IDIAP

Rapport technique

Apprentissage de prototypes de caractères à partir de l'image d'un texte manuscrit et avec l'aide d'un opérateur

Stéphane Brunet

DESS ISA - IFSIC - Université de Rennes I

mars - juin 95

INSTITUT DALLE MOLLE D'INTELLIGENCE ARTIFICIELLE PERCEPTIVE
CASE POSTALE 592 - 1920 MARTIGNY - VALAIS - SUISSE
TELEPHONE : ++41 26 22.76.64 - FAX : ++41 26 22.78.18
E-MAIL : IDIAP@IDIAP.CH

Numéro : 95.01

Sommaire

Liste des figures

Abstract

Ce rapport décrit la réalisation d'une interface permettant à un opérateur d'extraire de l'image d'un texte manuscrit des prototypes de caractères et de les accumuler dans une base de données devant servir ultérieurement à la reconnaissance de l'écrit. L'utilité d'une telle interface intervient dans le cas où la reconnaissance de l'écrit s'applique à de longs documents ayant un style particulier(par ex. documents anciens).
La méthode utilisée part d'une représentation des caractères par le contour et résout la difficulté que constitue l'isolement d'un caractère en sur-segmentant légèrement l'‘image des contours par rapport à une segmentation en caractères. L'opérateur rassemble ensuite les segments pour former les prototypes de caractère.

This report describes the realisation of an interface by which an operator can extract character prototypes from the image of a handwritten text and accumulate them into a database, which will be used afterwards for text recognition. Such an interface is usefull in the case where text recognition is required for long documents with a particular writing style (e.g. ancient documents).
The method applied in this system, based on a boundary representation of characters, solves in the following manner the difficulty of isolating a character. The boundary image is slightly over-segmented with respect to a segmentation into characters. The operator gathers then the segments composing a single character.

Chapitre 1

Introduction

1.1 La Vision artificielle

La vision artificielle constitue un défit pour la science et la technique de la fin du 20ième siècle et mobilise de nombreux chercheurs.
Elle met en coopération des domaines de connaissances aussi variés que les mathématiques (Topologie, Géométrie, Statistiques, Morphologie), le traitement du signal, l'informatique(logiciel, matériel), les techniques de reconnaissance des formes et d'intelligence artificielle, la linguistique ou bien encore la psycho-physiologie.
Le traitement automatique de l'écrit partage avec la vision artificielle un certain nombre de thèmes comme la reconnaissance de l'écrit et du scripteur et en possède d'autres plus spécifiques comme l'acquisition, la compression et la sauvegarde de l'écrit. C'est un axe de la recherche actuelle, notamment à L'IDIAP[0], qui fournit le contexte scientifique du travail que nous menons durant un stage d'une durée de quatre mois.

1.2 Le traitement électronique de l'écrit

1.2.1 Catégorisation des problèmes

- *reconnaissance de l'écrit - Identification du scripteur*

- *Ecriture on-line/off-line*
  L'écriture "on-line" est représentée par un vecteur fonction du temps dont la saisie a lieu à l'instant même où le scripteur la formule. En revanche, l'écriture "off-line" est représentée par les coordonnées spaciales de l'ensemble des points représentatifs de l'image la supportant; dans ce cas, l'écrit est figé, l'action est terminée. Naturellement, en ce qui concerne l'exploitation pour la reconnaissance de ces représentations, cela relève de problématiques différentes.

- *écriture imprimée/manuscrite*
  Cette distinction n'a de sens que pour la reconnaissance de l'écrit : implicitement, la reconnaissance du scripteur ne peut être faite que pour des documents manuscrits.
  Le tableau ci-dessous dresse un comparatif et montre que l'imprimé a des caractéristiques

[0]Intistut Dalle Molle d'Intelligence Artificielle perceptive - Martigny - Valais - Suisse.
Contact e-mail : gilbert.maitre@idiap.ch

que le manuscrit ne possède pas, ce qui explique la difficulté accrue de la reconnaissance de l'écriture manucrite.

caractéristiques	imprimé	manuscrit
existence modèles de caractères (fontes)	oui	non
régularité	+	-
variation des styles d'écriture	-	+
liaisons entre caractères	non	oui

– *écriture occidentale - autres écritures*
Selon les populations, les écritures utilisées ont des structures différentes : par exemple, l'écriture chinoise est constituée de symboles(idéogrammes) qui peuvent s'emboîter les uns dans les autres de façon hiérarchique. Cela la distingue fondamentalement de l'écriture occidentale qui elle se décompose en mots séparés par des espaces ou par la ponctuation, les mots étant formés par une suite de caractères pris dans un alphabet.

1.2.2 Quelques applications

En ce qui concerne la reconnaissance de l'écrit, le niveau atteint par la technologie a permis d'ores et déjà l'automatisation de certaines tâches administratives pénibles pour lesquelles le vocabulaire reconnu n'est pas flexible :

– tri automatique du courrier (reconnaissance de l'adresse et du code postal)

– lecture automatique de chèques (bancaires ou postaux)

– lecture automatiques de formulaires (par ex. recensement, feuilles d'impôt, etc..)

Concernant la reconnaissance du scripteur, l'application principale en est la sécurité au niveau de l'accès à des systèmes automatisés. Cependant, ces systèmes doivent encore être améliorés.

1.2.3 Reconnaissance de l'écrit manuscrit occidental

1.2.3.1 Entités à reconnaître

– Mots entiers : on utilise un lexique enrichi lors d'un apprentissage et qui ne peut contenir qu'un nombre limité de mots car sinon, les performances s'effondrent. La méthode peut être implantée avec de bons résultats dans des applications utilisant un vocabulaire restreint (non flexible).

– Caractère par caractère : un texte est composé d'une suite de caractères. On génère simplement une réponse ASCII de l'image du texte. La méthode n'utilise pas pour sa mise en oeuvre de modèle linguistique (on n'accède pas ainsi à l'interprétation du texte ASCII ni donc à la suppression des fautes d'orthographes).

1.2.3.2 Caractéristiques graphiques pour la reconnaissance des entités

– Caractéristiques globales : pour chaque entité, tous les points sont capturés dans une imagette extraite de l'image support. Cela implique un coût élevé pour le stockage des données. Cette méthode de représentation permet l'implantation rapide de la technique de calcul de corrélation de masques pour la reconnaissance, mais dont les performances sont malheureusement très sensible aux modifications du signal (accumulation de bruit, variations de taille, variations de forme, rotations du modèle considéré).

- Distribution statistique de points : le regroupement de points s'effectue en fonction de propriétés statistiques qu'ils réunissent (ex représesntation par zones de densité, caractéristiques "loci", méthodes des distances et des croisement, n-tuples [8]).

- Caractéristiques topologiques et géométriques : il s'agit de rassembler un ensemble de points vérifiant certaines propriétés géométriques ou topologiques. Cette technique est la plus communément utilisée car c'est celle qui offre pour la reconnaissance la meilleure tolérance aux distortions, variations de style, translations et autres rotations.

1.2.3.3 Exploitation des supra-caractéristiques de l'écrit

Elle se greffe sur un système de reconnaissance de l'écrit une fois faite la reconnaissance des entités de base, et peut en améliorer les performances de manière significative. En effet, il apparaît dans la littérature que pour la vision humaine, le contexte physique de l'objet reconnu quel qu'il soit est un critère psycho-visuel très important; d'où l'intérêt d'essayer de le formaliser d'une manière ou d'une autre.
On peut pour cela tenir compte des caractéristisques liguistiques de l'écrit : il s'agit de caractéristiques de haut niveau que l'on formalise en tirant parti de notre connaissance de la langue étudiée, à l'aide de règles syntaxiques avec lesquelles on peut décomposer l'organisation de caractéristiques graphiques élémentaires ou d'entités telles qu'elles sont définies plus haut.
Par ce type de démarche, on peut accéder par exemple et comme nous le souhaitons, à la reconnaissance de mots par la reconnaissance de caractères : après la reconnaisance brute des caractères d'un document, on utilise des modèles de mots répertoriés dans un dictionnaire et qui permettent de corriger les fautes d'orthographes faites par le scripteur.

1.3 Définition du travail à réaliser

1.3.1 Contexte général

A l'IDIAP, où les études menées se concentrent sur la reconnaissance optique de caractères, nous développons actuellement un système de reconnaissance des caractères qui se veut robuste par rapport à la détérioration de l'empreinte graphique du caractère. Cette propriété de robustesse est non seulement requise pour le traitement de certains documents(par ex. documents anciens, maculé, ...) mais aussi du fait que les méthodes de segmentation du texte en caractères ont des performances limitées. Un système a été testé sur des bases de données standards (NIST, CEDAR) pour lesquelles les échantillons de caractères sont accessibles dans des images individuelles et il existe pour chaque image l'indication du caractère représenté.
Actuellement, nous voulons compléter le système de sorte à pouvoir l'utiliser dans des situations réelles où les caractères ne sont pas isolés. Le travail proposé va dans cette direction.

1.3.2 Choix antérieurs

1.3.2.1 Représentation d'un caractère

Un caractère est représenté par un ensemble de prototypes.
définition:
un prototype est une instance de caractère sélectionnée lors d'un apprentissage.

