Commandes gestuelles interactives

Rafik Sekkal

UMR IRISA

Université de Rennes1
IFSIC
Master Recherche en Informatique
Parcours : Des données aux connaissances

Rapport de stage

Commandes gestuelles interactives

par

Rafik Sekkal

Maîtres de stage : Eric Anquetil
Adrien Delaye
Équipe de recherche : IMADOC (IRISA)

Rennes, 2010

Table des matières

Introduction

L'informatique nomade a connu un très grand progrès ces dernières années avec l'utilisation de différents types de périphériques ; on peut les trouver sous diverses formes : ordinateurs tablettes (Tablet PC), assistants personnels numériques (PDA), téléphones mobiles intelligents (Smartphone), etc. Ces terminaux combinent à la fois les fonctionnalités d'affichage d'un écran ordinaire et celle d'un dispositif de pointage. Ils permettent donc de réduire le nombre de périphériques (entrée/sortie) nécessaires pour la manipulation d'un système.

L'interaction homme machine se fait à travers des commandes définies par les concepteurs des systèmes afin d'exécuter des fonctionnalités prédéfinies. Une commande est un signal, un mot ou une série de symboles qui donnent à l'ordinateur l'ordre de démarrer, d'arrêter ou de continuer une opération ou une instruction. La première approche d'interaction était les interfaces en mode ligne de commande. Elles reposent sur une interaction en mode texte où les commandes sont représentées par des mots clés précédés par des arguments (par exemple : cd user, ifconfig), ou des raccourcis clavier (Ctrl+a, Ctrl+s). L'utilisateur est donc appelé à taper sa commande et attendre le résultat final du système. Une autre approche se base sur les menus interactifs qui offre à l'utilisateur une interaction visuelle pour l'aider à choisir sa commande et l'exécuter. Par exemple, dans le menu "Démarrer" de "Windows", il suffit de survoler un menu pour qu'un commentaire apparaisse, et que les sous-menus s'ouvrent automatiquement. L'activation de la commande pour les menus interactifs se fait en cliquant sur l'item choisi.

En parallèle, un nouveau type de commandes commence à apparaître sur les nouveaux périphériques cités précédemment : les commandes gestuelles qui consistent à déclencher une commande en traçant un geste spécifique qui lui est associé et prédéfini par les concepteurs du système. Dans la plupart de ces périphériques, les commandes gestuelles ne présentent pas d'interaction, l'utilisateur dessine son geste sur les données souhaitées (par exemple : sélectionner du texte, puis le supprimer) et le système reconnaît la commande et l'exécute. On peut distinguer les derniers périphériques mobiles d'*Apple (iPhone, iPad)* qui offrent un ensemble de commandes gestuelles qui sont en plus interactives comme le zoom ou la barre de défilement. Les commandes gestuelles offrent d'une part une facilité d'utilisation car elles ne nécessitent pas trop de manipulations en comparaison avec les méthodes standards (clavier, souris). D'autre part, la saisie est plus naturelle puisque les gestes sont tracés manuellement et peuvent être conçus suivant les habitudes de l'être humain (par exemple : le geste "Supprimer" consiste à faire des allers retours d'un point à un autre avec la notion de "rature").

Cependant, l'utilisation des interfaces gestuelles qui utilisent un nombre important de commandes (comme les interfaces dédiées au dessin des schémas électriques ou la composition musi-

cale) nécessitent un apprentissage du jeu de gestes qui risque de demander un temps considérable à l'utilisateur. Les concepteurs de ce type d'interfaces facilitent l'apprentissage en proposant différentes approches : la première consiste à définir un ensemble de gestes dont la forme est liée à la nature de la commande. Ceci aide l'utilisateur à se souvenir de la forme de geste en l'associant à la nature de la commande souhaitée. Une autre approche consiste à afficher une liste exhaustive de l'ensemble des commandes à l'appel de l'utilisateur. La dernière approche et celle que nous allons explorer repose sur un mécanisme d'interaction destiné à assister l'utilisateur dans l'apprentissage des commandes d'une façon implicite au moyen d'un menu sous-jacent conçu pour l'aider à mémoriser les gestes. Néanmoins, les systèmes existant présentent dans la plupart des cas des limites dans la phase d'apprentissage des gestes d'où l'intérêt de notre stage qui est destiné à aider l'utilisateur à mémoriser l'ensemble des gestes d'une manière rapide et naturelle.

