.

Les tracés sont dessinés un à un et rangés dans un tableau. Le regroupement des tracés connexes se fait indépendamment de leur ordre d'entrée en comparant la distance entre leur cadre au seuil minimum. Ce regroupement se fait à la fin de la saisie car l'écrivain peut très bien revenir sur un caractère pour le compléter.

Exemple :

On attend que toute la chaine soit écrite puis on procède à la séparation des tracés connexes.

V.3.2 - Reconnaissance des caractères par comparaison

Les divers tracés connexes sont regroupés, segmentés un à un par l'automate d'extraction et rangés dans un arbre de primitives (tableau TRACAR). Chaque liste de primitives est comparée à tous les chemins de l'arbre de décision.

Deux résultats sont donc possibles :

. Succès

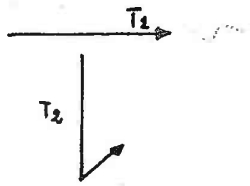
Le processus de reconnaissance a réussi à parcourir la liste jusqu'à une feuille. Le caractère reconnu est celui qui se trouve en cette feuille. Un score de reconnaissance est calculé pour ce caractère et le processus de reconnaissance remonte dans les deux arbres (arbre du caractère et arbre de décision) pour un nouveau cheminement. Plusieurs réponses sont donc possibles à ce niveau. Chaque chemin peut aboutir à un caractère (non nécessairement le même).

Dans le cas où on reconnait deux fois le même caractère, on ne retient que celui qui a été donné avec le meilleur taux de reconnaissance.

. Echec

On dit que la recherche a échoué si à une primitive de la liste en cours, ne correspond aucun cheminement possible dans l'arbre de décision . Cet échec peut être interprété de trois manières :

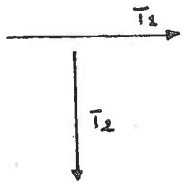
- Le caractère a été mal dessiné : des bavures peuvent entacher le caractère ou des déformations multiples viennent fausser la segmentation ; à la liste de primitives ne correspond aucun chemin dans l'arbre de décision .



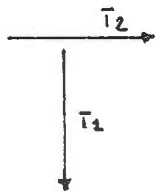
Code : $Ta_3 + L + Tg_3 + Tc_1$

Ce caractère est souvent refusé, car son apprentissage n'offre aucun intérêt. Il est impossible d'avoir deux fois de suite ce même code.

- Un nouvel écrivain a introduit une nouvelle écriture. Le modèle du caractère entré ne correspond pas au modèle appris.



modèle appris



modèle entré

L'ordre d'écriture des tracés n'est pas le même que celui du caractère appris d'où incompatibilité à la description :

$\neq$ → $- Ta_3 + L + Tg_3$

→ $- Tg_3 + L + Ta_3$

- La liste de primitives est trop longue (on arrive à la fin d'un chemin sans avoir parcouru toute la liste). Car le cheminement continue chaque fois que la comparaison est possible. La figure 68 ci-après donne un exemple où la liste de primitives est trop longue.

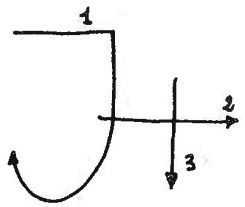


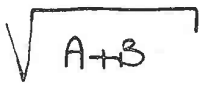
Figure 68 - Code : $Ta_1 + Tg_2 + C_{c_{12}} + L + Ta_3 + L + Tg_3$

On suppose dans ce cas que l'ensemble connexe regroupe plusieurs caractères. Le processus de reconnaissance remonte dans le chemin de l'arbre de décision jusqu'au dernier lever de crayon et cherche une fin de chemin.

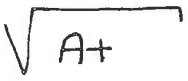
- en cas de succès, on considère qu'on a été trop loin dans l'arbre, on isole la première partie acheminée, soit ici le (J) et on recommence à partir de la racine de l'arbre de l'acheminement du groupe de tracés restant, soit ici le (+).

- en cas d'échec : le processus de reconnaissance le signale à l'écrivain qui décide de son apprentissage ou non. Car il peut arriver qu'un chemin soit trop long parce qu'on n'a pas appris le caractère.

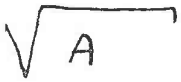
Exemples :



échec



échec



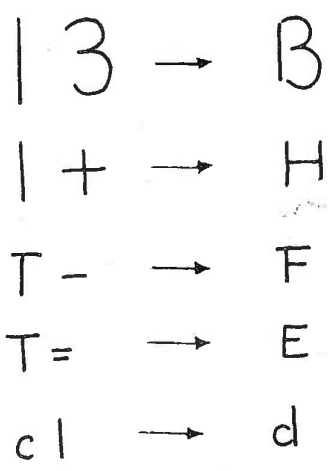
échec


succès : ✓

A+B	:	échec	+B	:	échec
A+	:	échec	+	:	succès (+)
A	:	succès (A)	B	:	succès (B)

Inconvénient :

Ce procédé peut introduire quelquefois des ambiguités : une suite de caractères différents et très rapprochés regroupés par la procédure de connexité peut être reconnue comme un seul caractère (voir les exemples de la figure ci-dessous) :



V.3.3 - Calcul du score de reconnaissance

Du fait des segmentations multiples réalisées par l'automate d'extraction sur les différents tracés du caractère à reconnaître, plusieurs réponses sont possibles. Pour fixer notre choix sur les meilleures réponses données, on établit un score de reconnaissance pour chaque caractère reconnu et on ne retient que ceux qui ont un score satisfaisant.

Ce score est donné par :

- un score de segmentation déjà calculé par l'automate d'extraction pour chaque tracé. Le score total sera une moyenne de ces scores.
  Il donne une estimation de la représentativité de la liste de primitives trouvées.

- un score de test : calculé par une moyenne des différents scores, établis lors du passage du caractère par des tests de lever d'ambiguité.

$$ST = \frac{\sum_{i=1}^{n} st_i}{n}$$ n est le nombre de tests parcourus

. On affecte un score de test élevé si le caractère n'a parcouru aucun test dans l'arbre (unicité de la représentation en primitives),

. Et on affecte un score faible au caractère qui a subi beaucoup de tests lors du parcours de sa liste dans l'arbre, car sa reconnaissance peut ne pas être très sure.

Calcul du score : st_i

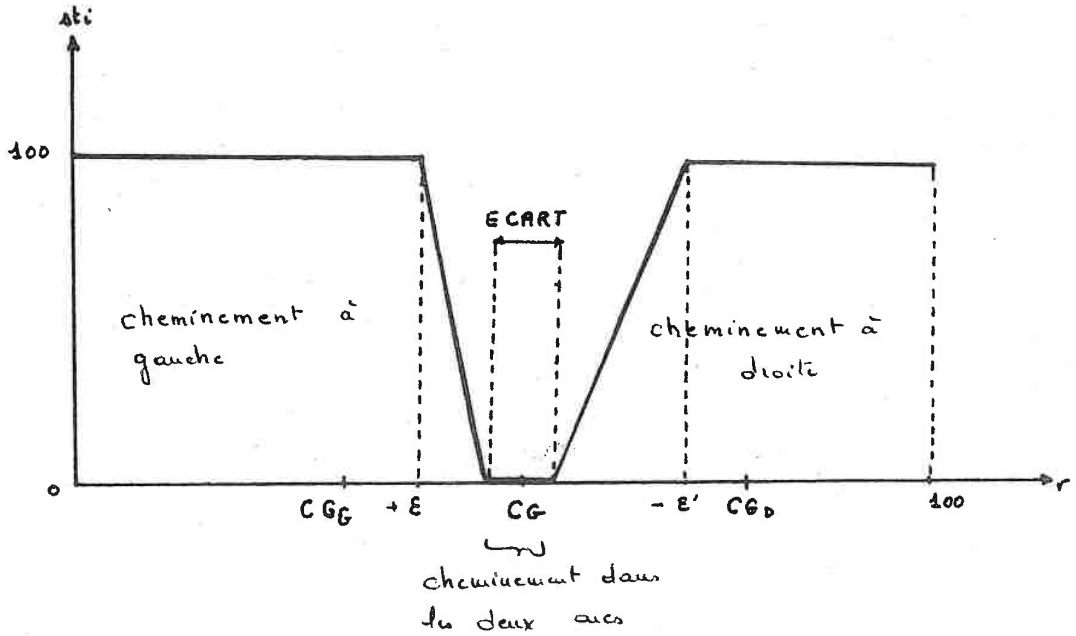
Après parcours de toute la liste de primitives du caractère, il s'agit de trouver un caractère correspondant à la liste entrée. La recherche se fait par application de testset parcours dans les feuilles d'après la réponse (r) obtenue (comme à l'apprentissage) et ceci jusqu'à ce qu'on trouve un test nul (ou absence de test).

Le score st_i est calculé à chaque passage par un noeud de la manière suivante :

$st_i = 100$ si il n'y a pas de test : on a atteint une feuille

$st_i = 0$ si le test existant n'est pas discriminant, c'est-à-dire si l'écart entre les réponses trouvées lors de l'apprentissage n'est pas assez grand.
dans ce cas, on achemine la liste dans le sous-arbre à partir de ce noeud sans lui appliquer de test.

$0 < st_i < 100$ si le test existant est discriminant
la fonction score est donnée par le graphe de la figure 69.
Il dépend de la réponse donnée par le test sur la liste entrée.

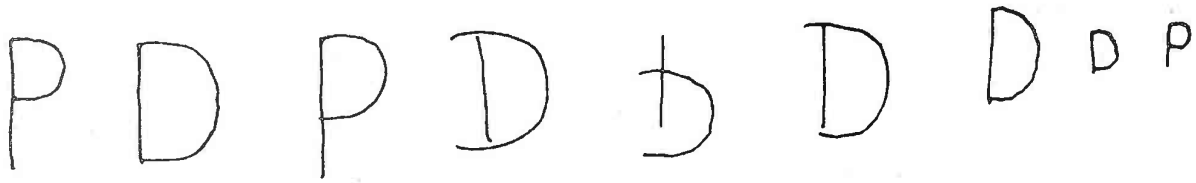


ε et ε' sont calculés en fonction de la valeur de l'écart (espacement entre les réponses) dans les ensembles de réponses.

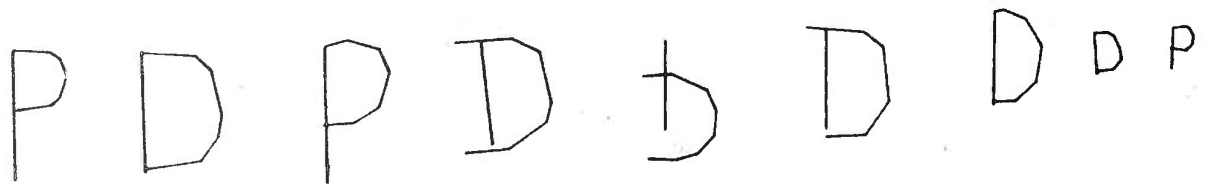
Figure 69 - Calcul du score test en fonction de la réponse donnée.

RESULTATS DE LA RECONNAISSANCE

1 2 3 4 5 6 7 8 9 0



RESULTATS DE LA RECONNAISSANCE



P D ? D D D D D P

P

V.4 - UTILISATION DU SYSTEME

L'idée de mettre au point un système de reconnaissance multi-écrivainsavec un apprentissage continu au cours du traitement, nous a conduit à élaborer dans ce sens une version entièrement conversationnelle. Son utilisation est assez souple, permettant à chaque écrivain d'intervenir directement pour contrôler et diriger le déroulement des opérations et d'ajuster certains paramètres de travail nécessaires à l'identification de son écriture, sans pour autant perturber le fonctionnement du programme.
La figure 70 donne le schéma du dispositif de traitement en fonction des commandes enregistrées.

Etant dépouvu de toute forme d'initiative, le système interroge constamment l'écrivain à chaque étape de l'analyse pour lui demander l'action suivante à entreprendre. L'intervention de l'écrivain se fait à deux niveaux différents du système à l'aide de commandes spéciales :

- au niveau initial

On distingue quatre commandes différentes :

1) Commande : P (Paramètres)

Cette commande actionne une procédure spéciale de rectification immédiate de paramètre de travail. L'écrivain prend compte des valeurs courantes de ce paramètre et les ajuste de façon à ce qu'ils s'adaptent bien à son écriture. Une fois corrigées, ces valeurs ne varient plus jusqu'à la fin du traitement.