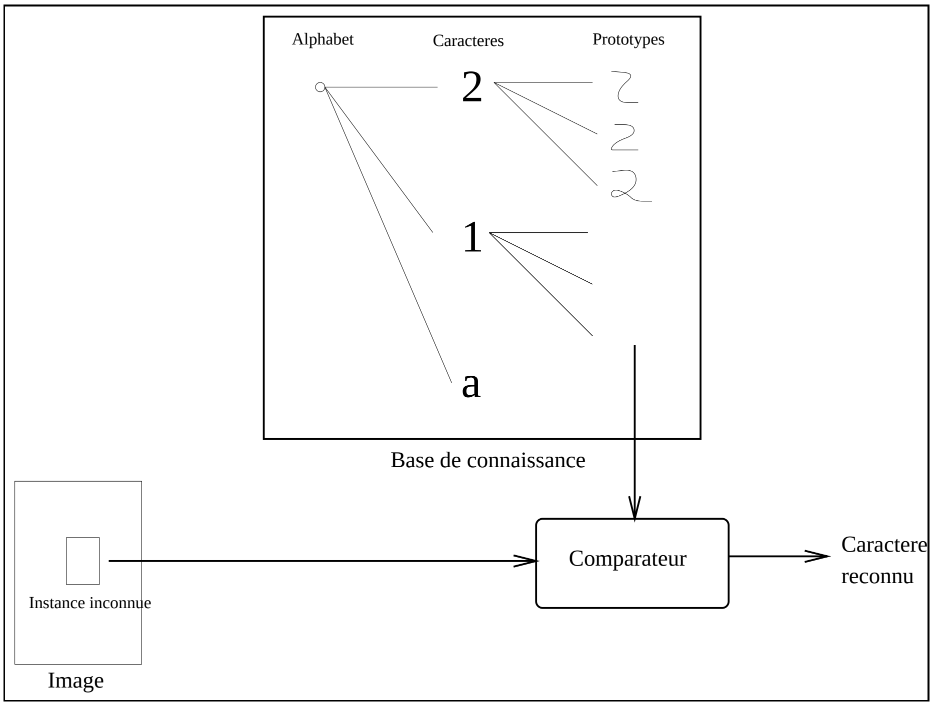

FIG. 1.1 - Reconnaissance d'une instance inconnue avec un comparateur

1.3.2.2 Représentation d'une instance de caractère

On a choisi une approche "*contour*" dont les performances se sont révélées meilleures par rapport à l'approche "squelette" [5]. Une instance est représentée par l'ensemble des points de son contour extraits de l'image du document manuscrit.

1.3.2.3 Reconnaissance d'une instance inconnue

Elle est réalisée en comparant l'instance inconnue avec les prototypes(cf figure 1.1). Plusieurs approches ont été proposés à l'IDIAP et sont décrites dans [4] et [5].

1.3.2.4 Apprentissage supervisé

Lors de ce traitement, un opérateur associe une instance inconnue à un symbole et ce couple est alors répertorié dans la base de connaissance.
Remarque :
Un système d'apprentissage supervisé peut-être étendu à un système d'apprentissage semi-automatique(dont une utilisation pratique peut-être faite pour la transcription d'archives) [0] . Dans un tel système, le superviseur humain est relayé par un algorithme de reconnaissance; la base de donnée est incrémentée lorsque l'algorithme de reconnaissance exécuté par le calculateur n'est pas parvenu à prendre une décision fiable et que le superviseur humain a levé l'ambiguité(cf figure 1.2). Une amélioration possible d'un tel système consiste à limiter le nombre de prototypes en mémoire en supprimant ceux qui ont des scores de reconnaissance faibles [5].

[0] C'est une des applications envisagées à l'IDIAP

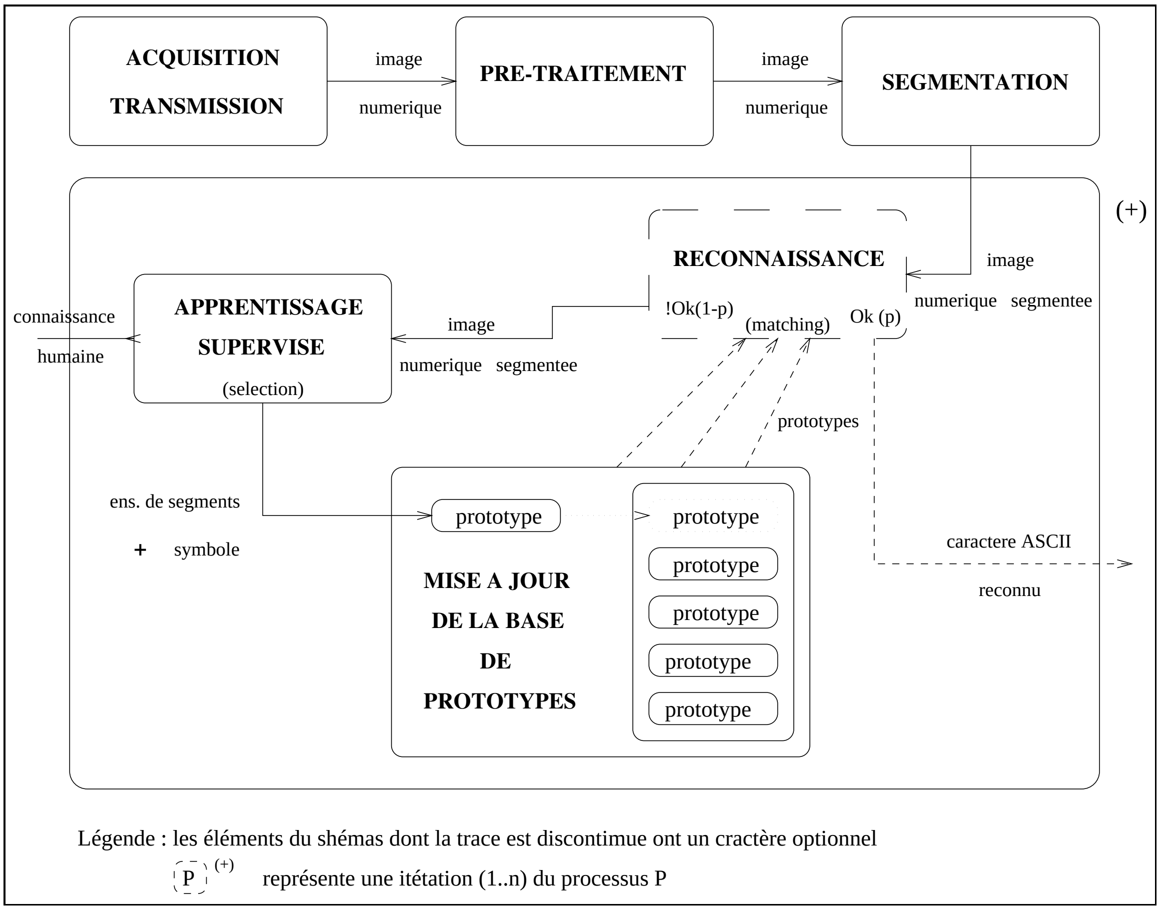

FIG. 1.2 - Apprentissage semi-automatique de prototypes de caractères

1.3.3 Problème à résoudre

On veut réaliser une interface permettant l'apprentissage supervisé de prototypes de caractères à partir de l'image d'un texte manuscrit et avec l'aide d'un opérateur.

1.3.4 Intérêt

- L'exploitation de cette interface permettra de développer une base de connaissance utile pour l'expérimentation et l'amélioration d'un système de reconnaissance de l'écrit manuscrit.
- Commme autre extension possible de ce travail, on peut également citer la production d'un outil pouvant faire partie d'un système utilisé en pratique(ex : dans un outil semi-automatique de transcription d'archives).

Chapitre 2

Apprentissage des prototypes

2.1 Principes

L'apprentissage des prototypes peut se décomposer en quatre sous-traitements (cf schéma 2.1) :

1. pré-traitement : on extrait les bords de l'image binaire du texte.
2. segmentation : elle réalise une sur-segmentation des bords par rapport à la segmentation en caractères. On découpe les bords en segments que l'on définit comme un ensemble de points connexes de l'image des bords pour lesquels l'argument du gradiant est homogène.
3. sélection des prototypes : l'opérateur effectue le regroupement d'un ensemble de segments dont il estime que l'agrégation reconstitue les bords d'un caractère. Il associe ensuite l'instance créée au symbole représentant ce caractère.
4. normalisation des prototypes.

2.2 Etapes du traitement

Soit le tableau F de taille $l \times m$ représentant l'image binaire sélectionnée et tel que :

$$\forall (i,j),\ 0 \leq i < l,\ 0 \leq j < m,\ F(i,j) \in \{0,1\} \tag{2.1}$$

la valeur 0 étant choisie pour les points appartenant aux figures apparaissant dans l'image, la valeur 1 pour les points du fond.