L'objectif du stage est de concevoir un mécanisme d'apprentissage des tracés associés aux commandes gestuelles sur un ordinateur tablette. Pour cela, nous allons utiliser les menus interactifs gestuels qui permettent d'aider l'utilisateur à mémoriser le geste grâce à une utilisation répétée. Un menu gestuel interactif est un dérivé gestuel des menus interactifs simples : l'activation de la commande se fait non pas en cliquant sur l'item choisi mais en traçant un geste. Le but de ce menu est d'aller vers une interface purement gestuelle où l'activation de la commande est basée sur une reconnaissance gestuelle du tracé sans l'intervention du menu interactif.

Le principe du menu est d'indiquer à l'utilisateur la forme du geste graphique à travers des points cibles à atteindre et des zones par lesquelles il faut passer pour activer la commande. Ce menu offre une interaction continue à l'utilisateur en lui rendant des résultats visuels intermédiaires au moindre mouvement effectué afin de l'inciter à poursuivre son geste ou à annuler l'action. L'apprentissage du vocabulaire de gestes se fait d'une manière implicite pour l'utilisateur. À force d'une manipulation répétée du menu, il commence à mémoriser le geste qui suit une forme régulière correspondant aux directions choisies. La topologie des menus permet d'induire un vocabulaire de gestes dont la forme est régulière, fluide et facilement identifiable dans la phase de reconnaissance gestuelle. Une fois l'ensemble des gestes appris, l'utilisateur peut exécuter sa commande sans faire appel aux menus en traçant le même geste qui a servi dans la phase d'apprentissage.

Dans la suite du rapport, la première section présente un rappel sur les différentes méthodes de représentations et d'apprentissage de menus interactifs gestuels. Ensuite, la deuxième section présente le concept proposé, la topologie du menu, les différentes modes d'interactions et leur impact sur la forme des gestes. Dans la troisième section on détaille la mise en œuvre de l'algorithme de reconnaissance gestuelle. La quatrième section présente les expérimentations et les résultats obtenus au cours de ce stage. Dans la dernière section, on conclut en présentant les perspectives et les travaux futurs qui restent à faire.

Chapitre 1

Méthodes de représentation et d'apprentissage des commandes

Dans ce chapitre, nous allons présenter les différentes approches de représentation des menus pour l'exécution des commandes, notamment les menus contextuels qui sont les premiers menus graphiques utilisés dans les systèmes informatiques. Ensuite, nous développons les différents modes d'interaction lors de l'utilisation de menus gestuels à savoir le mode novice et le mode expert. Puis nous présentons quelques approches proposées dans la littérature concernant les menus gestuels interactifs qui sont le type de menus que nous proposons d'exploiter pour remédier aux problèmes d'apprentissage dans le cas de commandes gestuelles. Nous présentons à la fin les limitations de chaque approche, et les solutions que nous avons envisagé.

1.1 Menus contextuels

Selon le modèle WIMP *(Windows, Icons, Menus and Pointing device)* qui présente les bases fonctionnelles des interfaces graphiques informatiques, un menu contextuel est l'ensemble des commandes prêtes à être utilisées suivant le contexte courant. Il s'ouvre lorsque l'utilisateur clique sur un objet de l'interface graphique, toutefois, l'activation de la commande ne se fait que lorsque l'utilisateur clique sur la zone associée, sans avoir à reconnaître le tracé.

Les menus déroulants ou les menus linéaires sont le premier type de menus apparus dans les systèmes informatiques. Ce type de menus regroupe les éléments en colonne avec un ordre prédéfini par le système. Un menu est une liste de choix présentés à l'utilisateur contenant toutes les commandes possibles à l'exécution à partir de l'application courante (figure 1.1.a).

Les menus circulaires (en anglais : "pie menu") sont une autre approche de menus contextuels. Ils présentent les items du menu à une distance constante autour du curseur. Au lancement, Don Hopkins [7] place le curseur dans le centre -une région inactive- et les commandes se trouvent au voisinage dans des régions actives (cf. figure 1.1.b). L'utilisateur pourra choisir sa commande en déplaçant le curseur jusqu'à la zone de la commande concernée.