On distingue trois familles de paramètres correspondant chacune à une étape différente de l'analyse :

- échantillonnage
  - SAECH : angle minimum
  - SLONG : longueur minimale
  - SBAV : seuil de bavures

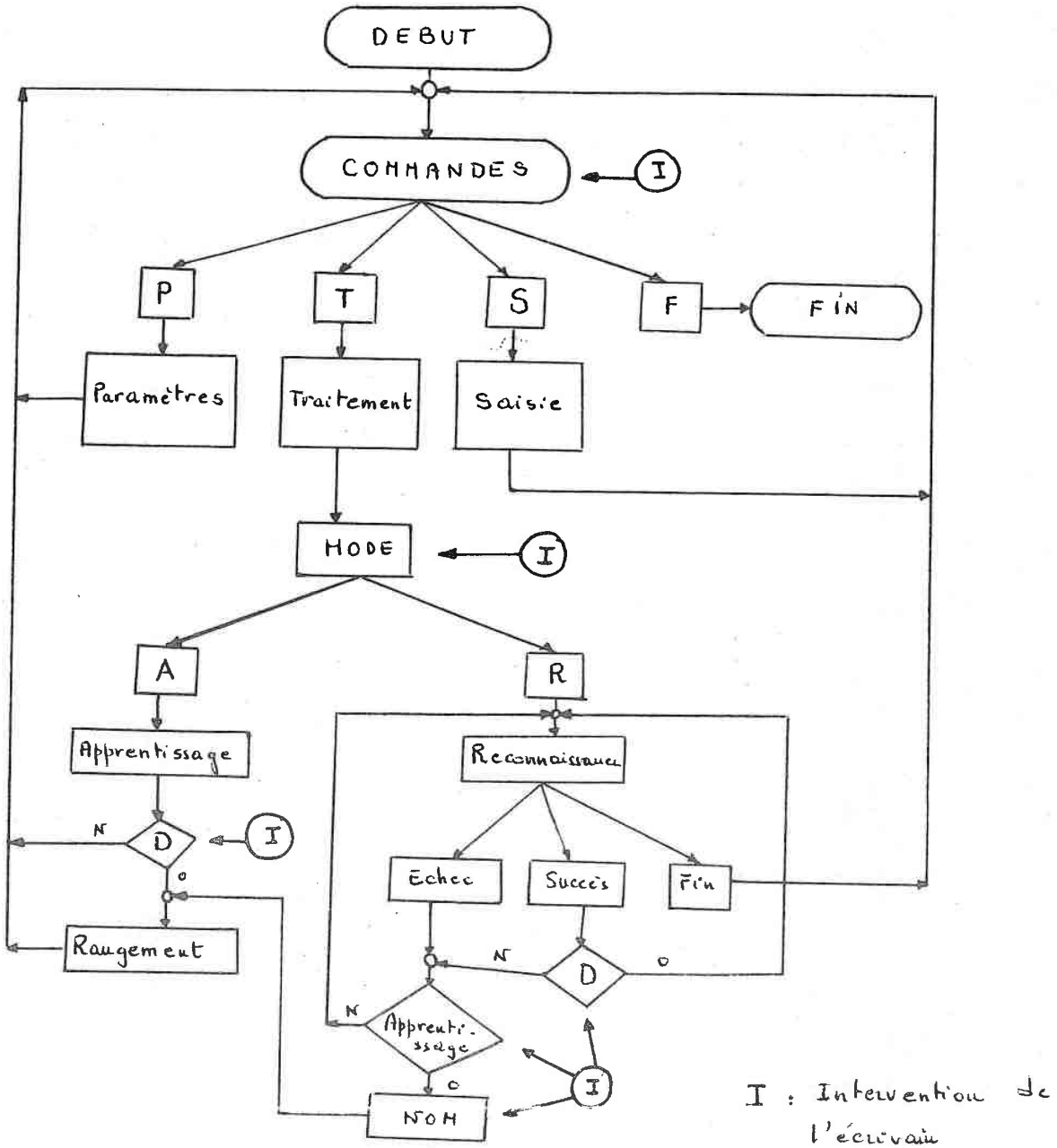


Figure 70 - Schéma de commandes du système

- Segmentation

  . Segmentation brute (ces paramètres interviennent dans les tests de l'automate d'extraction)

  SCOIN : seuil de coin

  SLONE : égalité entre deux longueurs

  SANE : égalité entre deux angles

  . Classification

  TETA1) directions des traits
  TETA2)

  TETA3 directions d'arcs de cercle

- Traitement

  . CSEL : nombre de chemins à sélectionner

  . SSEL : score minimum de sélection

  . SCON : seuil de connexité

  . SREC : score minimum de reconnaissance

2) Commande : S (Saisie)

Cette commande connecte la tablette pour la saisie directe du caractère. La saisie se fait de manière différente suivant qu'on veut faire de l'apprentissage ou de la reconnaissance :
A l'apprentissage, l'écrivain introduit un seul caractère et précise son nom en pointant le stylo dans une zone correspondante de la tablette.
A la reconnaissance, l'écrivain introduit un ensemble de caractères sans précision aucune pour chacun d'entre eux.
La saisie peut être relancée par l'écrivain chaque fois que des parasites ou des déformations accentuées entachent le caractère, car son traitement risque d'être hasardeux.

3) Commande : T (Traitement)

La procédure de traitement n'est activée par l'écrivain qu'après vérification de la saisie. Elle consiste à segmenter les différents tracés du ou des caractères entrés et les préparer pour le traitement ultérieur : l'écrivain précise le mode de traitement qu'il faut faire par la suite :

A : Apprentissage

R : Reconnaissance

chacune de ces commandes active la procédure correspondante.

4) Commande : F (Fin)

Cette commande est envoyée en fin de traitement pour désactiver la tablette graphique et arrêter le programme. L'arbre de décision ainsi que le tableau des tests sont sauvegardés sur disque .

- Au niveau du traitement :

L'écrivain intervient :

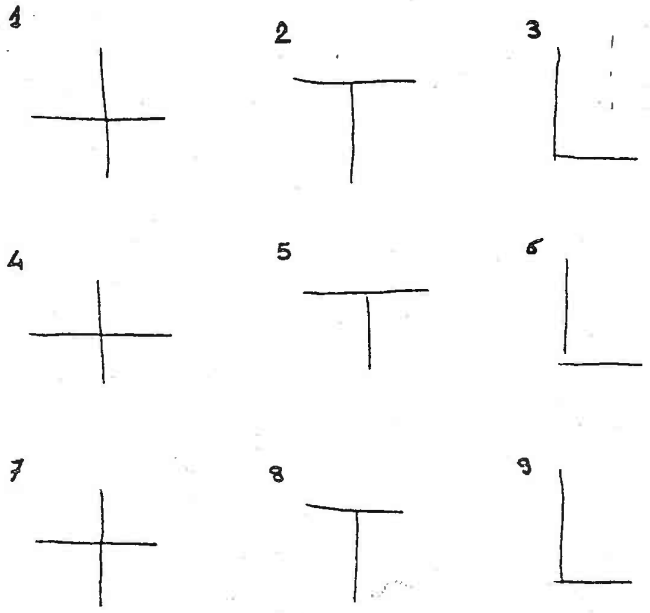
- à l'apprentissage :
  Il contrôle la bonne segmentation du caractère et s'assure bien du nom qu'on lui a attribué à la saisie , car un apprentissage erroné peut fausser la reconnaissance.
  En cas de non ambiguïté, il ordonne au système de l'apprendre.

- à la reconnaissance
  En cas d'échec, l'écrivain examine la liste de primitives du caractère et peut ordonner au système de l'apprendre si la liste correspond à son dessin. Il lui affecte un nom qu'il introduit directement au clavier.
  En cas de succès, le système peut donner un nom ambigu, c'est-à-dire qui ne correspond pas au caractère entré.

entré reconnu rectifié

L'écrivain rectifie le nom du caractère et l'envoie au module d'apprentissage.

En cas d'apprentissage continu réalisé à ce niveau, la procédure de traitement peut être relancée une deuxième fois pour une vérification supplémentaire.




```
1   ECHEC...LISTE DES PRIMITIVES(O/N)
     PRIMITIVES DU CARACTERE:  ?
                                DIRECTION      LONGUEUR
                    TRAIT           E              G
                    LEVER    SCORE= 99
                    TRAIT           S              G
                    LEVER    SCORE= 99
    OU BIEN
                    TRAIT           E              G
                    LEVER    SCORE= 99
                    TRAIT           SE             G
                    LEVER    SCORE= 82
    VOULEZ-VOUS L'APPRENDRE (O/N)O
                             NOM? +
2   SUCCES  CARACTERE:  +   S.CHEMIN: 99   S.TESTS: 99
                                                (O/N) N
    ECHEC...LISTE DES PRIMITIVES(O/N)
    VOULEZ-VOUS L'APPRENDRE (O/N)O
                             NOM? T
3   SUCCES  CARACTERE:  +   S.CHEMIN: 92   S.TESTS:  7
                                                (O/N) N
    ECHEC...LISTE DES PRIMITIVES(O/N)
    VOULEZ-VOUS L'APPRENDRE (O/N)O
                             NOM? L
4   SUCCES  CARACTERE:  +   S.CHEMIN: 99   S.TESTS: 95
                                                (O/N) O
5   SUCCES  CARACTERE:  T   S.CHEMIN: 99   S.TESTS: 98
       OU BIEN      :  +   S.CHEMIN: 90   S.TESTS: 99
                                                (O/N) N
    ECHEC...LISTE DES PRIMITIVES(O/N)
    VOULEZ-VOUS L'APPRENDRE (O/N)O
                             NOM? T
6   SUCCES  CARACTERE:  L   S.CHEMIN: 99   S.TESTS: 53
                                                (O/N) O
7   SUCCES  CARACTERE:  +   S.CHEMIN: 99   S.TESTS: 91
                                                (O/N) O
8   SUCCES  CARACTERE:  T   S.CHEMIN: 99   S.TESTS: 87
                                                (O/N) O
9   SUCCES  CARACTERE:  L   S.CHEMIN: 99   S.TESTS: 42
                                                (O/N) O
```

V.5 - PERFORMANCES ET RESULTATS

Implantation du système :

La version actuelle du système de reconnaissance des caractères a été écrite en FORTRAN IV (3500 instructions) et occupe une taille mémoire d'environ 16 K mots de 16 bits. Son implantation sur MITRA 125 de taille réduite nous a conduit à effectuer du recouvrement, ce qui ralentit de beaucoup le temps d'exécution à la reconnaissance.
D'autre part, vu le mode conversationnel qu'on a adopté dans cette version (interrogation au cours même de la reconnaissance, affichage sur écran ...), il nous est impossible d'évaluer exactement le temps de reconnaissance. On s'est plutôt intéressé aux résultats de la reconnaissance.

Résultats de la reconnaissance :

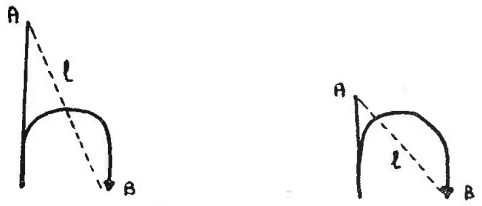
Nous avons testé le système sur un jeu de 35 caractères différents, mais choisis de telle manière qu'on puisse avoir une idée précise sur la performance. La matrice de confusion de la figure 71 donne les caractères entrés avec leur mode d'écriture (sens d'écriture précisé par les flèches) et le taux de reconnaissance obtenu pour chacun d'entre eux.
On précise dans la première colonne du tableau le nombre d'échantillons appris par caractère. Ce nombre a été déterminé dans une première phase en rentrant 30 acquisitions par caractère. Chaque fois qu'on tombe en échec sur un échantillon, on ordonne, dans ce cas, au système de l'apprendre.

Pour déterminer le taux de reconnaissance exact du système, nous avons traité 100 acquisitions pour chaque caractère. Ces caractères ont été rentrés par un seul écrivain de la façon la plus naturelle possible (sans application particulière).

Les résultats obtenus sont :

- taux de reconnaissance : 93 %
- taux de confusion : 2 %
- taux de rejet : 5 %

Le plus fort taux de confusion vient essentiellement des caractères h et n : 0,7 %



La distance l = |AB| qui a été retenue pour séparer les deux caractères, n'est pas assez discriminante. Le rapport distance/cadre du tracé est le même pour les deux caractères (l est moyennement grande par rapport au cadre de chaque caractère), d'où la faiblesse du test et la confusion obtenue.

Explication du rejet :

Le rejet est le résultat de deux phénomènes :

- bavures

Des bavures entachent les tracés des caractères à chaque lever de crayon .

Leur taille est relative à la vitesse du lever. Il est donc impossible de les supprimer dans tous les cas. Un seuil de longueur minimum a été imposé et on arrive actuellement à supprimer les bavures de petites tailles. Nous ne pouvons donner un seuil plus élevé car celà risque de supprimer des parties du tracé comme montrées par la figure ci-dessous.