2.2.1 Pré-traitement

Il s'applique au tableau F et comporte deux traitements successifs :

1. Lissage par convolution avec une Gaussienne[1] permettant de réaliser un adoucissement des transitions entre figure et fond (obtention d'une image en niveau de gris) :

$$\tilde{F} = G * F \tag{2.2}$$

2. Extraction des bords par :

   (a) Convolution de l'image avec un masque d'approximation du Laplacien[1].

$$\tilde{F}' = L * \tilde{F} \tag{2.3}$$

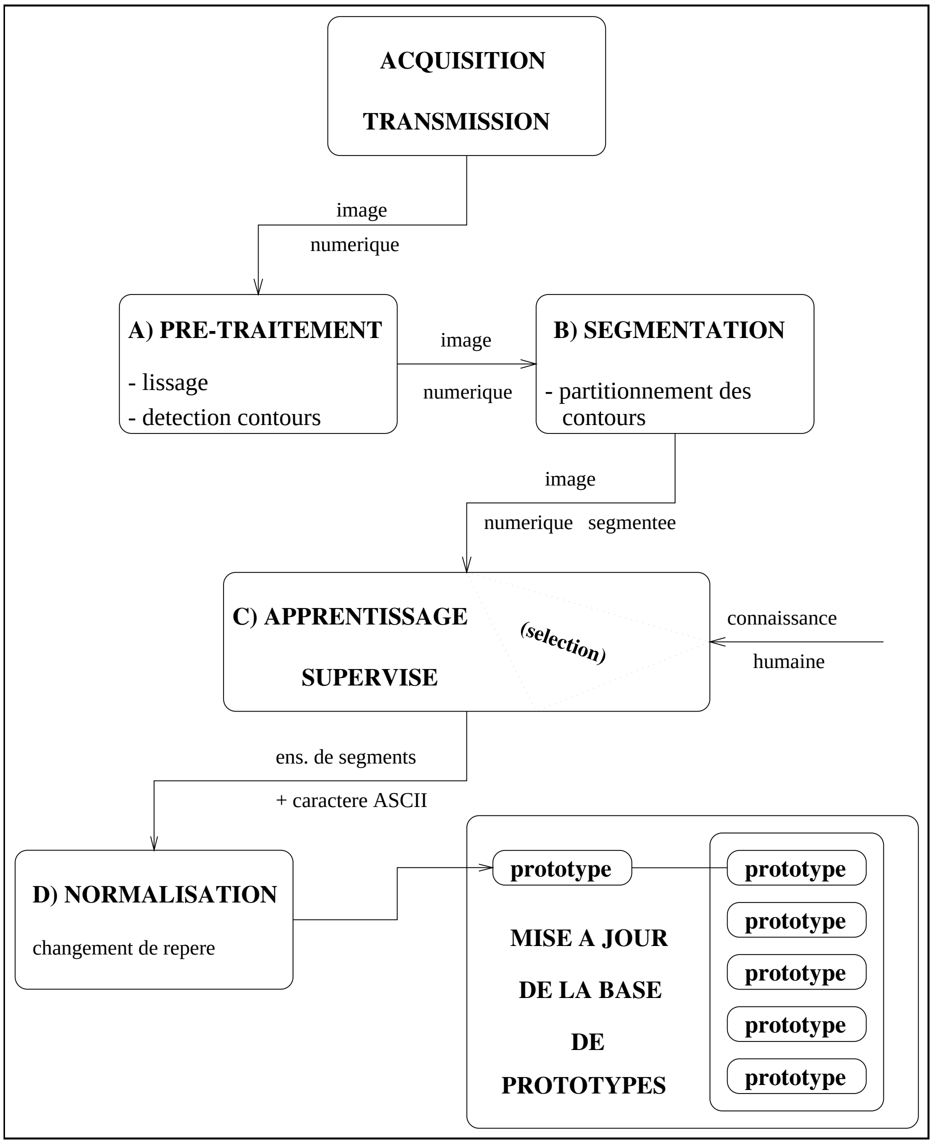

FIG. 2.1 - Chaîne de traitement dédiée à l'apprentissage de prototypes de caractères

(b) Extraction des passages à zéro significatifs du Laplacien.

$$S: \quad \tilde{F}' \quad \rightarrow \quad B_0 \tag{2.4}$$

Formellement, si e désigne le bord,

$$\forall (i,j),\ 0 \leq i < l,\ 0 \leq j < m,$$

$$B_0(i,j) = \begin{cases} e & \text{ssi} \quad |\tilde{F}'(i,j)| > \epsilon\,(\epsilon \text{ fixé}) \ \wedge \ \exists (x,y),\ 0 \leq x < l,\ 0 \leq y < m \mid \\ & \qquad |x-i| \leq 1 \ \wedge \ |y-j| \leq 1 \ \wedge \\ & \qquad |\tilde{F}'(x,y)| > \epsilon \ \wedge \ \tilde{F}'(i,j) \times \tilde{F}'(x,y) < 0 \\ \\ \neg e & \text{sinon} \end{cases}$$

(c) Finalement, application d'un opérateur morphologique d'amincissement du contour réalisé par la procédure "bithin"[1] puisque l'opération effectuée à l'étape 2.4 produit généralement un bord de plusieurs pixels de large.

$$T: \quad B_0 \quad \rightarrow \quad B_1 \ ; \quad \begin{array}{l} \forall(i,j),\ 0 \leq i < l,\ 0 \leq j < m, \\ B_1(i,j) \in \{e, \neg e\} \end{array} \tag{2.5}$$

2.2.2 Segmentation

On segmente l'image des bords en portions de bords, c'est-à-dire en ensembles de points connexes et homogènes vis à vis de l'orientation locale du bord. Le domaine angulaire est divisé en quatre régions de même taille centrées par rapport à l'inclinaison de l'écriture(par la suite, afin de rester général, nous désignons par n le cardinal de la partition réalisée). La valeur 4 semble être le choix le plus naturel : intuitivement, il fait référence à un découpage de l'écrit en directions principales(horizontales et verticales).
Les différentes étapes de ce traitement sont les suivantes :

1. Calcul de l'orientation locale des bords :
   Pour chaque point P de coordonnées (i,j) appartenant à un bord, on calcule l'argument du gradiant θ_{ij} à partir de l'image lissée $\tilde{F}$:

$$\forall(i,j),\ 1 \leq i < l,\ 1 \leq j < m,\ \theta_{ij} = \arctan\left(\frac{\tilde{F}(i,j) - \tilde{F}(i,j-1)}{\tilde{F}(i,j) - \tilde{F}(i-1,j)}\right) \tag{2.6}$$

2. Partition de l'espace d'orientation :
   Soit $\bar{\theta}$ l'estimation de l'inclinaison moyenne du texte :

$$\bar{\theta} = \frac{1}{N} \sum_N \theta_{ij} \tag{2.7}$$

$$\text{avec } N = card\{(i,j), B_1(i,j) = e\}$$

   le partitionnement des points de l'espace d'orientation s'effectue de manière symétrique par rapport à $\bar{\theta}$. Formellement, on associe à tout θ_{ij} l'entier $k_{ij} \in \{0,..,n-1\}$ à l'aide de la fonction $f: R \rightarrow Z$ définie comme suit :

$$\forall \sigma \in R,\ f(\sigma) = \lfloor \frac{\sigma - \bar{\theta}}{2\pi/n} + 0.5 \rfloor \bmod n \tag{2.8}$$

$$\text{avec } \lfloor\,\rfloor : R \rightarrow Z \text{ définie par } \lfloor x \rfloor = \sup\{y \in Z, y \leq x\}$$

   On construit une nouvelle image O définie comme suit :

$$g: \quad B_1 \quad \rightarrow \quad O \ ; \quad \begin{array}{l} \forall(i,j),\ 1 \leq i < l,\ 1 \leq j < m, \\ O(i,j) = \begin{cases} k_{ij} = f(\theta_{ij}) & \text{ssi } B_1(i,j) = e \\ nil & \text{sinon} \end{cases} \end{array} \tag{2.9}$$

3. Définition du segment :
   On propose la définition formelle suivante pour un segment S :
   soit C une composante connexe de B_1 et soit $k \in \{1,2,3,4\}$

$$S = c \text{ la sous-composante connexe de } C \mid \forall(i,j) \in c,\ F(\theta_{ij}) = k \tag{2.10}$$

   Si w est le nombre de segment distincts pris dans B_1, on renarque que :

$$\{(i,j) \mid B_1(i,j) = e\} = \bigcup_{\omega} S_i \tag{2.11}$$

2.2.3 Construction des prototypes

L'opération d'apprentissage d'un caractère consiste à regrouper un ensemble de segments et d'associer cet ensemble à un symbole représentant un caractère de l'alphabet. Un prototype du caractère P comprenant p segments (distincts) qu'on associe au symbole s peut s'exprimer de la façon suivante:

$$P = \{\bigcup_{p} S_i \ , \ s\} \tag{2.12}$$

2.2.4 Normalisation des prototypes

Elle consiste en plusieurs(p) changements de repère. L'origine du repère affecté à chaque point, est le coin inférieur gauche du plus petit rectangle englobant le segment auquel il appartient.

Chapitre 3

Aspects techniques

3.1 Introduction

Dans ce chapitre, nous présentons les résultats concrets obtenus au terme de notre mission. Durant cette période, l'essentiel du temps a été consacré au développement d'un logiciel répondant au cahier des charges qui avait été établi. Dans ce qui suit, nous donnons une courte description fonctionnelle du système ainsi qu'un guide d'utilisation, avant d'aborder en détail son implantation. Au cours de son développement, l'expérimentation du système a mis en avant différents problèmes dont certains, de principe, font dans ce chapitre l'objet d'une étude particulière.