Dans [6], on présente une étude expérimentale pour expliquer quel type de menu est meilleur dans quelle situation et présenter les avantages et inconvénients de chacun. Cette étude est effec-

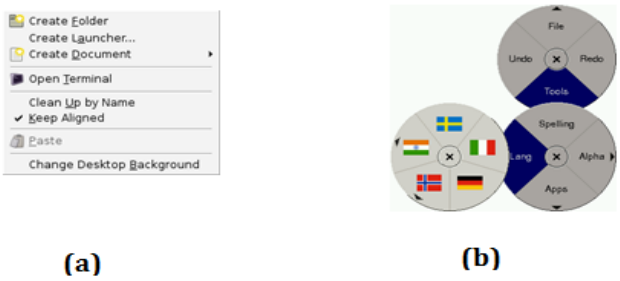


FIGURE 1.1 – Différents types de menus contextuels : a) menu linéaire dans Gnome, b) menus circulaires hiérarchiques

tuée sur les menus circulaires et les menus déroulants. L'expérience est faite sur des étudiants. On leur a proposé un ensemble de menus différents et à chaque fois ils devaient choisir le menu requête. L'évaluation consiste à mesurer les temps de recherche du menu et le taux d'erreur. Les résultats obtenus prouvent que les menus circulaires permettent de chercher un élément dans le menu et y accéder plus rapidement en parcourant une distance plus réduite que lorsqu'on utilise les menus déroulants. Cette étude montre aussi que les deux configurations souffrent du même taux d'erreur de sélection d'items notamment pour ceux qui se placent dans la région centrale du menu parce qu'ils ont le plus d'interaction avec les items voisins.

Menus pour périphériques à petites taille

ArchMenu et ThumbMenu sont des nouveaux types de menus proposés récemment par Huot[8] pour les terminaux à écran tactile de petite taille (PDA, smartphone...). Ces menus répondent aux problèmes d'utilisation liés à la mobilité lors de la manipulation (en marchant, dans les transports en commun...). Ces menus permettent l'interaction avec le pouce de la main qui tient l'appareil (figure 1.2), ainsi, l'utilisateur peut manipuler l'ensemble des menus d'une seule main. Aussi par contrainte d'espace d'affichage, ces menus ne couvrent que l'espace où le pouce pourra atteindre, par exemple le coin inférieur droit pour les utilisateurs droitiers. La forme des menus est inspirée des menus circulaires avec un demi-cercle. Quant au mécanisme d'expansion, ces menus proposent le déploiement en éventail dans le cas des menus hiérarchiques.

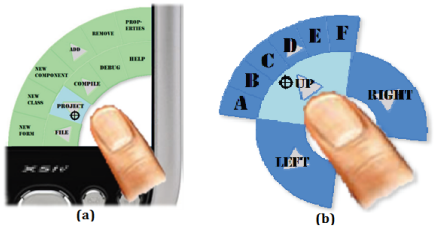


FIGURE 1.2 – a)ArchMenu, b)ThumbMenu

1.2 Interaction gestuelle

L'interaction gestuelle constitue un moyen non conventionnel d'employer les dispositifs de pointage en exploitant notre capacité de mémorisation et de reproduction des trajectoires ges-

tuelles, capacité que nous employons notamment pour l'écriture. consiste à permettre le déclenchement et l'exécution de commandes en réponse à un geste graphique saisi par l'utilisateur. Ceci est réalisable sur les interfaces qui combinent à la fois le dispositif d'affichage et celui de pointage qui permet de spécifier les positions du curseur. La taille de ce type de périphériques conduit généralement à des problèmes d'affichage lorsqu'on utilise les menus contextuels classiques. Pour éviter cela, les interfaces réagissant à des commandes gestuelles offrent aux utilisateurs un mode d'interaction particulièrement efficace et intuitif : les gestes sont réalisés directement sur la zone ciblée par la commande, à l'aide du doigt ou d'un stylet électronique. Ensuite, un classifieur prend en entrée le signal produit par le geste, et lui attribue la commande associée. Tout cela, se fait sans aucune interaction graphique intermédiaire et permet donc d'éviter la consommation de l'espace d'affichage du périphérique.
Il existe plusieurs façons de définir le jeu de gestes utilisé dans les applications qui reposent sur une interaction gestuelle.

Ensemble de gestes prédéfini

La majorité des concepteurs définissent un jeu de gestes prédéfini et rigide, chaque geste est dédié à l'exécution par le système d'une action bien précise. La forme des gestes se veut autant que possible liée par une relation *logique* (qui peut être une relation sémantique, une similarité graphique, ou un lien mnémotechnique) à la commande à effectuer. Par exemple, la commande *Copier* peut être associée au tracé de la lettre "C", la commande *Effacer* à un geste de rature... L'idée sous-jacente est que plus l'association commande-geste respecte une certaine *logique* accessible à l'utilisateur, plus il lui est facile d'apprendre les gestes associés aux commandes et de les mémoriser [14] et [16]. Plusieurs problèmes se posent avec ce type de systèmes :

- d'abord, l'apprentissage des gestes qui demeure une tâche fastidieuse pour l'utilisateur dès que le nombre de gestes à apprendre est conséquent ;
- ensuite, la *logique* du concepteur des gestes n'est pas universelle et comporte une part de subjectivité, et peut être considérée comme peu intuitive par certains utilisateurs ;
- enfin, il peut s'avérer impossible de concevoir des gestes intuitifs pour réaliser certaines commandes purement abstraites.