- variation de l'écriture

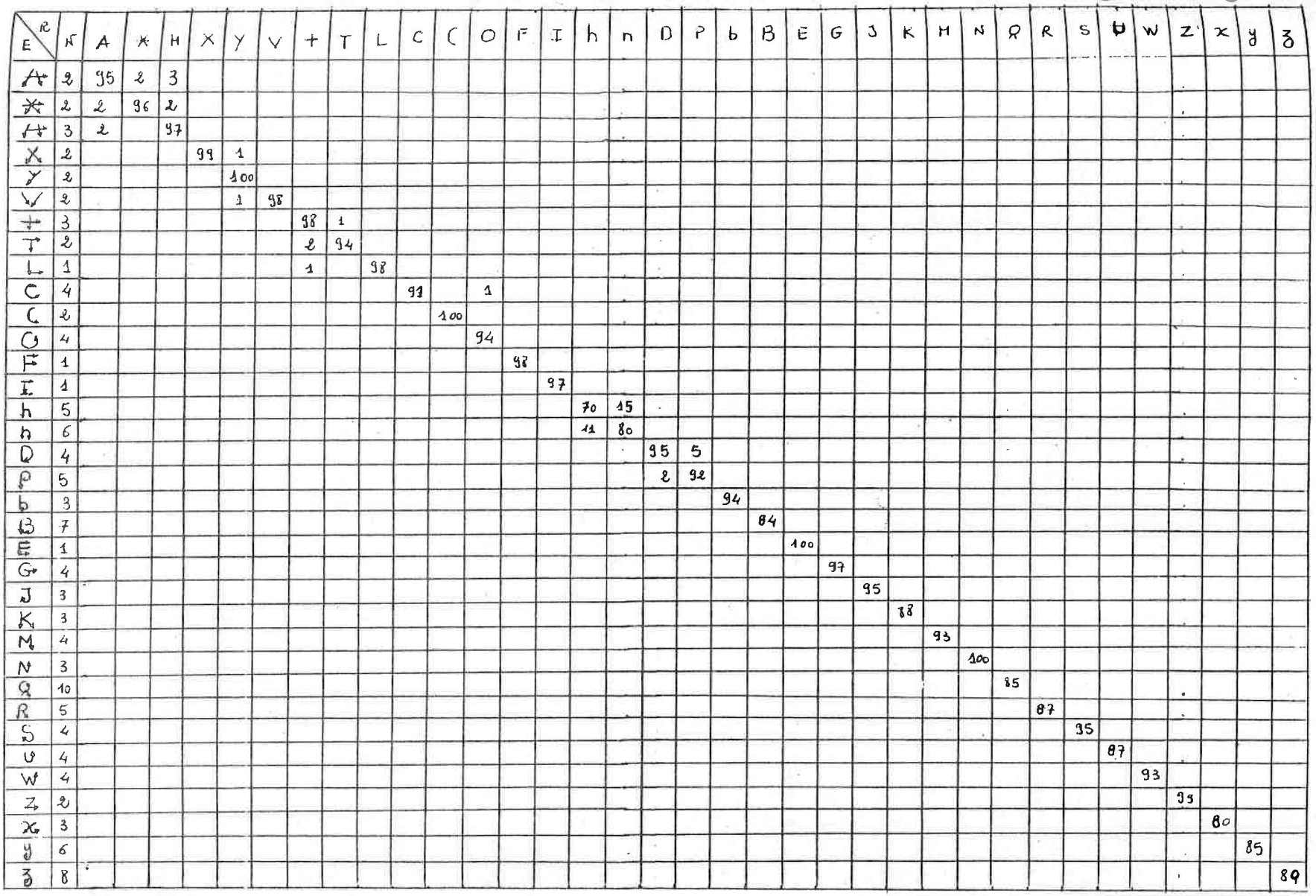
L'écriture manuscrite varie énormément. Il est impossible de prévoir toutes ces variations à l'apprentissage, car cela accroitrait énormément la taille du dictionnaire. La figure 71b donne les quelques variations qu'on a aperçues à l'écri-

ture du M. Nous pensons que cela provient aussi de la sensibilité de l'algorithme de segmentation aux variations du tracé malgré les larges marges de tolérance qu'on a introduites lors des comparaisons. Cela est dû au fait que les paramètres de segmentation sont très généraux. Ceci explique bien le faible taux de reconnaissance obtenu pour les caractères b , Q , L , u , y , z malgré le nombre important d'échantillons introduits à l'apprentissage pour chacun d'entre eux. Il faut revenir là-dessus pour les adapter plus à l'écriture manuscrite. BERTHOD ne se heurte pas à ce genre de difficultés parce qu'il fait une segmentation moins fine.

Figure 71b -Différents échantillonnages réalisés pour la même écriture

Nous dirons en conclusion que les performances du système sont assez moyennes ; il est donc nécessaire d'élargir le contexte et étudier les caractères manuscrits dans un cadre plus général. C'est ce qu'on a essayé de faire par l'analyse syntaxique des formules mathématiques, et c'est ce qu'on va voir dans le chapitre suivant.

Le but initial de ce travail était de fabriquer un outil de segmentation de dessins. Son utilisation pour la segmentation des caractères a servi simplement de test, ce qui explique les performances moyennes obtenues.



E = caractères entrées
R = caractères reconnus
N = nombre d'échantillons appris

Figure 71 - Matrice de confusion

V.6 - CONCLUSION

Nous avons exposé dans ce chapitre un procédé original d'apprentissage et de reconnaissance de caractères manuscrits segmentés par un automate d'extraction en plusieurs listes de primitives élaborées.

Le schéma de l'algorithme suit un arbre de décision regroupant au départ un noyau réduit de caractères et complété par un apprentissage continu au cours de la reconnaissance. Des tests de lever d'ambiguité sont introduits au niveau des feuilles pour séparer les représentations identiques.

Chaque chemin de l'arbre représente une succession de primitives correspondant à un seul caractère. Le passage de l'algorithme d'un arc à un autre se fait par comparaison du code de la primitive ou par la réponse d'un test.

A la reconnaissance, l'algorithme compare toutes les listes de primitives du caractère entré à celles de l'arbre et ne retient que celles qui ont abouti à une feuille. Un score de reconnaissance est affecté à chaque réponse.

Cependant, malgré la finesse de la segmentation et la robustesse des tests introduits dans l'arbre, notre programme peut commettre des erreurs telles :

- mauvaise reconnaissance,
- reconnaissance multiple et ambiguë

La sensibilité de la segmentation (voir le cas du M précédent) conduit à un taux de rejet important.

Hormis le premier type d'erreur, nous avons essayé de résoudre le deuxième type par analyse de ces réponses dans des formes beaucoup plus contextuelles telles les formules mathématiques. Un analyseur syntaxique aidé d'une grammaire de coordonnées essaie de reconstruire la formule par choix des meilleures réponses. Les terminaux de la grammaire sont ici les caractères reconnus par notre système. L'analyse syntaxique des formules fera l'objet du chapitre suivant.

L'approche qu'on a décrite ici se caractérise néanmoins par la finesse du découpage en primitives et permet à priori une adaptation à n'importe quel graphisme (alphabet latin : majuscules, minuscules, chiffres arabes, alphabet arabe ..., ou bien dessin) ; mais cette finesse de segmentation est malheureusement trop sensible à une mauvaise écriture, d'où les inconvénients signalés ci-dessus.

CHAPITRE 6

RECONNAISSANCE DES FORMULES MATHEMATIQUES

VI.1 - INTRODUCTION

La méthode structurelle décrite précédemment nous a fournit des réponses multiples d'ambiguïté. C'est pourquoi il est indispensable d'intégrer cette reconnaissance dans un niveau structurel plus élevé où le caractère manuscrit tient le rôle de composant syntaxique élémentaire. Cette extension permettra dans certains cas de choisir la réponse correcte en fonction des informations syntaxiques fournies par ce niveau supérieur.

Notre choix a été porté ici sur les formules mathématiques bidimensionnelles dont les caractéristiques fondamentales peuvent être résumées dans les trois points suivants :

a) possibilité de pouvoir traiter dans la même formule tout le jeu de caractères dont on dispose (caractères alphanumériques, opérateurs et signes),

b) sans avoir recours à un vocabulaire énorme, pouvoir décider du bon choix des caractères, par simple analyse syntaxique de la formule,

c) graphisme comportant une réelle structure, ce qui nécessite une approche telle que nous l'avons choisie.

VI.1.1 - Réflexions sur la méthode d'analyse

Nous considérons une formule mathématique comme un assemblage cohérent de variables et d'opérateurs liés entre eux par des relations topographiques de type position et alignement. Analyser syntaxiquement une formule revient donc à retrouver ces relations et la départager en sous-expressions.

La méthode d'analyse syntaxique choisie relève un peu de cette constatation. Son principe consiste à localiser l'opérateur "central" de l'expression à analyser, délimiter les zones de ses opérandes et attribuer un nom à ce partage (non terminal donné par la règle de description de la grammaire associée).

Cette analyse est ensuite réitérée sur les opérandes jusqu'aux éléments termi-

naux (identificateurs). Si le non-terminal trouvé ne correspond pas au but recherché, on choisit, à l'aide du contexte, la règle qui contient ce non-terminal et on réitère l'analyse sur les opérandes trouvés, et ainsi de suite jusqu'à atteinte du but (axiome de la grammaire).

Cette méthode nous éloigne un peu des méthodes classiques dites ascendantes ou descendantes appliquées généralement sur des chaines linéaires (langages naturels, langages de programmation, concaténation d'éléments simples) et qui analysent la chaine de gauche à droite.

Pour nous, l'élément de départ sera choisi, au milieu de la formule, parmi les opérateurs en fonction de son importance (priorité).

Cette méthode, dite ascendante-descendante, tient ses ressources d'une méthode beaucoup plus générale décrite par R. MOHR et J.P. HATON (46). Elle fut utilisée avec succès dans notre laboratoire par G. MASINI (42) pour l'interprétation de plusieurs modèles de dessins, et ensuite par J.F. MARI (47) pour la compréhension du discours parlé. Nous reviendrons dans ce chapitre sur son application aux formules mathématiques avec plus de détails.

La description de la méthode ainsi que le fonctionnement de l'algorithme d'analyse seront précédés par un aperçu sur les diverses ambiguités observées à la reconnaissance des caractères et par une description détaillée du langage des formules traitées.

VI.1.2 - Analyse des ambiguités observées à la reconnaissance des caractères

Comme nous l'avons vu précédemment, le système n'est pas toujours capable de séparer les caractères (mauvaise écriture ou apprentissage incomplet). La figure 72 donne un exemple de ces résultats obtenus réellement sur une expression arithmétique. Nous allons voir comment on peut résoudre de telles ambiguités par l'analyse syntaxique des formules mathématiques.

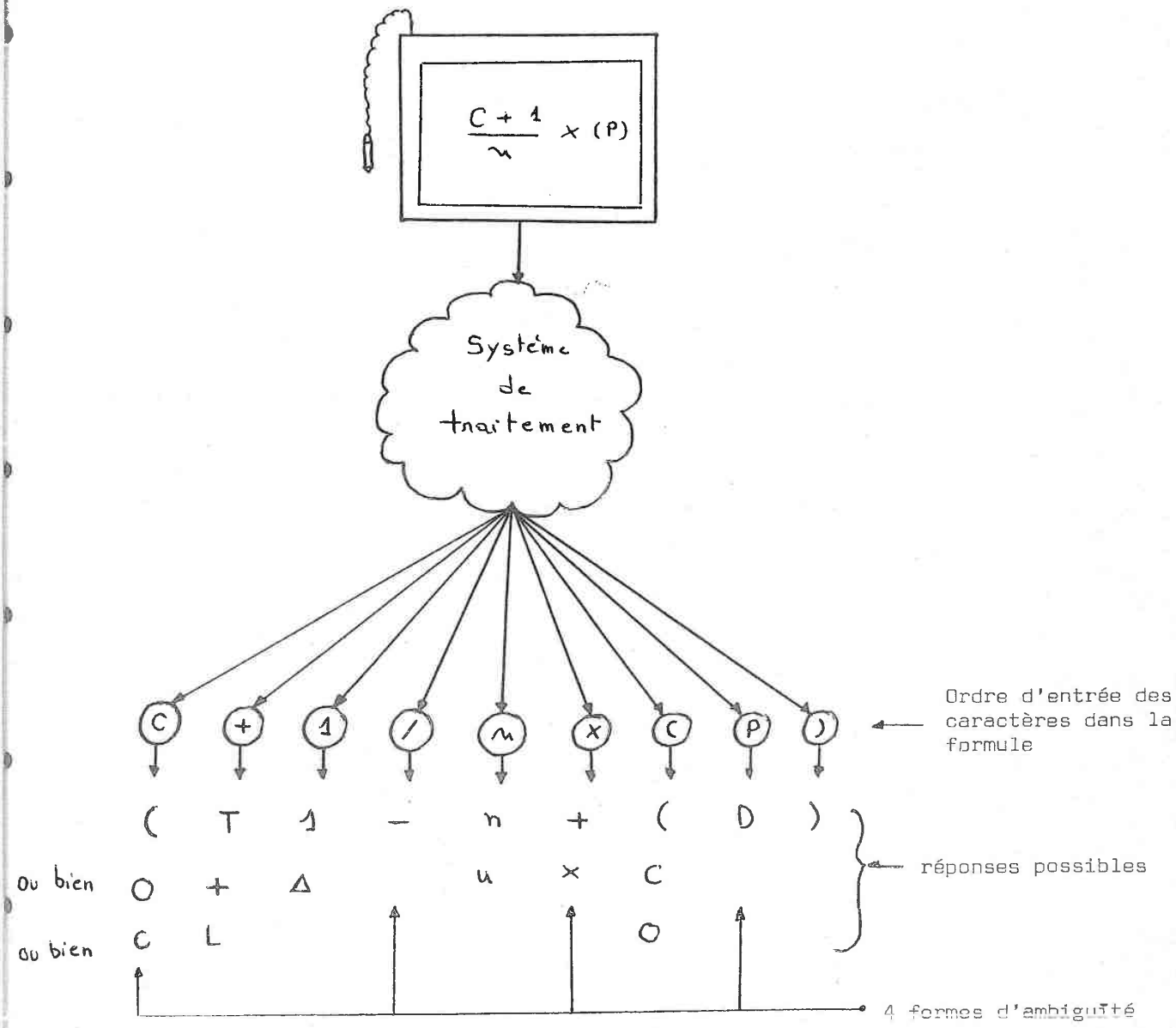


Figure 72 - Ambiguïtés observées à la reconnaissance des caractères

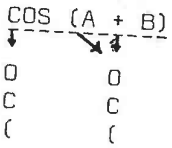
Nous allons montrer dans quel contexte, nous pourrons résoudre chaque forme d'ambiguité.