3.2 Description fonctionnelle du système

L'image de la page de texte est affichée à l'écran de sorte à être complètement visible. Il est possible d'y sélectionner une région rectangulaire à l'aide de la souris. L'opération de sélection terminée, la fenêtre d'apprentissage apparaît. la région sélectionnée y est reproduit trois fois à des échelles différentes : une première fois à l'échelle de l'mage complète(réduite), une seconde fois à la taille normale et enfin une troisième fois aggrandie(taille spécifiée lors du lancement du programme ou par défaut 3 pixels écran par pixel image). C'est dans la sélection aggrandie que l'opérateur effectue l'apprentissage : il peut, à l'aide de la souris, sélectionner ou désélectionner un segment, annuler toutes les sélections déjà effectuées, ou bien encore effectuer une désélection point par point; les touches du clavier servent à entrer le symbole du caractère créé. Les symboles (touches claviers) déjà utilisés sont représentés par des icônes. La sélection d'un icône avec la souris engendre l'affichage de tous les prototypes répondant au symbole associé. La sauvegarde est automatique en fin de session et concerne uniquement les prototypes créés lors de la dernière session. Le schéma A.2 fournit une capture écran de l'interface dans sa version actuelle.

3.3 Utilisation du logiciel

3.3.1 Exécution du programme

Pour l'obtenir, il suffit de taper la commande suivante :
Interface <nom du fichier image (format vis)> [-M <coefficient d'agrandisement{1,2,3,4,5}>]

[-R <coefficient de réduction{1,2,3,4}>][-S <coefficient pour la segmentation(float)>])
Remarques :

- les paramètres situés entre crochets ont un caractère facultatif.
- les valeurs par défaut de M, R et S sont respectivement 4, 3 et 0.0.
- le coefficient S paramètre le lissage par hystérèsis est décrit dans la section 3.5.2.1. La valeur par défaut (0.0) inhibe le lissage par hystérésis.

3.3.1.1 Manipulations dans l'image du texte

- Un click sur le bouton de gauche de la souris permet d'amorcer la procédure de sélection pour l'apprentissage. Ensuite, la sélection de l'imagette dans laquelle va se dérouler l'apprentissage s'effectue en pressant une fois le bouton gauche de la souris et en le conservant enfoncé. Dès l'instant où l'opérateur relâche le boutton gauche, la fenêtre d'apprentissage apparaît..(suite dans la section 3.3.1.2).
- Un click sur le bouton central de la souris permet d'amorcer une procédure de visualisation d'une imagette à sa taille normale. La sélection de cette imagette se déroule de la même façon que celle d'une imagette pour l'apprentissage.
- Un click sur le bouton droit de la souris termine la session d'apprentissage et déclenche le processus de sauvegarde des modifications de la base de connaissance pendant cette session.

3.3.1.2 Manipulations dans la fenêtre d'apprentissage

L'image des contours sur-segmentée apparaît magnifiée M fois par rapport à la taille normale (M = 3 par défaut). Les points du contour sont représentés par une des quatre couleur : rouge, vert, violet et jaune suivant le domaine angulaire dans lequel se trouve l'orientation du contour en ce point.

- La sélection d'un segment s'effectue en clickant avec n'importe lequel des boutons de la souris sur un des points du segment choisi par l'opérateur. Ce segment est alors redessiné en gris.
- La désélection d'un segment, opération symétrique à la précédente, s'effectue en clickant à nouveau sur un des boutons. Le segment désélectionné reprend sa couleur initiale.
- Un click avec le bouton gauche de la souris sur l'icône **CORRECTION** permet d'effectuer une désélection point par point lorsque le segment sélectionné par l'opérateur n'appartient pas entièrement au prototype qu'il souhaite créer. Cette situation intervient dans deux cas, soit lorsque la zone recouvrant le segment est bruitée, et lorsque que des caractères se chevauchent ou se touchent(l'algorithme de parcours du contour pour la sélection relie au point initial tous les points connexes(connexité à 8) dont l'argument du gradiant appartient au même sous-domaine angulaire).
- L'annulation de toutes les sélection est déclenchée lorsque l'opérateur clicke avec le bouton gauche sur l'icône **ANNULATION**.
- Un click avec le bouton gauche de la souris sur l'icône **AFFECTATION** permet d'affecter un symbole à l'instance de prototype donnée par l'ensemble des segments sélectionnés. L'opérateur tape au clavier le caractère auquel il souhaite affecter cette

instance(pour les caractères majuscules, il doit avoir préalablement activé le mode majuscule du clavier utilisé (avant le click)). La validation est réalisée en tapant une fois sur la barre d'espace. Une demande de confirmation apparaît dans la cellule de l'interface prévue à cet effet, à laquelle l'opérateur doit répondre par oui (touche o) ou par non(choix par défaut). Dans le premier cas, une nouvelle fenêtre affiche le prototype de caractère créé à sa taille réelle. Ensuite celui-ci est rajouté à la base de connaissance. Enfin le système réinitialise la sélection dans la fenêtre d'apprentissage.

- En clickant sur l'une des icônes estampillées d'un caractère ASCII, l'opérateur obtient l'affichage d'une fenêtre contenant l'ensemble des prototypes répondant à ce symbole.
- Un click avec le bouton gauche de la souris sur l'icône **QUITTER** permet de sortir de la fenêtre d'apprentisage et de revenir à l'image du texte, la sélection en cours étant alors abandonnée.
- Un click sur l'icône **HYSTERESIS** permet de lancer un processus de sur-segmentation de la sélection, dont nous justifions l'ajout dans la plus récente version du programme dans la section 3.5.2.3.

3.4 Implantation du système

3.4.1 Architecture du système

3.4.2 Architecture logicielle

Le schéma 3.1 présente une décomposition hiérarchique du logiciel en modules externes, modules propres, sous-modules et fonctions de base ainsi qu'une décomposition hiérarchique des liens de dépendance logiciel(relations de mise en oeuvre).
Remarque : la bibliothèque C standard (*stdio, stdlib, stddef, tring*) n'y figurent pas, par soucis de clareté.

3.4.3 Bibliothèques de logicielles utilisées

L'implantation de l'interface d'apprentissage est réalisée en C - l'environnement d'exécution est X-windows s'exécutant sur SparcStation. On utilise deux librairies de routines C existant déjà à l'IDIAP :

- VISLIB, dédiée au traitement de l'image [16].
- WINLIB, dédiée principalement à la visualisation d'images, elle permet une utilisation pratique des primitives X-windows.

3.4.4 Format des images traitées

Le format VIS[16] s'impose du fait de l'utilisation de la bibliothèque VISLIB.
Remarque : après acquisition de l'image d'un texte, nous disposons d'un fichier au format TIFF qu'il faut alors convertir au format VIS. Nous utilisons à cet effet l'utilitaire *tiff2vis*, qui a cette fonctionnalité.

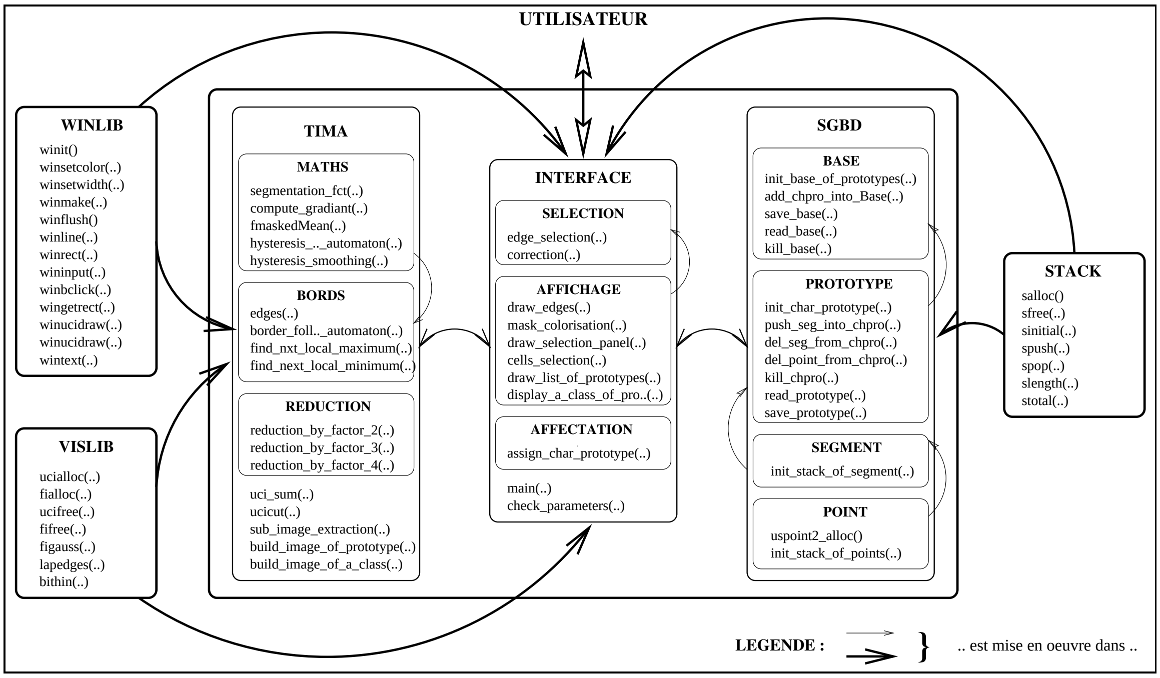

FIG. 3.1 - Architecture logicielle du système

3.4.5 Structures de données mises en oeuvre

Elles sont définies en langage C dans le fichier *segment.h*; la structure de donnée(définie en langage C dans le fichier *segment.h*) se décompose en plusieurs entités hiérarchiques. Le plus souvent, ces entités mettent en oeuvre un ensemble de piles dynamiques pouvant être gérées comme des tableaux et pour lesquelles la réallocation est effectuée, si nécessaire, lors de l'empilement (définition et implantation sont faites repectivement dans les fichiers *stack-type.h et stack.c*).
Les entités sont :

- *la base* : c'est une pile (un pointeur de pile pour chaque classe de prototypes) de piles de protototypes.
- *le prototype* : c'est la réunion des composants suivants:
  - un identificateur pour le symbole associé au prototype (caractère ASCII(pour l'instant) ou chaîne de caractères).
  - le nom du fichier associé au prototype.
  - un tableau de 4 piles de segments(une pour chaque classe d'orientation du gradiant).
  - un tableau de 4 piles d'entiers pour les décalages respectifs des segments du prototype selon l'axe des x dans le repère choisi(bord inférieur gauche de l'image sélectionnée ou après normalisation, coin inférieur gauche du plus petit rectangle recouvrant le segment courant).
  - un tableau de 4 piles d'entiers pour les décalages respectifs des segments du prototype selon l'axe des y.
- *un segment* : c'est une pile de points.
- *un point* : c'est un couple d'entiers : les coordonnées du point.