Choix de gestes par l'utilisateur

Pour remédier aux deux derniers problèmes de l'approche précédente, on peut donner à l'utilisateur le choix de définir la forme des gestes qu'il souhaite associer aux commandes. Cette stratégie nécessite d'avoir des moteurs de reconnaissance capables de reconnaître de nouvelles formes, tout en demandant le moins d'exemples d'apprentissage possible, ce qui reste un problème encore ouvert [1].

À partir d'un menu graphique

Cette dernière stratégie, celle qui nous intéresse, a été conçue pour répondre aux problèmes d'apprentissage des gestes. Elle consiste à construire le jeu de gestes à partir d'un menu graphique interactif. L'idée est de décorréler la sémantique de la commande de la forme du geste et de la lier à un menu interactif.

1.3 Menus gestuels interactifs

Les menus gestuels interactifs ont pour objectif principal d'assister l'utilisateur dans l'apprentissage du vocabulaire de gestes en l'aidant à tracer la forme du geste par la représentation de l'ensemble des directions offertes à lui en fonction de son emplacement actuel. Ce sont des menus contextuels que l'on déclenche lorsque l'on clique sur un objet graphique. Les items peuvent être statiques ou s'adapter en fonction de la zone de travail.

1.3.1 Modes d'interaction

Afin de favoriser l'apprentissage du jeu de gestes, les menus gestuels interactifs s'adaptent à l'utilisation en offrant deux modes de fonctionnent différents : un mode *novice* et un mode *expert*.

Mode novice

En mode novice, les menu gestuels interactifs fonctionnent généralement de la même façon que les menus interactifs ordinaires (menus linéaires et menus circulaires). Ce sont des menus contextuels qui s'affichent au moment où l'utilisateur clique sur le dispositif de pointage. L'utilisateur interagit avec le menu pour atteindre sa commande à l'aide d'un stylo. Il peut donc explorer l'ensemble des niveaux et les items qui s'y trouvent pour activer sa commande. En considérant qu'il ne connaît pas les l'emplacement de l'item souhaité, on lui accorde la possibilité de faire un retour arrière à tout moment en lui proposant un mécanisme approprié.

Mode expert

En répétant l'utilisation du menu, l'utilisateur apprend implicitement les gestes permettant d'activer ses commandes, jusqu'à être capable de les mémoriser. En comportement expert, l'utilisateur trace directement les mêmes commandes qui ont été apprises dans la phase d'apprentissage, sans que le menu ne soit affiché comme sur une interface à commande gestuelles classique. L'utilisation du menu dans le mode expert épargne à l'utilisateur les interactions visuelles et le temps de leur affichage.

Ce type de menu assure la transition naturelle du mode novice au mode expert suite à son utilisation fréquente. Nous présentons dans la suite quelques approches de menus gestuels interactifs présentées dans la littérature.

1.3.2 *Marking menus*

Les *Marking menus* de Kurtenbach [9] sont les premiers menus gestuels interactifs proposés pour aider l'utilisateur à passer du mode novice en mode expert. Ils sont inspirés des menus circulaires, ils présentent les items dans des secteurs de taille égale. L'utilisateur effectue son geste en atteignant la cible souhaitée. Ensuite, si le menu contient des sous-menus, une expansion est faite en affichant une autre zone sectorisée contenant l'ensemble des items possibles dans ce sous-menu (figure 1.3). Sinon, le geste est envoyé à un classifieur pour sa reconnaissance.

Plusieurs approches sont dérivées des *Marking Menu*, elles reposent sur le principe d'affichage des directions dans le mode novice pour assister l'utilisateur dans son tracé.