Ambiguités :

1) Réponses multiples

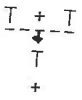
a) C → O
C
(

Contextes favorables



La syntaxe de l'expression trigonométrique nous amène à choisir le C dans la première liste et la parenthèse dans la deuxième

b) + → L
T
+

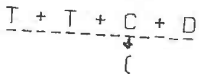


On choisit le + si on considère qu'une variable est réduite à une seule lettre (TTT ne peut pas être une variable, elles sont réduites à 1 caractère).

2) Mauvaise reconnaissance

Cas où le caractère d'origine est reconnu avec un autre nom.

a) C → (



La parenthèse ouvrante sera refusée car il manque la parenthèse fermante correspondante.

b) P → D

La distinction entre le P et le D ne peut être faite que si l'on sait que l'une des deux lettres ne désigne pas un identificateur.

3) Confusion

a) X → +

b) S → 5

$$\frac{T + T}{+} \quad \Big) \quad \Big) \quad \Big) \quad \Big) \quad \Big) \quad \Big) \quad \text{décision impossible, car les deux caractères de la liste sont possibles}$$

X

$$\frac{S + T + S}{5} \quad \Big) \quad \Big)$$

S

4) Cas particulier

$$\frac{T + T}{C - D} \quad \Big\} \text{ contextes}$$

Le trait de fraction est donné par le système de reconnaissance comme un signe -. Cette distinction ne peut être faite que par le contexte : présence d'une expression au dessus et au dessous du symbole.

VI.2 - DESCRIPTION DU LANGAGE DES FORMULES

Les remarques faites précédemment ont mis en évidence l'importance du contexte pour la reconnaissance. Nous allons ici définir le langage des formules afin d'avoir un support formel qui nous servira à déterminer les contextes possibles et comment ils peuvent intervenir dans les différents choix.

Ce langage est nettement plus restreint que celui d'ANDERSON (48) pour une application analogue. Rien n'interdit de reprendre avec nos idées son langage complet (avec exponentiation, intégrale , sommation, ...), mais le système qui en résulterait serait évidemment bien plus lourd.

Structure d'une formule :

Nous considérons une formule mathématique comme un assemblage cohérent de composants syntaxiques comprenant :

- au niveau élémentaire :

  . les caractères alphabétiques (majuscules, minuscules)
  . les chiffres arabes
  . quelques opérateurs arithmétiques + , - , × , / et - Unaire
  . et les parenthèses.

- au niveau syntaxique :

  . les variables (réduites à une lettre)
  . les constantes numériques (suite de chiffres)
  . des expressions arithmétiques linéaires (parenthésées ou non)
  . des fonctions trigonométriques (SIN, COS, TAN)
  . et des fractions.

La figure 73 donne un exemple général du modèle de formules qu'on se propose de traiter.

$$- 10 + A \times \frac{\frac{(C - D) \times A}{(C)}}{\frac{a}{b} + \frac{c}{d}} - 1532 \times \left(\frac{SIN(A) + COS(B)}{TAN\ (c + d)} - 1 \right)$$

Figure 73 - Exemple de formule

Format des données :

Du fait du mode d'entrée des caractères et de l'importance de la place qu'occupe chacun d'eux dans la formule, ces caractères sont présentés à l'analyseur avec les informations suivantes :

. Nom (A, B, + , × , ...)

. Position dans la formule
  - ordre d'entrée (1, 2 ...)
  - position dans la liste des réponses (les caractères sont classés par le score de reconnaissance)

. Position dans l'espace de la tablette
  Cadre : XMIN, YMIN, XMAX, YMAX

. Position dans l'espace de la formule
  Centre : Xcentre, Ycentre

Le centre du caractère est relatif à un quadrillage fait au préalable sur l'espace de la formule afin d'avoir une large marge de tolérance sur l'emplacement du caractère. X centre et Y centre seront donnés par les numéros de l'horizontale et la verticale les plus proches du centre réel du caractère $\left(\frac{XMIN + XMAX}{2}, \frac{YMIN + YMAX}{2} \right)$. La figure 74 a en donne un exemple.

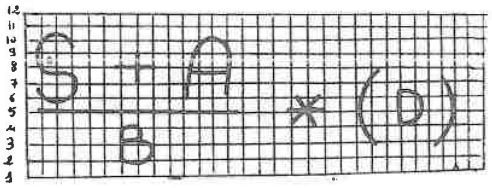


Figure 74 a - S centre : Xcentre = 8(± 1)
Ycentre = 2(± 1)

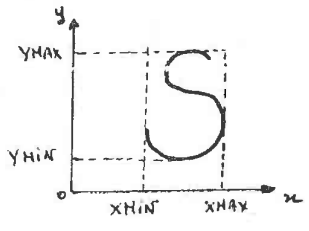


Figure 74 b - Cadre du S

La formule sera donnée avec ces annotations dans deux tableaux : CARAC et CADRES :

CARAC

Ordre d'entrée →	1	2	3	4	5	6	7	8	9
	S	+	Δ	-	B	Δ	C	D	)
	5	T	A		8	A	O	P	O
		L				*	(		

Figure 75 a -
Tableau CARAC de la formule de la figure 74 a

CADRES

XMIN	XMAX	YMIN	YMAX	Xc	Yc

Figure 75 b -
Tableau CADRES de la formule

Grammaire :

La grammaire de formules qu'on propose peut être définie par les 4-uples.
G = (N, T, P, S)

où :
N = ensemble des symboles non-terminals
T = ensemble des symboles terminals : A, B, ..., Z, a, b, ..., z, + , - , * , /, (,) ; avec $T \cap N = \emptyset$; V = NUT : vocabulaire de la grammaire
P = ensemble des productions de la grammaire
S = élément de départ (S ∈ T)

Règles de la grammaire : (les chiffres entre // indiquent les numéros des règles)

EXPRESSION → TERME /1/ EXPRESSION + TERME /2/ EXPRESSION - TERME /3/
TERME → FACTEUR /4/ TERME * FACTEUR /5/
FACTEUR → - PRIMAIRE /6/ PRIMAIRE /7/
PRIMAIRE → (EXPRESSION) /8/ PR-SIMPLE /9/

PR-SIMPLE → IDENTIFICATEUR /10/ FONCTION /11/ NOMBRE /12/ EP-PARENTHESEE /13/
IDENTIFICATEUR → LETTRE /14/
FONCTION → NOM EP-PARENTHESEE/15/
NOM → SIN /16/ COS /17/ TAN /18/
NOMBRE → NOMBRE CHIFFRE /19/ CHIFFRE /20/
EP-PARENTHESEE → (EXPRESSION) /21/

Quelques exemples générés par la grammaire :

A + B * C - D

$$A + \frac{C * D}{E - C}$$

$$\frac{B - \frac{C}{SIN(A+B)} * \left(\frac{A}{B} - \frac{C}{D} \right)}{\frac{C * TAN\ (-A)}{2 + COS\ (C)}}$$

VI.3 - ANALYSEUR DE FORMULES

Nous allons donner ici un analyseur syntaxique de formules mathématiques bidimensionnelles en nous inspirant de l'algorithme général donné en (46).

Avant de passer à l'exposé de l'analyseur proprement dit, nous allons donner un aperçu sur le fonctionnement général de l'algorithme de base en nous restreignant au cas d'une chaine linéaire.

VI.3.1 - Principe de l'algorithme général

L'algorithme débute l'analyse à partir d'un terminal (ai) désigné par son importance et son emplacement dans la chaine d'entrée (a) (ai n'est pas forcément le début ou la fin de a), cherche ensuite la règle de la grammaire de description qui contient cet élément, soit : A :: = λ ai λ'
et vérifie la syntaxe des voisinages λ et λ' par analyse descendante droite-gauche pour λ et gauche-droite pour λ' .
On obtient à cette issue la ramification de la figure 75 de racine A et de feuilles λ, ai, λ' .
Si A n'est pas le but à atteindre, l'analyse est itérée sur A (on choisit B :: = μ A μ') et ainsi de suite jusqu'à description de toute la chaine et atteinte du BUT.
Des tests accompagnent généralement l'algorithme pour guider le choix des règles à chaque itération et limiter ainsi les retours arrière.

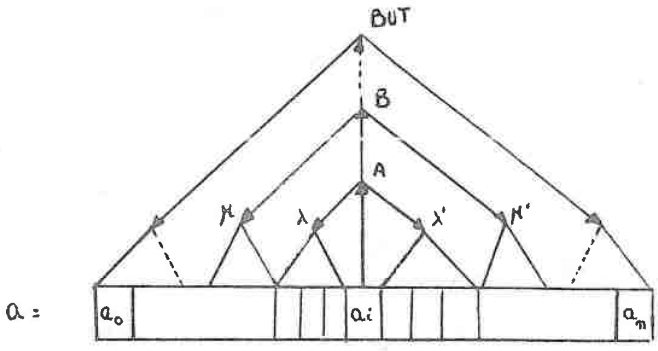


Figure 75 - Analyse ascendante-descendante de la chaine linéaire a

Cette méthode d'analyse parait assez superflue pour le traitement de chaines linéaires déterministes dont on connait parfaitement la succession des éléments et où les techniques habituelles ascendantes ou descendantes se sont montrées très performantes.

Nous la trouvons néanmoins plus adaptée à l'analyse de formes plus complexes mais très riche en contextes tels les images ou les dessins dans le plan où la notion de début et de fin est moins évidente. Elle est encore très utile à la reconnaissance de la parole continue où il peut être intéressant de commencer l'analyse en partant d'un mot clé bien prononcé.

La reconnaissance des formules mathématiques est une application où cette technique d'analyse s'adapte bien en choisissant comme point de départ des opérateurs arithmétiques.

VI.3.2 - Description de l'analyseur de formules

Cet analyseur de formules est aussi ascendant-descendant. Il démarre l'analyse à partir d'un opérateur prioritaire dans l'expression à analyser et essaie de la partager en sous-buts ou opérandes qu'il analyse par la suite de la même manière et ceci jusqu'aux identificateurs (1ère itération de la figure 76). Si la zone explorée par l'analyseur couvre la zone du but à analyser, l'analyse est considérée comme terminée, sinon elle est itérée sur le non-terminal trouvé (2ème itération du même exemple) et ainsi de suite jusqu'à atteinte du but final (EXPRESSION).

L'analyse ascendante cherche, en fonction de l'élément de départ (opérateur ou non-terminal) et de ses contextes gauche et droit la règle de la grammaire à utiliser ; cette règle donne des instructions à l'analyse descendante pour délimiter les zones des opérandes et opérateurs avoisinants.(44)

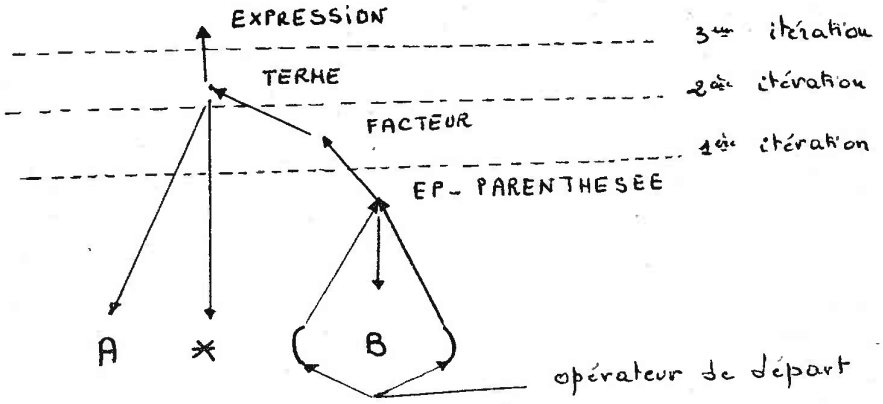


Figure 76 - Arbre syntaxique de l'expression A * (B)

Procédure d'analyse :

D = caractère de départ choisi dans l'expression à analyser.

PROCEDURE ANALYSE (ZONE , BUT , D) : résultat arbre (t)

TANT QUE D ≠ BUT FAIRE

choisir une règle $B \rightarrow C_1\ D\ C_2$
délimiter la zone Z_1 de C_1
délimiter la zone Z_2 de C_2
choisir D_1 dans Z_1 , D_2 dans Z_2
construire t_1 par ANALYSE (Z_1 , C_1 , D_1)
t_2 par ANALYSE (Z_2 , C_2 , D_2)
$t \leftarrow B\ X\ (t_1 + D + t_2)$

/* L'opération X signifie que le terme de gauche B est racine de l'arbre que forme le deuxième terme */

$D \leftarrow B$

FAIT

résultat t

FIN ANALYSE

VI.3.3 - Exemple d'analyse

Il s'agit d'analyser l'expression arithmétique A × B × C donnée par le programme de reconnaissance des caractères par le tableau suivant.