Enfin, il existe une table qui contient, pour chaque symbole, l'index correspondant dans la base ainsi que le nombre d'ajouts effectués lors de la dernière session d'apprentissage.

3.4.6 Structure de la base de donnée sur disque

Pour chaque prototype créé et normalisé, la sauvegarde est effectuée en fin de session, dans un fichier dont le nom respecte la syntaxe suivante :
$< nom\ de\ fichier >$::= $CP\_ < symbole > \_ < numéro >$
$< symbole >$::= Caractère ASCII
$< numéro >$::= le plus petit entier tel que celui-ci ne soit pas utilisé comme champ $< numéro >$ dans un fichier correspondant à un prototype de même classe
Cette grammaire procure une sécurité suffisante car il n'y a pas de réécriture d'un fichier. D'autre part, on comble les trous pouvant apparaître, après suppression de prototype, dans la suite des numéros de fichiers d'une classe.
Dans ce fichier, les données sont organisées comme suit:

- le symbole du prototype (caractère ASCII)
- respectivement pour chaque classe d'orientation du gradiant, et respectivement pour chaque segment de la classe,
  - décalage du segment en x [RET][1]
  - décalage du segment en y [RET]
  - nombre de points du segment [RET]
  - copie du bloc mémoire associé à la pile de points du segment (donc dans un format binaire qui permet une économie de place)[RET].

La base est décrite dans le fichier texte *Base_descr*. C'est ce fichier qui permet la reconstruction de la base lors de la session d'apprentissage suivante. On y trouve :

- le nombre de classes de prototypes dans la base [RET]
- pour chaque classe de prototypes,
  - le nombre d'éléments de la classe [RET]
  - la liste des noms de fichier des différents prototypes de la classe séparés par [RET]

3.5 Expérimentation du système

3.5.1 Construction d'une base de connaissance

Ce travail succède à la phase de développement et de validation du logiciel qui a été réalisé et correspond au début de son exploitation dans une situation réelle. C'est une tâche assez longue et pénible mais que l'on a voulu rendre la moins fastidieuse possible pour l'opérateur en implantant une interface d'apprentissage conviviale. Il est important de lui accorder une attention particulière car nous mesurons au combien elle est nécessaire pour l'expérimentation de nouvelles méthodes de reconnaissance.
Les documents à partir desquels débutent l'apprentissage sont des textes extraits des archives Valaisannes. L' annexe A.1 illustre les premiers apprentissages réalisés sur une série de caractères.

[1] retour chariot

3.5.2 Défaillances observées du système

3.5.2.1 Défaillance de la segmentation aux frontières de la partition du domaine angulaire

(a) *Enoncé du problème*

Lorsque l'orientation d'un bord oscille autour de l'orientation moyenne $\bar{\theta}$ *modulo* $\pi/2$, on observe une sur-segmentation des bords. En effet, ces oscillations se traduisent par autant de petits segments qui, pris isolément apportent peu d'informations , alourdissent les modèles créés et les rendent peu performants du fait de la grande variabilité intra-prototype qui résulte de cette sur-segmentation. Celà représente de plus une surcharge de travail du point de vue du nombre d'opérations de sélection nécessaires à l'apprentissage. L'annexe A.4 illustre ce phénomène.

Nous voulons rassembler ces petits segments sous un label unique pour former un segment plus représentatif.

(b) *Solution proposée*

Notre choix s'est arrêté sur un lissage par hystérésis des valeurs de l'argument du gradiant pour les points des bords. C'est un choix motivé entre autres par l'existence de références dans la littérature [1] et par l'intérêt que représente son implantation; de nombreuses autres méthodes répondent au problème posé et pourraient être implantées et testées dans la suite de ce travail.

i. *principes :*

Le lissage par hystérésis est un algorithme itératif qui mesure l'importance des variations locales des valeurs des points du domaine étudié(d'une image, d'un bord, ..) et ignore celles qui sont inférieures à un seuil paramétrable [1]. La figure 3.2 illustre ce mécanisme dans un cas pratique(seuil 0.5 radians, 3 itérations).

La machine possède deux états UP et DOWN marquant respectivement la croissance et la décroissance de la fonction f qui donne les valeurs des points du domaine étudié. Si pour le point situé à la position $n+1$, l'état est DOWN et on observe que $f(n+1) \leq f(n)$, alors l'état reste DOWN et la valeur de sortie du système $g(n+1)$ est affectée à $f(n+1)$. De même si l'état est UP et $f(n+1) > f(n)$, alors l'état reste UP et $g(n+1) = f(n+1)$.

Si en revanche l'état est DOWN et $f(n+1) > f(n)$, soit j_0 le plus petit entier tel que $f(n+j_0) > f(n+j_0+1)$, faisant du point situé à la position $n+j_0$ le prochain maximum local; si $f(n+j_0)-f(n) < h$, indiquant que le maximum local représente une variation négligeable, alors l'état reste DOWN et $g(n+1)$ prend la valeur $g(n)$. Par contre si jamais $f(n+j_0)-f(n) \geq h$, indiquant que le prochain maximum local représente une variation significative, alors l'état devient UP et $g(n+1)$ prend la valeur $f(n+1)$.

Parallèlement, si l'état est UP et $f(n+1) \leq f(n)$, soit k_0 le plus petit entier tel que $f(n+k_0) < f(n+k_0+1)$, faisant du point situé à la position $n+k_0$ le prochain minimum local; si $f(n)-f(n+k_0) < h$, indiquant que le minimum local représente une variation négligeable, alors l'état reste UP et $g(n+1)$ prend la valeur $g(n)$. Par contre si jamais $f(n)-f(n+k_0) \geq h$, indiquant que le prochain minimum local représente une variation significative, alors l'état devient DOWN

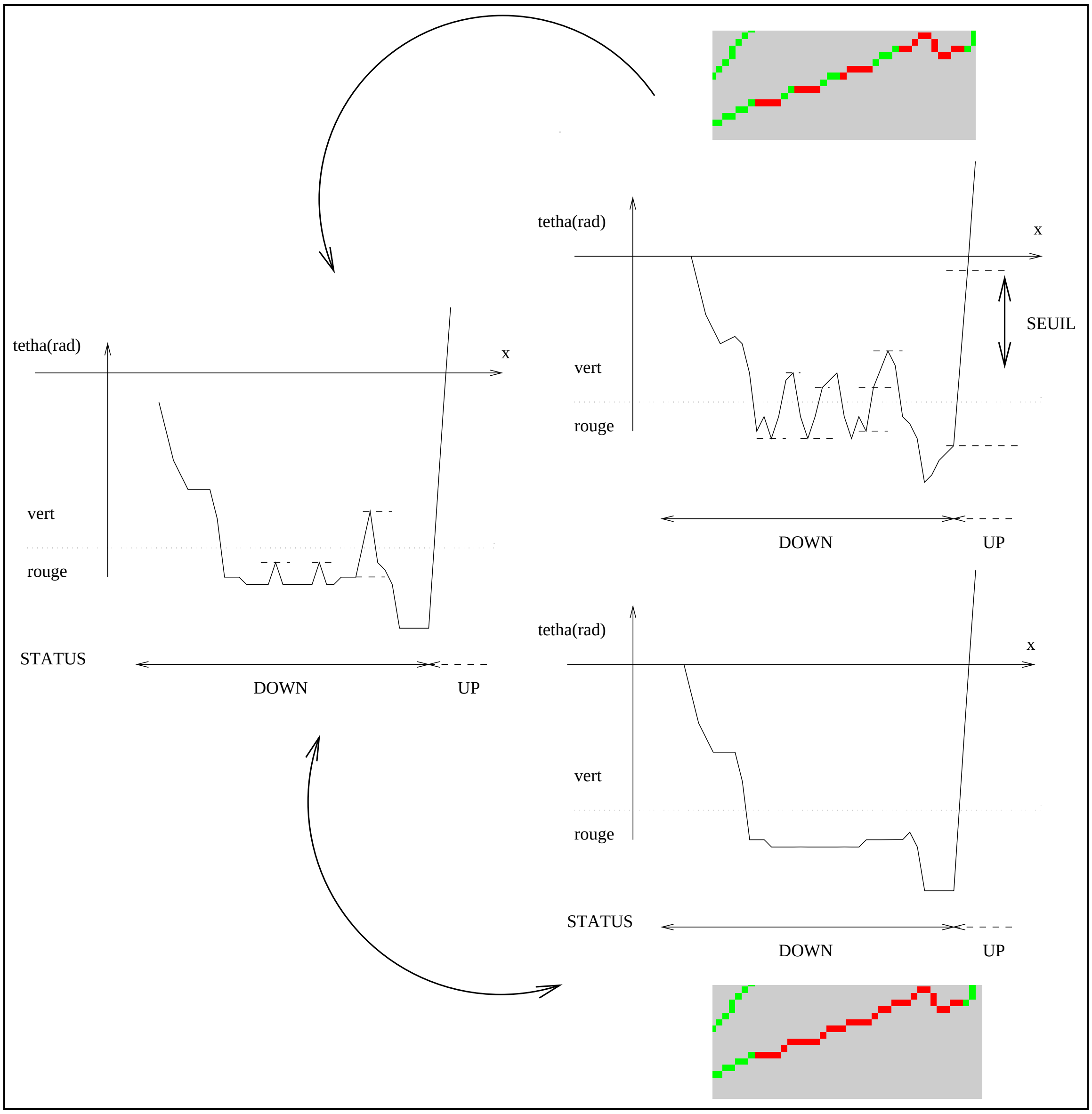

FIG. 3.2 - Illustration du lissage par hystérésis de l'argument du gradiant pour les bords

et $g(n+1)$ prend la valeur $f(n+1)$.
Les tableaux 3(b)i et 3(b)i dressent un récapitulatif du comportement de l'automate d'hystérésis.