1	2	3	4	5	← ordre d'entrée
A	×	B	×	C	← chaine de caractères à analyser

↑
opérateur de départ (on le note '×' 4, 4 est son numéro dans le tableau)

Nous reviendrons plus tard sur le choix du point de départ.

Paramètres de la procédure d'analyse :

ZØNE : toute la formule : (1, 5)
1 : numéro du premier caractère dans la zone
5 : numéro du dernier caractère dans la zone
nous imposerons une écriture bien ordonnée pour que la délimitation de la zone soit facile à faire.

BUT : EXPRESSION (axiome de la grammaire à atteindre)

D : '×'4 (le choix de départ est toujours un opérateur s'il existe)

Il s'agit de construire l'arbre t par analyse de la zone (1, 5) en partant de l'opérateur × situé en 4ème position dans le tableau de la formule. Le but à atteindre étant EXPRESSION.

Soit :

t = ANALYSE ((1, 5), EXPRESSION, '×'4)

/× on choisit la règle 5 dans la grammaire qui contient l'opérateur '×'

Règle 5 : TERME → TERME × FACTEUR
(B) (C_1) (D) (C_2)

Délimiter la zone de C1 : TERME

La zone du TERME est cherchée à gauche de l'opérateur '×' dans le tableau de la formule.

Connaissant la définition du TERME d'après les règles de la grammaire, on affecte la zone (1, 3) à ce terme (Z_1 = (1, 3))

Délimiter la zone de C2 : FACTEUR

La zone du FACTEUR est cherchée à droite de l'opérateur "central" ($\times_4$) ; suivant les mêmes considérations: Z_2 = (5, 5).

Après avoir délimité la zone de chaque opérande, il s'agit de les analyser par la même procédure ANALYSE et construire leur arbre respectif t_1 pour l'opérande de gauche et t_2 pour l'opérande de droite.

Analyse de l'opérande de droite :

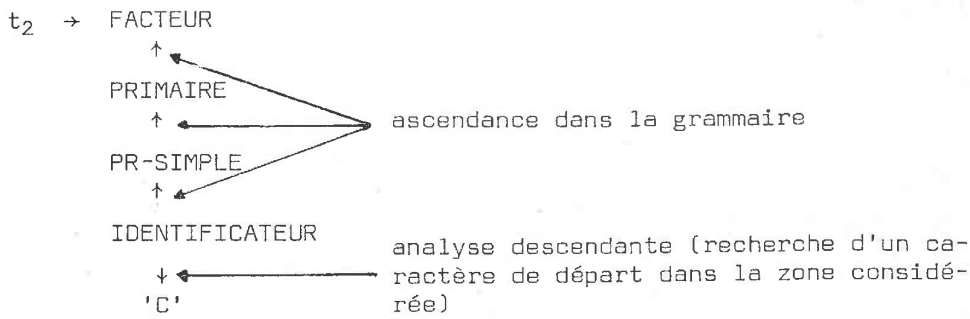
Etant réduite à un seul caractère, le caractère de départ à choisir dans cette zone sera ce caractère (identificateur).

t_2 = ANALYSE ((5, 5), FACTEUR, 'C')

on choisit la règle 14 : IDENTIFICATEUR → LETTRE

Ensuite par analyse ascendante de l'identificateur trouvé, on remonte jusqu'au FACTEUR. L'analyse de l'opérande de droite est considérée comme terminée.

L'arbre t_2 construit par la procédure ANALYSE est le suivant :



Analyse de l'opérande de gauche :

Il s'agit de construire l'arbre t_1 par :

ANALYSE ((1, 3), TERME, '×'$_2$)

'×'$_2$ est l'opérateur de départ choisi dans la zone en majuscules.

On choisit la règle 5 :

TERME → TERME × FACTEUR
(1,1) (2,2) (3,3) ← zones

t_{11} = ANALYSE ((1,1), TERME, 'A')

t_{11} →
TERME
↑
FACTEUR
↑
PRIMAIRE
↑
PR-SIMPLE
↑
IDENTIFICATEUR
↓
'A'

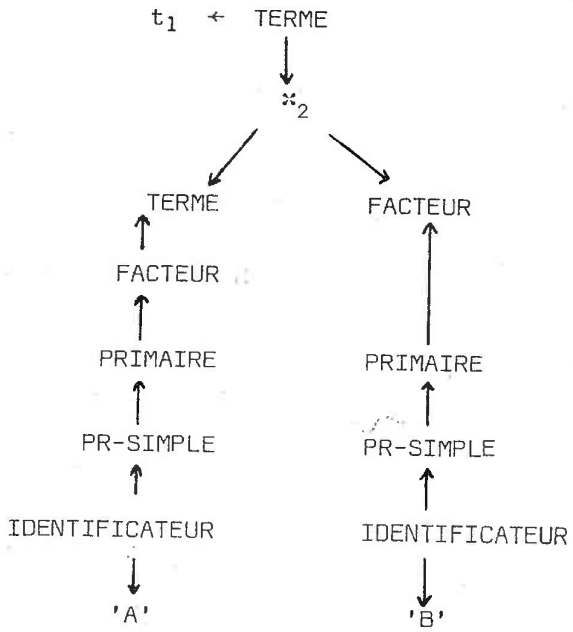
t_{12} = ANALYSE ((3,3), FACTEUR, 'B')

t_{12} →
FACTEUR
↑
PRIMAIRE
↑
PR-SIMPLE
↑
IDENTIFICATEUR
↓
'B'

L'analyse des deux opérandes TERME à gauche et FACTEUR à droite ayant réussi, nous construisons l'arbre t_1 comme suit :

t_1 ← TERME × (t_{11} + '×'$_2$ + t_{12})

ce qui donne :



$t \leftarrow \text{TERME} \times (t_1 + '\times'_4 + t_2)$

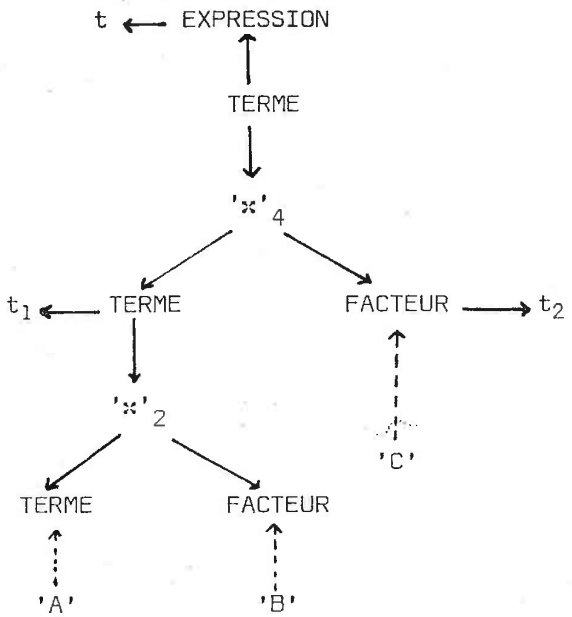
$\text{TERME} \neq \text{EXPRESSION}$

d'où : $t = \text{ANALYSE} ((1,5), \text{EXPRESSION}, \text{TERME})$

↑ analyse ascendante de TERME

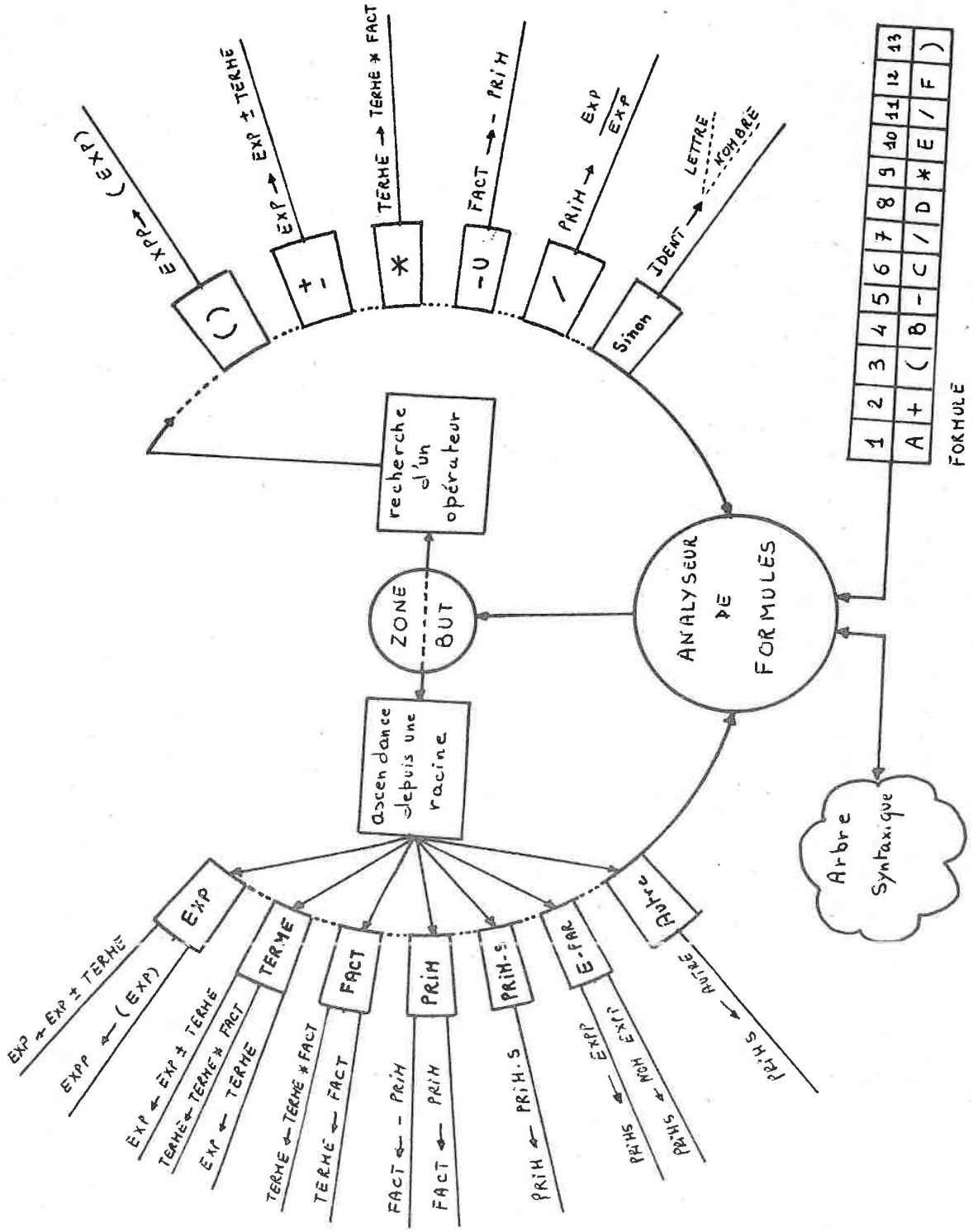
On choisit la règle 1 : EXPRESSION → TERME

L'analyse de l'expression arithmétique a donc réussi et l'arbre construit est le suivant :


t
EXPRESSION
TERME
'x' 4
t1
TERME
FACTEUR
t2
'x' 2
'C'
TERME
FACTEUR
'A'
'B'

VI.4 - FONCTIONNEMENT DE L'ANALYSEUR

Le fonctionnement général de l'analyseur peut être explicité par le schéma de la figure 77.



VI.4.1 - Choix du caractère de départ

Les caractères de départ sont choisis parmi les opérateurs arithmétiques et les signes mathématiques (+ , - , × , -Unaire , / et ()) du fait de leur importance dans la formule.

L'ordre de recherche observé est le suivant :

1. Parenthèses ()
2. + , -
3. ×
4. -Unaire
5. /

Si l'expression à analyser ne contient aucun de ces opérateurs, elle se réduit à une variable ou à une constante et son analyse est évidente.

Cet ordre de découpage nous permet de respecter la structure de l'expression conformément à la description syntaxique. De plus, il nous permet d'explorer les voisinages et localiser aisément leur emplacement dans l'espace de la formule.

Prenons l'exemple de la figure 77b , la position de l'opérateur + détermine directement le trait de fraction principal de la cascade de fractions dans la formule. En démarrant l'analyse à partir de cet opérateur, nous pourrons connaitre le centre (Xcentre, Ycentre) de la cascade de traits et déterminer par simple calcul de distance son trait de fraction principal.

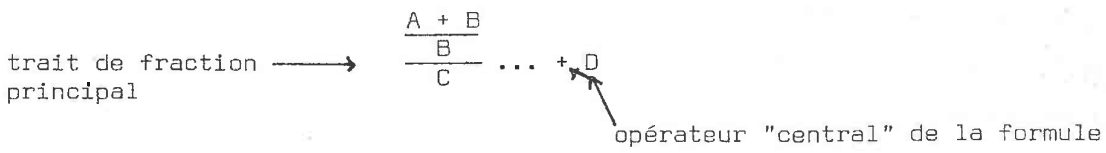


Figure 77 b - Lever d'ambiguïté sur les traits de fraction

L'ordre observé sur les quatre premiers opérateurs sera explicité naturellement dans les procédures de recherche du caractère de départ.

Nous donnons un exemple de recherche des opérateurs + dans la formule.

Recherche des opérateurs ±

La recherche des opérateurs ± se fait dans le cas où on ne trouve pas de parenthèses séparatrices dans l'expression à analyser.
Les opérateurs ± possèdent la même propriété syntaxique et ont un ordre de priorité égal.
Ils se trouvent dans les règles 2, 3 :

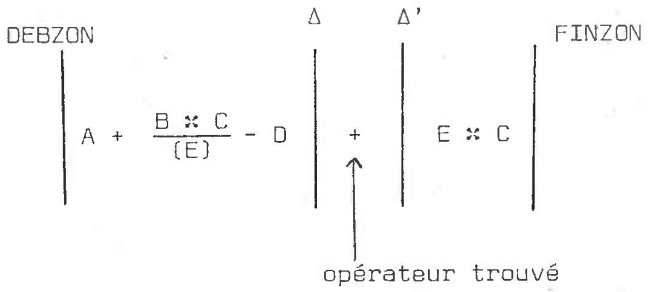
EXPRESSION → EXPRESSION ± TERME

Entrée : zone de travail de longueur ⩾ 3 ne contenant pas de parenthèses séparatrices.

Contraintes de recherches : trouver un opérateur ± séparant horizontalement une expression d'un terme (c'est-à-dire l'opérateur ± doit avoir un contexte dessus et dessous vide).

D'après la définition du terme dans la grammaire, le terme ne doit pas contenir d'opérateurs ± séparateurs. Nous prendrons dans ce cas l'opérateur ± le plus à droite dans l'expression à analyser.

Exemple :



Délimitation des zones :

zone DEBZON	→ FINZON	:	EXPRESSION (racine)
zone DEBZON	→ Δ	:	EXPRESSION (opérande gauche)
zone Δ	→ Δ'	:	(opérateur +)
zone Δ'	→ FINZON	:	TERME (opérande droit)

à vérifier

- zone des opérandes ≠ vide
- caractère final de l'opérande gauche ≠ { + , - , × }
- caractère initial de l'opérande droit ≠ {×}

Délimitation des zones :

La délimitation des zones des opérandes se fait directement dans les procédures de recherche des opérateurs de départ. L'analyseur consulte la grammaire et cherche la règle de description qui contient l'opérateur trouvé. Cette règle donne les opérandes à chercher aux alentours de l'opérateur. Ce qui facilite la délimitation de leur zone.

Recherche des identificateurs et constantes :

Si aucun des ces opérateurs n'a été trouvé, l'expression à analyser est considérée par l'algorithme comme réduite à une lettre ou à une suite de chiffres.

Nous aurons à utiliser l'une des deux règles suivantes :

IDENT → LETTRE

ou NOMBRE → NOMBRE CHIFFRE/CHIFFRE

Entrée : zone de recherche de longueur ⩾ 1

Contraintes de recherches : Si l'expression est une lettre, sa zone doit avoir une longueur = 1 et cette lettre doit faire partie du jeu de caractères imposé. Si l'expression à analyser est une constante, alors vérifier l'homogénéité de ses chiffres.

Exemples : A 123479

Figure 78 a - lettre

Figure 78 b - constante numérique entière

Dans le cas où on trouve une suite de chiffres, l'analyseur la remplace dans l'arbre syntaxique par sa valeur numérique.

VI.4.2 - Analyse ascendante d'un non-terminal

Principe : Ayant déjà construit la ramification de racine A (A → λ a λ') ; trouver en fonction du contexte la règle de description qui contient cette racine et délimiter les zones de ses voisinages pour pouvoir les analyser par la suite.
Le schéma de la figure 37 donné précédemment montre le choix à faire à partir du non-terminal trouvé.
Nous allons donner deux exemples d'ascendance.

Analyse ascendante depuis un TERME

Le terme apparait dans la grammaire dans les 4 règles suivantes :

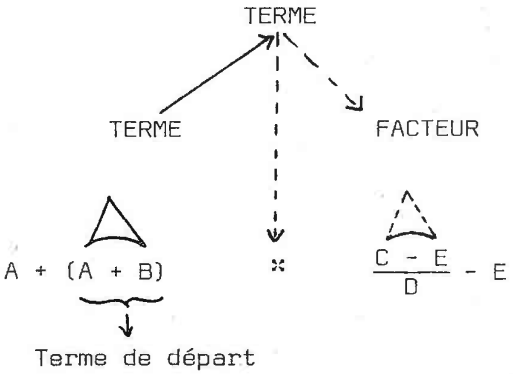
2 : EXPRESSION → EXPRESSION + TERME
3 : EXPRESSION → EXPRESSION - TERME
4 : EXPRESSION → TERME
5 : TERME → TERME * FACTEUR

Le choix d'une de ces règles sera guidé par le contexte du TERME dans l'expression à analyser.

Si voisinage droit = *
cas : TERME → TERME * FACTEUR

Le FACTEUR, par définition (description syntaxique), ne doit pas contenir d'opérateur séparateur (comme + ou - ou *). Sa zone sera donc délimitée depuis l'opérateur * trouvé jusqu'à un autre opérateur séparateur du genre indiqué, sinon jusqu'à une parenthèse fermante séparatrice (dans le cas où le TERME se trouve dans une expression parenthésée), sinon jusqu'à la fin de l'expression (la recherche se fait de la gauche vers la droite).

Exemple :



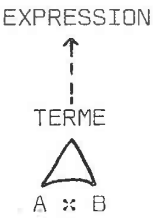
Si voisinage droit = Ø ou)

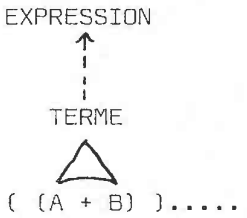
. Si voisinage gauche = Ø ou (

cas : EXPRESSION → TERME

Le terme est réduit à une expression.

Exemples :



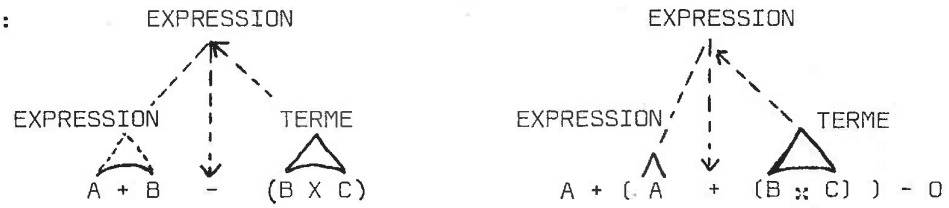


. Si voisinage gauche = + ou -

cas : EXPRESSION → EXPRESSION ± TERME

La zone de l'expression sera délimitée à gauche de l'opérateur ± trouvé depuis cet opérateur jusqu'à une parenthèse ouvrante séparatrice sinon jusqu'au début de l'expression à analyser.

Exemples :



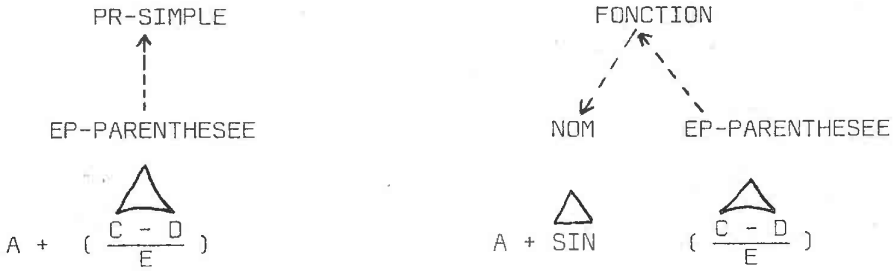
Ascendance depuis une expression parenthésée

L'expression parenthésée est donnée par les deux règles suivantes :

FONCTION → NOM EP-PARENTHESEE /15/
PR-SIMPLE → EP-PARENTHESEE /13/

Le choix de la règle peut être fait par analyse du voisinage gauche ;
en effet : si voisinage gauche = opératuer ou Ø choisir la règle n° 13
si voisinage gauche = N ou S considérés comme les finales des noms (SIN, COS, TAN), alors choisir la règle n° 15 et vérifier la syntaxe du nom.

Exemples :



Les pages suivantes illustrent un exemple d'analyse syntaxique de formules mathématiques.

Exemples d'analyse :

Analyse de la formule :

$$x - \left\langle \frac{x - y}{SIN(z)} + \frac{\frac{x}{y}}{x - y} \right\rangle$$

Chaine d'entrée : x - (- x - y SIN (z) + - x - y x - y)

DEBUT D'ANALYSE

Recherche d'un caractère de départ dans :

$$x - \left\langle \frac{x - y}{SIN(z)} + \frac{\frac{x}{y}}{x - y} \right\rangle$$

Règle 12 : EXP-PARENTHESEE ::= (EXPRESSION)

$$\left\langle \frac{x - y}{SIN(z)} + \frac{\frac{x}{y}}{x - y} \right\rangle$$

Recherche d'un caractère de départ dans :

$$\frac{x - y}{SIN(z)} + \frac{\frac{x}{y}}{x - y}$$

Règle 2 : EXPRESSION ::= EXPRESSION + TERME

$$\frac{x - y}{SIN(z)} + \frac{\frac{x}{y}}{x - y}$$

Recherche d'un caractère de départ dans :

$$\frac{\frac{x}{y}}{x - y}$$

Règle 8 : PRIMAIRE ::= $\frac{\text{EXPRESSION}}{\text{EXPRESSION}}$

$$\frac{\frac{x}{y}}{x - y}$$

Recherche d'un caractère de départ dans :

$$x - y$$

Règle 3 : EXPRESSION ::= EXPRESSION - TERME

$$x - y$$

Recherche d'un caractère de départ dans :

y

Règle 14 : IDENTIFICATEUR ::= 'y' (fin d'analyse de : TERME)

y

Recherche d'un caractère de départ dans :

x

Règle 14 : IDENTIFICATEUR ::= 'x' (fin d'analyse de : EXPRESSION)

x

... fin d'analyse de : EXPRESSION

$$x - y$$

Recherche d'un caractère de départ dans :

$$\frac{x}{y}$$

Règle 8 : PRIMAIRE ::= $\frac{\text{EXPRESSION}}{\text{EXPRESSION}}$

$$\frac{x}{y}$$

Recherche d'un caractère de départ dans :

y

Règle 14 : IDENTIFICATEUR ::= 'y' (fin d'analyse de : EXPRESSION)

y

Recherche d'un caractère de départ dans :

x

Règle 14 : IDENTIFICATEUR ::= 'x' (fin d'analyse de : EXPRESSION)

x

... fin d'analyse de : PRIMAIRE

Analyse ascendante de : PRIMAIRE

Règle 7 : FACTEUR ::= PRIMAIRE
Règle 4 : TERME ::= FACTEUR
Règle 1 : EXPRESSION ::= TERME

$\frac{x}{y}$

... fin d'analyse de : EXPRESSION

Analyse ascendante de : PRIMAIRE
Règle 7 : FACTEUR ::= PRIMAIRE
Règle 4 : TERME ::= FACTEUR

$\frac{\frac{x}{y}}{x-y}$

... fin d'analyse de : TERME

Recherche d'un caractère de départ dans :

$$\frac{x-y}{SIN(3)}$$

Règle 8 : PRIMAIRE ::= $\frac{\text{EXPRESSION}}{\text{EXPRESSION}}$

$$\frac{x-y}{SIN(3)}$$

Recherche d'un caractère de départ dans :

$$SIN(3)$$

Règle 12 : EXP-PARENTHESEE ::= (EXPRESSION)

$$(3)$$

Recherche d'un caractère de départ dans :

z

Règle 14 : IDENTIFICATEUR ::= 'z' (fin d'analyse de : EXPRESSION)

... fin d'analyse de : EXP - PARENTHESEE

Analyse ascendante de : EXP - PARENTHESEE

SIN (3)

Règle 15 : FONCTION ::= SIN. EXP - PARENTHESEE

SIN (3)

Analyse ascendante de : FONCTION

Règle 11 : PR-SIMPLE ::= FONCTION
Règle 9 : PRIMAIRE ::= PR-SIMPLE SIN (3)
Règle 7 : FACTEUR ::= PRIMAIRE
Règle 4 : TERME ::= FACTEUR
Règle 1 : EXPRESSION ::= TERME (fin d'analyse de : EXPRESSION)

Recherche d'un caractère de départ dans :

x - y

... analyse identique à celle faite précédemment sur le même exemple

Analyse ascendante de : PRIMAIRE

$$\frac{x - y}{SIN\ (3)}$$

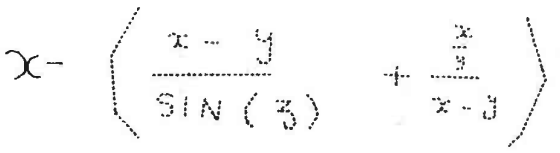
... fin d'analyse de : EXPRESSION

$$\frac{x - y}{SIN\ (3)}$$

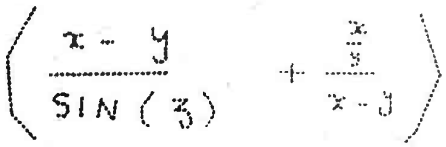
... fin d'analyse de : EXP - PARENTHESEE

$$\left\langle \frac{x - y}{SIN\ (3)} + \frac{\frac{x}{y}}{x - y} \right\rangle$$