ii. *mise en oeuvre*
L'implantation de ce lissage fait coopérer 2 automates :
   - l'automate d'hystérésis que nous venons de décrire.
   - un automate de suivi de bord (suivi de lisière, "crack following") :
     L'automate est défini par 4 points $\{P, Q, U, V\}$ formant un masque carré et tels que : $P \in bord$, $Q \in fond$, U et V sont 4 adjacents respectivement à P et à Q dans le sens d'orientation(sens inverse des aiguilles d'une montre). Ces contraintes permettent de définir 4 masques possibles que l'on classe selon le sens d'orientation et auxquels ont attribue un code (cf figure 3(b)ii). Le tableau 3.3 illustre le fonctionnement de l'automate(Les sorties figurent en

Condition	Entrée
$f(n+1) > f(n), f(n+j_0 - f(n) \geq h$	a
$f(n+1) > f(n), f(n+j_0) - f(n) < h$	b
$f(n+1) \leq f(n), f(n) - f(n+k_0) < h$	c
$f(n+1) \leq f(n), f(n) - f(n+k_0) \geq h$	d

TAB. 3.1 - Entrées de l'automate

Etat courant	a	b	c	d
UP	UP $f(n+1)$	UP $f(n+1)$	UP $g(n)$	DOWN $f(n+1)$
DOWN	UP $f(n+1)$	DOWN $g(n)$	DOWN $f(n+1)$	DOWN $f(n+1)$

TAB. 3.2 - Spécification des sorties de l'automate pour la position $n+1$

caractères gras).

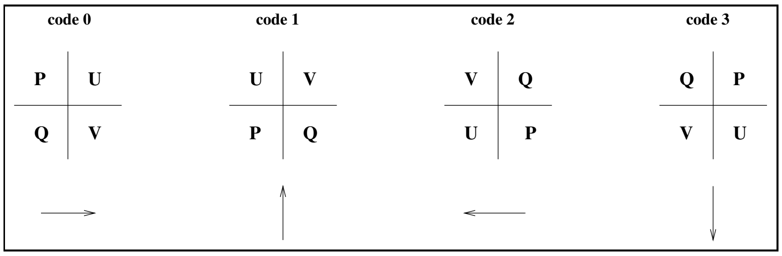

FIG. 3.3 - Masques possibles pour l'automate de suivi de contours

Détection des bords :
On effectue une recherche de points des bords de la gauche vers la droite et de bas en haut. Lorsqu'on atteint un point d'un bord, on vérifie que celui-ci n'a pas encore été parcouru; si c'est le cas, on déclenche l'automate de suivi de bord avec le *code* 3, ceci conjointement à l'automate de lissage.

iii. *Utilisation*
Dans la fenêtre d'apprentissage, l'opérateur peut déclencher le lissage en clickant avec le bouton gauche de la souris sur l'icône HYSTERESIS. Les bords sont alors réaffichés. L'opérateur peut renouveler l'opération autant de fois qu'il le souhaite jusqu'à ce que la segmentation réalisée lui semble correcte. Il a également la possibilité de réafficher la segmentation initiale en clickant sur l'icône ANNULATION.
Remarques :
- Expérimentalement, les valeurs du seuil h (correspondant au paramètre S donné à l'exécution de l'application), pour lesquelles on obtient les meilleurs résultats pour la segmentation, sont des valeurs proches de 0.5 radians.
- L'annexe A.5 résulte de l'exécution de trois itérations de lissage (h= 50 radians) sur la sélection prise dans la figure A.4 et témoigne d'une efficacité satisfaisante.
- Parmi les défauts à créditer au lissage que nous opérons, citons le suivant : ce lissage dépend du sens de parcours des bords. En parcourant le bord dans le sens inverse, on obtiendrait une segmentation sensiblement différente.

U	V	**P+1**	**Q+1**	Tourner	**code généré**
*	*bord*	V	Q	à droite	$code - 1 \bmod 4$
bord	*fond*	U	V	non(continuer)	code
fond	*fond*	P	U	à gauche	$code + 1 \bmod 4$

TAB. 3.3 - Spécifications des sorties de l'automate de suivi de bord

3.5.2.2 Sous-segmentation des bords

(a) *Enoncé du problème*
Ce phénomène se produit notamment pour les segments reliant deux caractères et a des conséquences gênantes pour l'apprentissage(ex. : annexe A.6); dans ce cas la sélection de ce segment et son ajout aux composantes de tel ou tel prototype, ou son oubli posent problème à l'opérateur !

(b) *Solutions proposées*
Plusieurs comportements de l'opérateur permettent de répondre au problème posé :

i. on décide de n'affecter le segment de liaison à aucune des instances de prototype qu'il relie; on préfère créer une classe de prototype pour les liaisons ou mieux n classes contextuelles de prototypes de liaison pour lesquelles un contexte est formé d'un couple de caractères (ex : pour la figure de la l'annexe A.6, on associe le segment (blanc à un prototype de label c-o).Cette dernière solution à l'avantage d'avoir une exploitation relativement simple, au niveau syntaxique dans la partie reconnaissance.

ii. on décide d'affecter systématiquement le segment à toutes les instances de prototype auxquelles on a le sentiment qu'il appartient. Dans ce cas, on conserve plus d'information que nécessaire sur les instances de prototype connectées par ce segment et la redondance qui en découle peut s'avérer gênante pour la reconnaissance.

iii. on décide d'affecter systématiquement le segment à l'instance de prototype située immédiatement à sa gauche.Le problème de redondance du segment dans plusieurs instances de prototype ne se pose plus mais par contre, cette solution occasionne une variabilité intra-prototype accrue qui n'est certainement pas très favorable à la reconnaissance.

iv. on décide d'effectuer un découpage du segment de liaison puis on affecte sucessivement chacun des deux sous-segments créés de la manière suivante : celui de gauche à l'instance de gauche et celui de droite à l'instance de droite. Pour réaliser cette solution avec l'interface d'apprentissage dont nous disposons, l'opérateur peut utiliser la désélection manuelle point à point en activant avec la souris le mode CORRECTION. Les limites de cette solution sont posées par la difficulté de définir précisément la frontière entre deux caractères liés.

Dans l'état actuel de l'exploitation de l'interface d'apprentissage, c'est cette dernière procédure qui est opérée(voir annexe A.7)

3.5.2.3 Défaut de segmentation observé en bordure de l'image sélectionnée

(a) *Enoncé du problème*
En bordure de l'image sélectionnée, on peut observer que lorque l'empreinte graphique est coupée dans la fenêtre sélectionnée, un bord factice est créé à l'endroit de la

coupure(cf figure A.8). A noter que celà se produit uniquement pour le bord droit et le bord supérieur, c'est-à-dire lorsque le calcul du gradiant(cf Section 2.2.2, Equation 2.6) peut s'opérer. Ce n'est pas le cas pour les points d'abcisse nulle et d'ordonnée nulle qui correspondent respectivement aux points du bord gauche et du bord inférieur de l'image.
D'autre part, on observe que l'orientation attribuée aux segments concernés est erronnée (on constate en pratique que cette erreur correspond à un décalage de π pour l'argument du gradiant en chaque point des segments).
Explication:
La création d'un bord fictif ,lorsque qu'une figure déborde de l'image, est un effet de bord du calcul de $\tilde{F}$. En effet, ce calcul est réalisé par la convolution de F avec une Gaussienne. Par conséquent, tous les points de $\tilde{F}$ en bordure d'image ont des valeurs basses puisque les points en dehors de la sélection sont pondérés à zéro pour ce calcul. ce qui occasionne l'apparition de minima locaux qui produisent le bord .
Le décalage de π pour l'argument du gradiant a la même cause et correspond à un changement de signe pour le numérateur ou le dénominateur de l'argument de arctan dans la formule 2.6 de la section 2.2.2.

(b) *Solution proposée*
Il faut effectuer les calculs de la gaussienne, du Laplacien et du gradiant sur une une image plus grande que l'image sélectionnée, lorsque celà est possible c'est à dire lorsque l'image sélectionnée n'est pas elle même trop près du bord de l'image initiale (F).