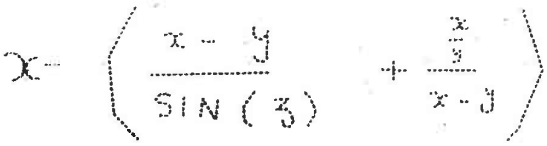
Analyse ascendante de : EXP - PARENTHESEE



Règle 13 : PR-SIMPLE ::= EXP-PARENTHESEE



Règle 3 : EXPRESSION ::= EXPRESSION - TERME



Recherche d'un caractère de départ dans :

x

Règle 14 : IDENTIFICATEUR ::= 'x' (fin d'analyse de : EXPRESSION)

... fin d'analyse de la formule

Notation postfixée de la formule :

x x y - z SIN / x y / x y - / + -

Donnez des valeurs aux variables :

?x = 34

?y = 10

?z = 1.5

Valeur = 0.93679

VI.4.3 - Gestion du retour arrière

Un retour arrière est effectué par l'analyseur, dans l'arbre syntaxique de la formule, à la suite d'une anomalie observée au cours de l'analyse.
Cette anomalie peut advenir dans les procédures de recherche ascendante ou descendante pendant les délimitations du contexte ou à l'analyse d'un identificateur ou d'une constante (mauvaise lettre ou suite de chiffres non homogène). Ceci provient généralement d'un mauvais choix d'opérateur effectué dans une étape précédente de l'analyse. L'analyseur remonte dans ce cas jusqu'au dernier choix de l'opérateur et essaie de redémarrer l'analyse avec un nouveau choix d'opérateur. Le retour arrière peut ainsi s'effectuer jusqu'au choix initial si l'analyse est mise en échec dans tous les sous-buts de l'arbre (voir fig. 79)

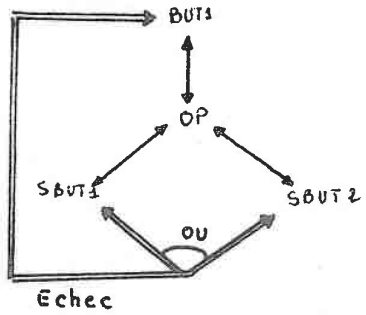


Figure 79 - Retour arrière dans l'arbre syntaxique de la formule

En cas d'échec signalé par l'analyseur dans SBUT1 ou SBUT2 dans l'exemple de la figure 79, l'analyseur remonte jusqu'au BUT1 pour le choix d'un autre opérateur.

Correction des identificateurs :

La correction des identificateurs se fait localement dans la procédure d'analyse correspondante par le choix du bon caractère (caractère convenant) dans la liste ambiguë donnée par le programme de reconnaissance.

Prenons l'exemple du caractère X de la figure 79 bis . Ce caractère est donné par le programme de reconnaissance par trois réponses différentes dans l'ordre indiqué par le schéma. L'analyseur commence l'analyse avec la première réponse de la liste, soit Y. En cas d'échec, l'analyseur descend dans cette liste jusqu'à la rencontre du caractère qui convient, soit ici X.

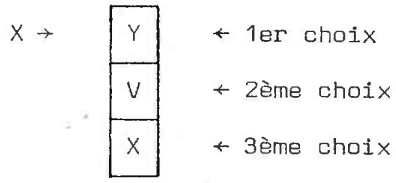


Figure 79 bis - Choix d'une variable

La recherche est identique pour les chiffres.

Corrections des opérateurs :

Les opérateurs susceptibles d'être corrigés sont les parenthèses et les opérateurs + et ×

Ils peuvent être donnés pour une suite de caractères ambigus

formule entrée (C + T × (A + O) + T)

caractères reconnus C (T L Δ C × T (O + T)
O O + + (A L O) L
(C L T A + C

Le rôle de l'analyseur consiste à trouver dans chacune de ces listes le caractère qui convient pour la reconstruction d'une formule syntaxiquement exacte ; au cas idéal, retrouver la formule de départ.

Correction des parenthèses :

(nombre de parenthèses ouvrantes égal à celui des fermantes avec alignement correct)
L'analyseur cherche, au départ, un parenthésage correct qui peut servir d'appui pour la recherche des opérateurs et la délimitation du contexte. Cette correction se fait dans un module à part avant les procédures d'analyse ascendante ou descendante. En cas d'échec sur ce parenthésage, l'analyseur remonte jusqu'à ce module et cherche un autre parenthésage possible.

Pour chaque parenthésage détecté, le module principal vérifie :

- Pour l'ensemble des parenthèses :
  - . nombre de parenthèses ouvrantes = nombre de parenthèses fermantes
  - . il n'y a pas de chevauchement entre ces parenthèses

Ensuite pour chaque couple de parenthèses (ouvrante , fermante), il vérifie :

- l'alignement ($\hat{m}$ abscisse)
- le contenu
  - . $\neq \emptyset$

  Si contenu = un caractère
  vérifier que le caractère n'est pas un opérateur
  Si contenu = deux caractères
  vérifier que le premier caractère est, soit :
  - . un chiffre
  - . un opérateur (-)

- le voisinage immédiat

  voisinage droit : doit être Ø ou un opérateur (+ (- (×
  ou une parenthèse fermante

  voisinage gauche : il peut être soit : Ø
  soit : opérateur
  : une parenthèse ouvrante
  sinon un caractère S ou N final du nom d'une fonction trigonométrique.

Retour sur l'opérateur +

Le retour arrière sur l'opérateur + se fait quand l'analyse d'un de ses voisinages EXPRESSION à gauche ou TERME à droite (délimité par analyse descendante) est mise en échec. L'analyseur essaie dans ce cas, soit de redémarrer l'analyse avec un autre opérateur + situé à sa droite et donné par la reconnaissance en deuxième, troisième ... position dans la matrice de réponses si opérateur existe, sinon de le remplacer par un autre caractère de sa liste et chercher un autre opérateur + à sa gauche.

Ceci revient finalement, chaque fois qu'on trouve un opérateur + dans l'expression à analyser, à retenir tous les autres choix possibles dans une pile avec les informations suivantes :

- nom
- position dans la formule
  - . ordre d'entrée
  - . position dans la liste
- état de l'arbre syntaxique
  - . nom du sous-but à analyser
  - .. position dans l'arbre

En cas d'échec, l'analyseur remonte à l'état signalé par le sommet de pile et redémarre l'analyse avec un nouveau choix d'opérateur +

Exemple :

Soit à analyser l'expression donnée par la matrice de réponses suivantes :

1	2	3	4	5	6	7
A	+	T	T	T	T	T
			+		+	

PILOP

1	+	6,2
1	+	4,2
Etat	Op.	Position

L'analyse démarre avec la première ligne de la matrice. On choisit l'opérateur + n° 2 et on retient dans PILOP le 4ème et le 6ème

1ère phase : EXP + TERME
(A) (TTTTT) ← échec sur le terme

2ème phase : EXP + TERME
(A+TTT) T

3ème phase : EXP + TERME
(A) (TTT) ← échec

4ème phase : EXP + TERME
(A+T) (T) ← succès

VI.5 - CONCLUSION

Nous avons montré dans ce chapitre l'utilité d'une méthode contextuelle pour l'analyse et la compréhension des formules mathématiques bidimensionnelles. La méthode qu'on a adoptée ici est une méthode syntaxique ascendante-descendante pilotée par une grammaire de coordonnées où chaque règle de description précise à l'analyseur le contexte à chercher en fonction de l'opérateur (analyse descendante) ou le non-terminal (analyse ascendante) de départ.

Le résultat obtenu à la fin de cette analyse est un arbre syntaxique qui redécrit de façon cohérente ces liens topographiques entre opérateurs et opérandes.

L'approche réalisée ici nous assure trois avantages :

- résolution de l'ambiguité par utilisation du contexte
- analyse quasi déterministe grâce au bon choix des points de départ
- stratégie indépendante de l'algorithme de reconnaissance

Cette application, outre les avantages qu'elle nous offre, peut être considérée dans notre cas comme un cadre de choix pour tester nos travaux antérieurs sur la reconnaissance structurelle des caractères manuscrits et fixer nos idées sur l'importance du contexte pour l'analyse et l'interprétation de formes complexes. C'est pourquoi nous pensons l'orienter, dans l'avenir, vers la réalisation d'un système intéractif qui piloterait la segmentation des caractères au niveau inférieur en cas d'ambiguité ou de mauvaise reconnaissance.

Cette intervention se fera dans le système de façon ascendante-descendante entre le niveau inférieur de segmentation et de reconnaissance et le niveau supérieur d'analyse syntaxique comme le montre la figure 80.

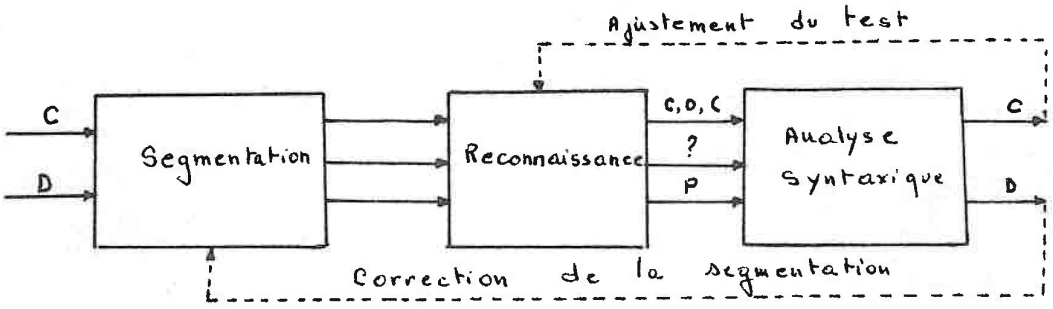


Figure 80 - Interactions envisagées entre les niveaux du système

Le programme de reconnaissance peut donner trois formes d'erreurs :

- reconnaissance ambiguë

Un même caractère peut être donné par plusieurs réponses possibles. Le caractère d'origine en fait partie.

$$C \rightarrow \begin{matrix} C \\ (\\ O \end{matrix}$$

au cas où l'analyseur retient la bonne réponse (c'est-à-dire le C dans l'exemple), il peut ordonner au MAITRE d'ajuster ou de créer le test de lever d'ambiguïté entre ce caractère et les autres réponses dans l'arbre de décision. Ceci pourra se faire de façon automatique sans l'intervention de l'écrivain.

- reconnaissance erronée

Le caractère reconnu ne correspond pas au caractère entré :

$$S \rightarrow 5$$

Si l'analyseur retrouve le caractère d'origine (S) pendant l'analyse, il ordonne à l'automate de segmentation de corriger la segmentation du S . Ce dernier, après consultation de l'arbre de décision et examen de toutes les listes de primitives du S, il peut ajuster sa segmentation (ceci pourra nous amener jusqu'à rectification automatique des paramètres de segmentation).

- mauvaise reconnaissance

Il n'y a pas de reconnaissance possible pour le caractère entré.

 → ? bavure

 → ? écriture penchée

 → ? écriture fantaisiste

La mauvaise reconnaissance peut être due à des mauvaises écritures comme le montre la figure ci-dessus :

Deux cas sont possibles :

- L'analyseur donne une ou plusieurs réponses possibles.
  Dans ce cas, l'automate de segmentation essaie de corriger le caractère en supprimant certaines primitives (bavures dans le premier cas de figure) et essaie de trouver une segmentation qui se rapproche le plus de celle des modèles rangés dans l'arbre de décision.

- L'analyseur ne donne pas de réponse.
  Dans ce cas, on peut rejeter le caractère en demandant à l'écrivain d'écrire un autre

L'analyse syntaxique de formules mathématiques a été déjà abordée par ANDERSON. Seulement, l'analyse adoptée est une analyse syntaxique classique gauche → droite renforcée par quelques heuristiques qui est,aux dires d'ANDERSON lui-même, très lente ; alors que la méthode que nous avons choisie, même si elle est appliquée sur une structure de formules plus petites que celles choisies par lui, est bien mieux adaptée et permet par des tests bidimensionnels de limiter les choix, voire d'analyser de façon déterministe si l'écriture de la formule est correcte.

CONCLUSION

La reconnaissance de l'écriture est un domaine qui relève de l'intelligence artificielle et qui réclame un large éventail de techniques de description robustes pour pouvoir résister aux variations et distorsions incessantes de ces formes et rigoureuses pour conduire à la meilleure solution.

Bien que ces deux conditions ne soient que rarement réunies, tout au moins pour l'écriture manuscrite, nous avons néanmoins tenté par les deux volets de cette étude - reconnaissance de caractères et analyse de formules mathématiques - de regrouper ces deux conditions dans notre système en faisant appel à des techniques purement structurelles.