Chapitre 4

Conclusions

4.1 Récapitulatif

Le problème qui nous a été posé lors de ce stage consistait à réaliser une interface d'apprentissage des caractères à partir de documents manuscrits. Tenant compte des orientations antérieures choisies par l'équipe image à l'IDIAP dans le cadre de la conception et de la réalisation d'un système de reconnaissance de caractères robuste, en particulier de l'approche "contour" choisie pour la représentation d'une instance de caractère, nous avons résolu le problème de l'isolement d'un caractère de la manière suivante : les contours sont segmentés par rapport à une classification de leur orientation en quatre catégories de direction principales. Le résultat est une sur-segmentation de l'image par rapport à la segmentation en caractères. L'opérateur regroupe ensuite les segments pour former une instance de prototype à laquelle il affecte un symbole.
Lors de l'expérimentation dans une version minimale, nous avons mis en avant des défaillances dont souffrait la segmentation réalisée, notamment des problèmes de sur-segmentation lorque l'orientation du contour se situe au niveau des frontières de la partition du domaine angulaire. Nous avons résolu ces problèmes de manière satisfaisante en implantant un lissage par hystérésis sur les valeurs d'orientation des bords. On a pu également observer une sous-segmentation des bords des segments reliant deux caractères, et qui est gênante pour l'attribution à un prototype. Nous avons résolu le problème en implantant un mode de désélection manuelle point par point dans l'interface permettant ainsi à l'opérateur d'isoler tout de même un caractère.
Enfin, nous avons expérimenté que la version finale de l'interface permet de réaliser avec succès l'apprentissage.

4.2 Perspectives pour la reconnaissance

Pour chaque prototype de caractère appris, on dispose d'un ensemble de segments qui forment son contour. Pour la reconnaissance, les perspectives sont assez nombreuses. On peut utiliser une représentation complète du contour et appliquer par exemple les méthodes robustes de reconnaissance développées à l' IDIAP [5]. L' alternative consiste à utiliser une représentation réduite des bords, en exploitant par exemple la segmentation effectuée dans notre système d'apprentissage. Pour cela, nous devons résoudre les problèmes suivants :

- le choix d'une représentation pour les segments(par ex. calcul du moment).
- le choix d'une représentation pour la structure des prototypes de caractère (par ex. utilisation des graphes).
- le choix d'une méthode de reconnaissance utilisant les représentations que nous venons d'énumérer.

Bibliographie

[1] Robert M. Haralick and Linda G. Shapiro - Computer and Robot Vision volume I

[2] George Wolberg - Digital Image Warping - IEEE Computer Society Press Monograph

[3] David F. Rogers - Procedural Elements for Computer graphics

[4] Thomas M. Breuel - A System for the recognition of handwritten text - IDIAP Martigny Switzerland

[5] Gilbert Maître - Experiments with robust similarity measures for OCR - IDIAP TR95-03

[6] S. Impedovo - Fundamentals in Handwritting Recognition - Dipartemento di informatica, universita di Bari, Italy

[7] R.H Davis and J Lyall - Recognition of handwritten characters, a review - department of computer science, university of Edinburgh

[8] B.N Chatterj - Feature Extraction methods for character recognition - Department of Electronics and electrical communication engineering, indian institute of technology, Kharagpur 721 302, India

[9] A.W. Senior - Of-line handwritting Recognition : a review and experiments - Cambridge University, Engeneering Department, England

[10] H.L. Teuling and L.R.B. Schomaker - Unsupervised learning of prototype allographs in cursive script recognition - From Pixel to features III

[11] C.A. Higginss and D.M. Ford - A new segmentation method for cursive script recognition - From Pixel to features III

[12] K. Yamamoto, H.Yamada and T. saito - Current state of recognition method for japanese characters and database for research of handprinted character recognition - From Pixel to features III

[13] S. Impedovo - Automatic reading of typed/handwritten numerals - From Pixel to features III

[14] J.C. simon and O. Baret - Cursive word recognition - From Pixel to features III

[15] B.Taconet - A new global off-line recognition method for handwritten words - From Pixel to features III

[16] T.M. Breuel - User's guide to the VIS Vision Library - idiap 1992

Annexe

Annexes A

Figures

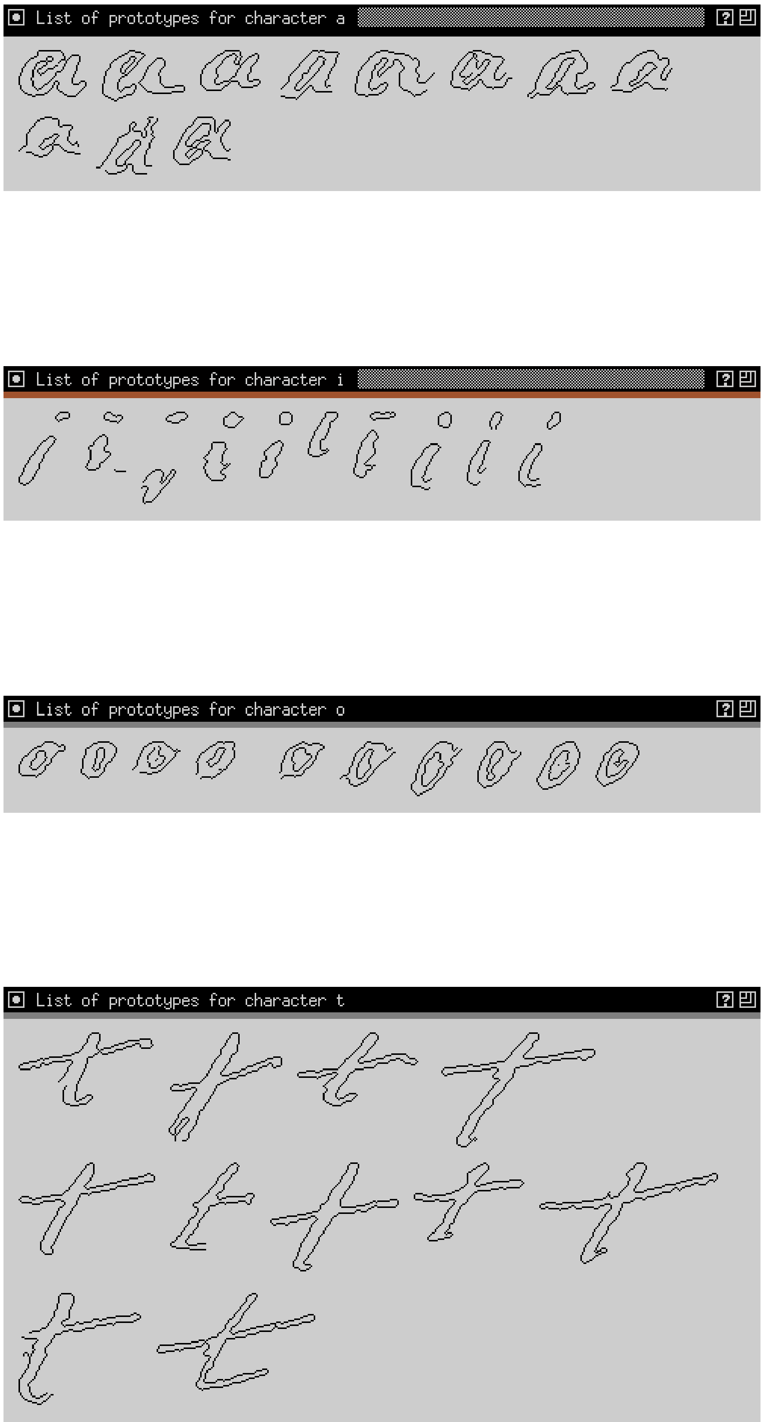

Fig. A.1 - Echantillons extraits de la base de connaissance

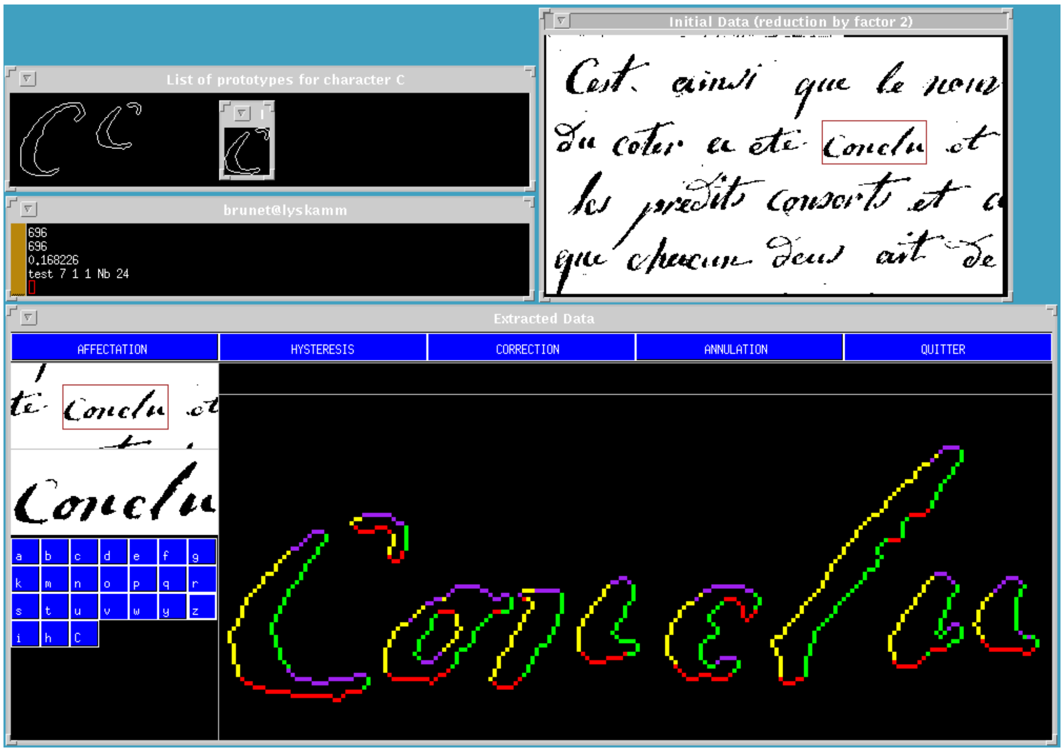

Fig. A.2 - L'interface graphique (fenêtre d'apprentissage)

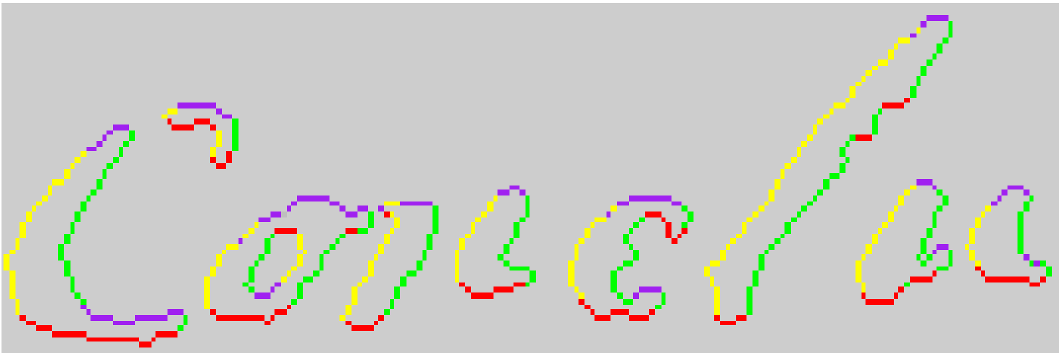

Fig. A.3 - Segmentation correcte du texte

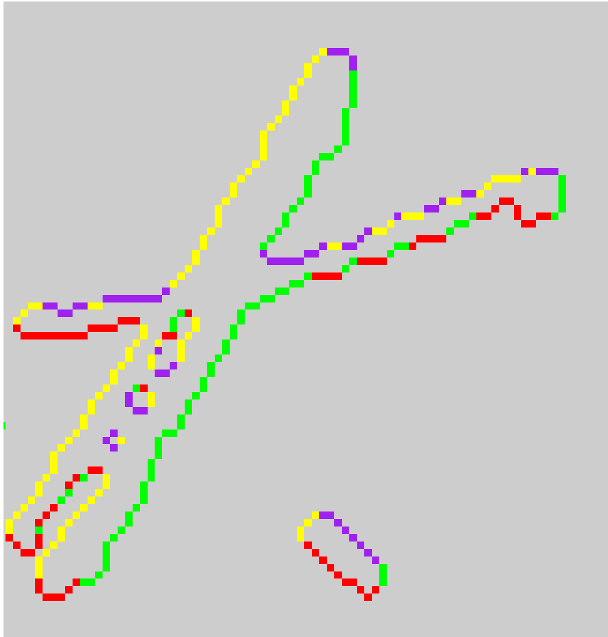

Fig. A.4 - Sur-segmentation des bords aux frontières de la partition du domaine angulaire

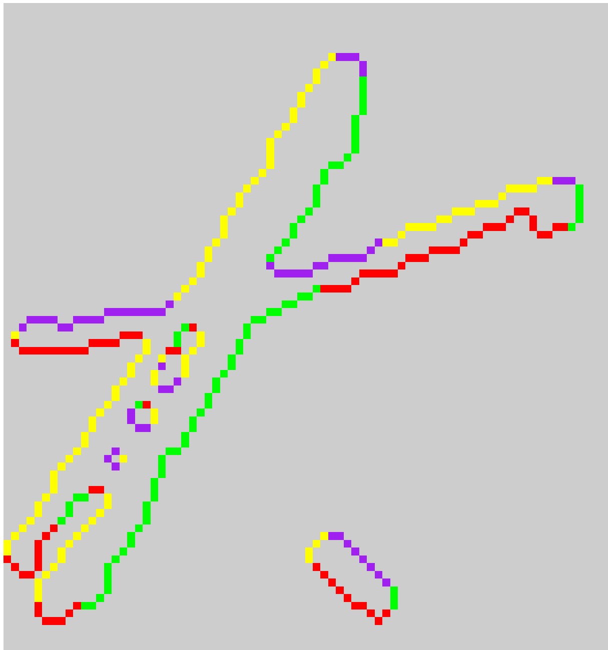

Fig. A.5 - Meme image après application du lissage par Hystérésis

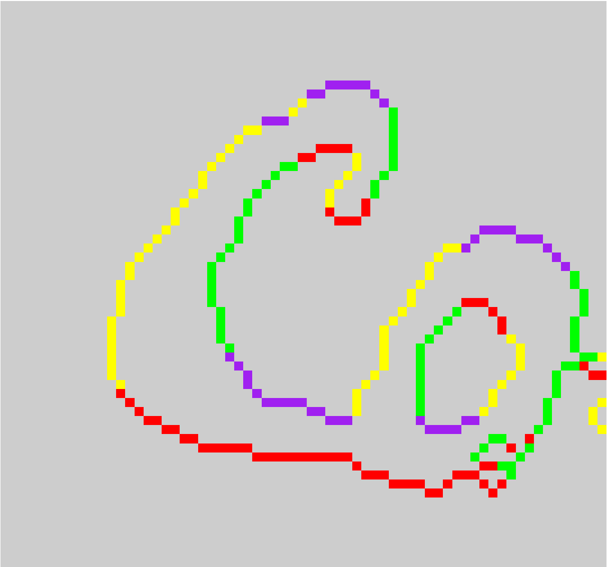

Fig. A.6 - Illustration de la sous-segmentation

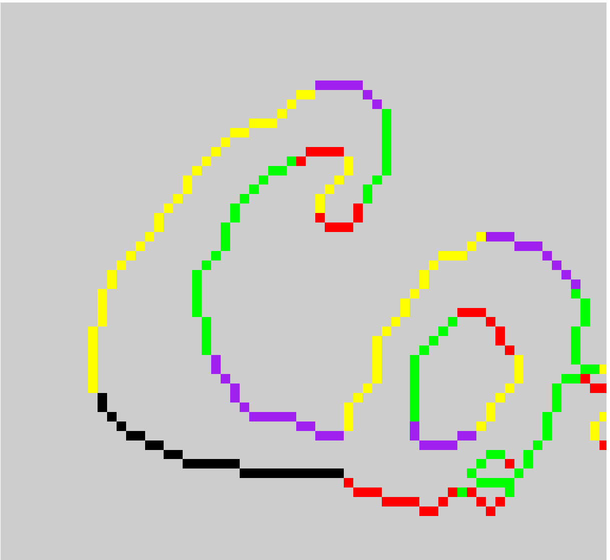

Fig. A.7 - Correction manuelle de la sélection répondant au problème de sous-segmentation

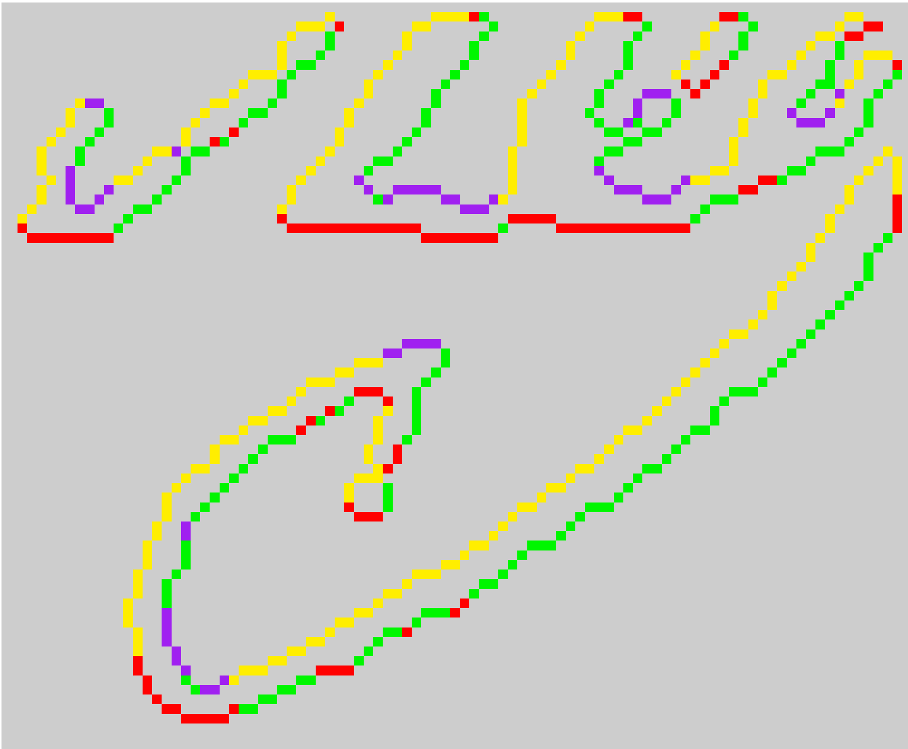

Fig. A.8 - Illustration des défaillances de la segmentation en bordure de l'image sélectionnée