Ces deux impératifs nous ont conduit à définir :

Au niveau du tracé manuscrit :

Un automate d'extraction de primitives robustes de types : segments de droite de différentes pentes et longueurs et des arcs de cercle de différentes directions et courbures. Cet automate présente un aspect original de segmentation : détection de primitives locales et globales facilement reconnaissables à la simple vue du tracé, et ceci à l'aide de tests simples introduits par le concepteur et pouvant être ajustés à n'importe quelle figure. Les réponses de l'automate sont multiples et valuées, ce qui permet à la procédure de décision de faire son choix sur la meilleure occurence.
Ce même automate a été utilisé pour la segmentation de différentes classes de dessins graphiques. Il présente une grande tolérance aux variations signalées ci-dessus.

Au niveau du caractère :

Un procédé d'apprentissage cohérent et efficace permet à la procédure de décision une recherche rapide et aisée. Un arbre de décision est rempli automatiquement et renforcé par des tests de séparation en cas d'ambiguïté. Ces tests portent sur des mesures globales faites sur les caractères telles les positions relatives des points d'intersection des divers tracés par rapport à leurs extrémités. Des réponses multiples et valuées sont là-aussi fournies. Elles permettent à l'analyseur de formules de faire son choix sur le caractère le mieux reconnu (donné par le score le plus élevé).

Nous pensons avoir ainsi contribué à la réalisation d'un outil général de segmentation en primitives qui tient au mieux compte des variations et distorsions des formes graphiques tracées à la main. L'avantage d'un tel outil par rapport à ceux qui existent, réside dans sa souplesse et la simplicité d'adaptation à n'importe quelle forme. Le type de primitives, leur nombre ainsi que leurs classes topographiques sont des paramètres du système. Chaque concepteur peut les ajuster sans difficulté en fonction de la forme à traiter et du résultat à obtenir.

Il est tout à fait possible d'étendre la segmentation faite sur les caractères aux dessins graphiques ou encore à des formes plus complexes telles les images (nécessitant à priori une squelettisation au préalable). Nous pensons l'utiliser dans l'avenir pour l'identification de pièces de fonderie.

Pour les caractères, l'approche structurelle semble néanmoins trop générale pour résoudre toute la complexité observée sur ces formes. Deux techniques d'amélioration ont été réellement envisagées pour accroitre les performances du système de reconnaissance. La première visait à assouplir l'élaboration fine faite sur les primitives (trop sensibles), ce qui nous aurait conduits aux systèmes existants qui se heurtent eux aussi aux mêmes difficultés (variations et distorsions). La deuxième visait à introduire la notion de contexte comme nous l'avons fait dans ce système. Le système est capable de réduire tout seul l'ambiguité (1,1 % dans notre cas) et de décider lui-même de la reconnaissance du caractère.

L'interaction prévue entre les deux niveaux du système permet d'obtenir un score de reconnaissance plus élevé.

En conclusion, nous pouvons nous poser une dernière question :
Existe-t-il vraiment une méthode de description rigoureuse dont l'application au caractère manuscrit conduirait malgré toutes les variations de l'écriture à une reconnaissance parfaite (100 %) ?
Nous pensons, quant aux méthodes structurelles, qu'elles sont largement insuffisantes à moins d'introduire le contexte : TOUSSAINT (49). C'est ce que nous avons essayé de montrer par le dernier volet de cette étude.

BIBLIOGRAPHIE

/1/ HUSSAIN A.B.S. , DONALDSON R.W. ,

Suboptimal sequential decision schemes with on-line feature ordering .
IEEE trans on computers - July 1972

/2/ PATTERSON J.D.

The linear mean square estimation technique of recognition .
IJCPR, 1 nov. 1973, Washington - pp. 67-76

/3/ CASEY R.C. , NAGY G.

An autonomous reading machine .
IEEE Trans. on computers. Vol. C17 , n° 5, pp. 492-503

/4/ CARCENAC DE TORNE B.

Identification des adresses postales.
Congrès AFCET - fév. 78 - pp. 601-611 tome 2

/5/ SPANJENSBERG A.A.

Combination of differents systems for the recognition of handwritten digits.
IJCPR 11 August 74, pp. 208-209

/6/ DEMARTINI J. , ROMERO C. (Université de Nice),

Reconnaissance des lettres manuscrites par modélisation et corrélation spaciale.
Congrès AFCET, Fév. 78, pp. 553-559, tome II

/7/ TEITELMAN W.

Real time recognition of hand drawn characters .
FJCC (1964) pp. 559-575

/8/ MUNSON J.M.

Experiments in the recognition of hand printed text.
Part. I : character recognition.
FJCC (1963) p. 1125

/9/ TOUSSAINT G.T. , DONALDSON R.W.

Algorithms for recognizing contour-traced handprinted characters.
IEEE Trans on Computer - Juin 70

/10/ OTT R.

On feature selection by means of principle axis transform and non linear classification.
IJCPR 11 August 74, pp. 220-222

/11/ CANDREW K. , GHAHRAMAN D.

A statiscal analysis of Interdepence in character sequence
Information sciences, 8, 173-188 (1975)

/12/ SIMON J.C. , CHECROUN A. , ROCHE C.

A methode of comparing two patterns independant of possible transformations and small distorsions.
Pattern recognition pp. 73-81 (janv. 1972)

/13/ MILLER G.M.

On line recognition of hand generated symbols.
FFJC (1969) pp. 399-412

/14/ GAILLAT G.

Une procédure statistique de décision avec apprentissage et son application à la reconnaissance des caractères manuscrits.
Thèse de 3ème cycle PARIS VI, Mai 1975

/15/ GAILLAT G.

A simple learning decision algorithm for character recognition and pattern classification.
Pattern Recognition vol. 10 pp. 99-104 - Pergamon Press 1978

/16/ VO VAN T.

Contribution à la conception de systèmes de reconnaissance de caractères non stylisés.
Thèse de 3ème cycle - Besançon, décembre 1977

/17/ POWERS V.M.

Pen direction Sequence in character Recognition.
Pattern Recognition. Pergamon Press 1973
Vol. 5 pp 291-302

/18/ LEGER E.

Méthode de segmentation de formes graphiques et son application à leur identification automatique.
Congrès AFCET-IRIA, fév. 78

/19/ HOSKING K.H.

. A contours method for the recognition of handprinted character. Machine perception of patterns and pictures WHITERFRIARS (1972)

/20/ CHAPELLE A. , SERRA J.

LORETTE G. , REQUIER J.P.

Reconnaissance automatique de caractères manuscrits bâton
Congrès AFCET, fév. 78, pp. 560-567

/21/ NARASHIMAN R. , REDDY V.S.N.

A syntax-aided recognition scheme for handprinted english letters.
Pattern recognition Pergamon Press 1971, vol. 3, pp. 345-361

/22/ CASKEY D.L , COATES C.L.

Machine recognition of handprinted characters.
IJCPR 1, nov. 1973, pp. 41-49

/23/ BALM G.J.

An introduction to optical character reader considerations.
Pattern recognition Pergamon Press 1970 vol. 2, pp. 151-166

/24/ BROWN R.H.

On line computer recognition of handprinted characters.
IEEE TRANS. on computer. Déc. 64, pp. 750-752

/25/ AMIN A.

Reconnaissance des caractères arabes
NANCY I (thèse à paraître)

/26/ BERTHOD M.

Une méthode syntaxique de reconnaissance de caractères manuscrits en temps réel avec apprentissage continu.
Thèse de 3ème cycle PARIS VI, 1975

/27/ BERTHOD M. et JANCENNE P.

Le prétraitement des tracés manuscrits sur une tablette graphique.
Congrès AFCET, Toulouse, Sept. 79

/28/ SHAPIRO LINDA G.

Inexact pattern matching in ESP^3.
Departement of computer science Kansas state University Manhattan, Kansas 66506

/29/ YOSHIDA M. , EDEN M.

Handwritten Chinese character recognition by an analysis-by-synthesis method.
Proc 1st int. conference in Pattern Recognition, pp. 197-204 (1973)

/30/ RAJASEKARAN S.N.S. , DEEKSHATULU B.L.

Recognition of printed Telugu characters.
Computer graphics and image processing 6, 335-360 (1977)

/31/ ITO M.R. , CHUI T.L.

Real time recognition of handprinted characters Using syntactic stroke identification.
Département of electrical enginering
Université of British Columbia Vancouver, B.C. CANADA

/32/ CHEHIKIAN A.

Conception et réalisation d'un automate de reconnaissance de caractères alphanumériques.
Thèse d'état. Grenoble, juin 1977

/33/ BOUVIER G.

Système d'acquisition et de reconnaissance de caractères alphanumériques.
Thèse de 3ème cycle, Grenoble, Mai 1977

/34/ LEDLEY R.S.

High-speed automatic analysis of biomedical pictures.
Sciences 146, pp. 216-229 (1964)

/35/ LEDLEY R.S. , ROTOCO L.S. , GOLAB T.Y. , JACOBSEN J.D. , GINSBERG M.D. , WILSON J.B.

FIDAC : Film Imput to Digital Automatic Computers and Associated syntax directed pattern recognition programming system.
Optical and Electro-optimal information processing.
Cambridge, M.A., MIT Press, pp. 591-613 (1965)

/36/ HOROWITZ S.L.

A syntactic algorithm for peak detection in waveforms with application to cardiography.
Communication ACM, n° 5, vol. 18 (1975)

/37/ TOHME S.

Prétraitement du chiffre manuscrit.
Congrès AFCET - Fév. 1978

/38/ STALLINGS W.

Approchesto chinese character recognition
Pattern recognition Pergamon Press 1976, vol. 8 pp. 87-98

/39/ CASTAN S., PERENOU G.

Reconnaissance structurelle de graphismes
Congrès AFCET, fév. 78

/40/ IKEDA K., YAMAMURA T., MITAMORA Y., FUJIWARA S., KIYONO T.,

On line recognition of hand-written characters utilizing positional and stroke vector sequence.

/41/ PAVLIDIS T.

"Structure pattern recongition : Primitive and juxtaposition relations" in frontier of pattern recognition,
S. WATANBLE ED. New-York : Academic Press 1972, pp . 421-451

/42/ MASINI G.

Réalisation d'un système de reconnaissance structurelle et d'interprétation de dessins.
Thèse de 3ème cycle, NANCY I (1978)

/43/ MOHR R.

Descriptions structurées et analyses de formes complexes.
Application à la reconnaissance de dessins.
Thèse d'état. NANCY I (1979)

/44/ BELAID A.

Reconnaissance en ligne de formules mathématiques manuscrites.
2ème congrès AFCET-IRIA, sept. 79

/45/ BELAID A.

Segmentation de tracés en vue de leur analyse. Application à la reconnaissance des caractères manuscrits.
Congrès AFCET-IRIA, nov. 78

/46/ MOHR R. , HATON J.P.

A parsing algorithm for imperfect patterns and its application to speech and image recognition.
in 3d IJCPR, San Diego, USA Nov. 76

/47/ MARI J.F.

Contribution à l'analyse syntaxique et à la recherche lexicale en reconnaissance du discours continu.
Thèse de 3ème cycle NANCY I, juin 1979

/48/ ANDERSON R.H.

Two-dimensional mathematical notation.
In syntactic Pattern Recognition (Fu Ed.) Springer-Verlag (1977)

/49/ TOUSSAINT G.T.

The use of contex in pattern recognition.
Pattern recognition vol. 10, pp. 189-204
In Pergamon Press 1978

E R R A T A

page 10 — 8ème ligne : assez prometteuse...

page 12 — 27ème ligne : telles le trait

page 25 — dernier paragraphe, 1ère ligne:nous montrerons ici comment on l'a..

page 33 — dernière ligne : SBAV à la place de PBAV

page 34 — 12ème ligne : SBAV = $\frac{1}{5}$ de la...

page 74 — avant dernière ligne : $S_c(C_{+alf}) = \frac{S_c(a)+S_c(1)+S_c(f)}{3}$

page 72 — dernière ligne : deux réponses sont là aussi...

page 91 — 19ème ligne:du pointeur dans TCAR

page 99 — dernière ligne : quelle que soit la...

page 101 — 19ème ligne : (seuil minimum...)

page 106 — ligne 6 : racine de l'arbre,l'acheminement...

page 112 — ligne 19 : l'écrivain prend en compte les...

page 125 — 4ème ligne : jusqu'à atteindre le but...

page 131 — 7ème ligne : N=ensemble des symboles non-terminaux

8ème ligne : T=ensemble des symboles terminaux

page 134 — 6ème ligne : ...telles les images

page 138 — 4ème ligne : ...choisi dans la zone du TERME.

page 145 — 7ème ligne : ...de la figure 78...

page 147 — 6ème ligne : ...opérateur ou...

page 158 — avant dernière ligne : du S, peut ajuster...

page 161 — 13ème ligne : ...et longueur, et arcs de cercle...

NOM DE L'ETUDIANT : Monsieur BELAID Abdelwaheb

NATURE DE LA THESE : DOCTORAT de TROISIEME CYCLE d'INFORMATIQUE

VU, APPROUVE

et PERMIS D'IMPRIMER

NANCY LE 9 octobre 1979

POUR LE PRESIDENT DE L'UNIVERSITE DE NANCY I
Le vice-président,

M. BOULANGE