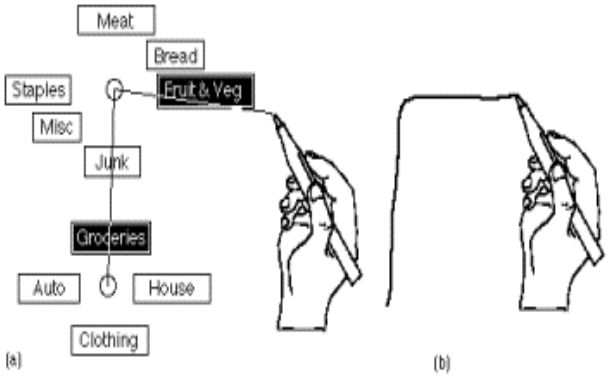


FIGURE 1.3 – Les *Marking menus* : a) mode novice, b) mode expert

Menus fleurs

Le menu fleur est un nouveau type de *Marking Menu*[3]. L'intérêt de ce type de menu par rapport au menu circulaire est le nombre de commandes pouvant être affichées dans un niveau. Il a été montré que pour avoir une bonne efficacité dans les menus circulaires on ne peut pas faire plus de 8 éléments par niveau. Les menus fleurs peuvent contenir jusqu'à 56 éléments à la fois à cause de leur organisation en groupes. Dans la figure 1.4.a, *New* et *Open* sont placés dans le même menu qui est *Fichier*. Un groupe peut contenir jusqu'à 7 éléments dans les 8 directions usuelles. La figure 1.4.c représente les sept degrés d'orientations possibles qu'on peut avoir dans chaque groupe. En pratique, un menu fleur ne contient que 20 éléments.

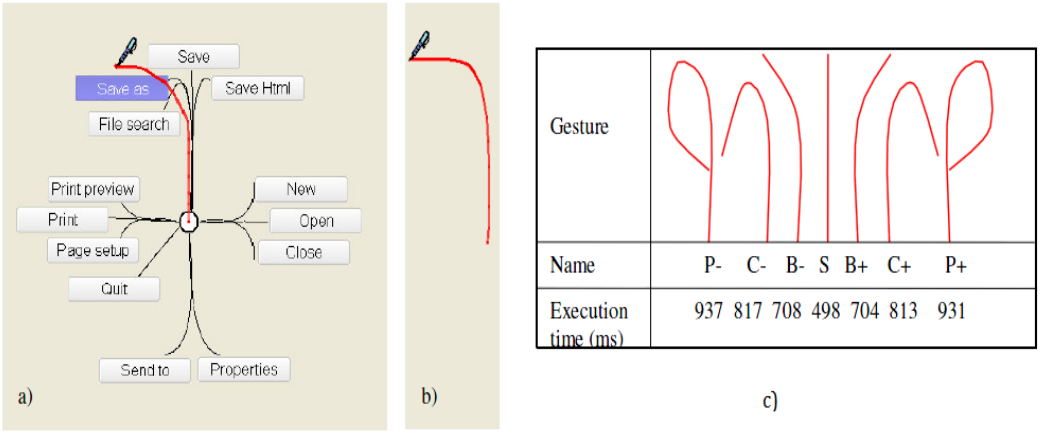


FIGURE 1.4 – Menu fleur : a) mode novice, b) mode expert, c) les 7 degrés d'orientations (ici dans la direction Nord)

Menus feuilles

Les menus feuilles ou "Leaf menu" sont une extension des menus linéaires par des raccourcis gestuels. Ce type de menus a été proposé par Bailly [4] pour remplacer les raccourcis clavier habituels dans les ordinateurs personnels. Un menu feuille fonctionne de la même manière qu'un menu linéaire sauf qu'il s'affiche par défaut en bas à gauche pour éviter les problèmes d'occul-

tation du doigt ou de la main. La figure 1.5 nous montre le fonctionnement du menu feuille. En mode novice (figure 1.5.a), l'utilisateur presse une cible, puis un menu linéaire apparaît après 0.3 seconde d'attente (si il bouge avant ce délais, le geste sera interprété comme étant en mode expert). La figure 1.5.b montre tous les chemins correspondants aux commandes du menu. Ensuite (figure 1.5.c), en choisissant une commande, (figure 1.5.d), un retour visuel confirme l'élément choisi et montre le geste associé. En mode expert, l'utilisateur exécute le geste directement sans attendre l'affichage du menu (figure 1.5.e).

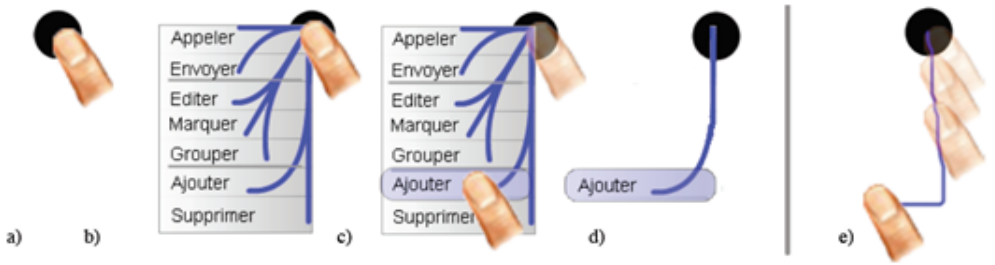


FIGURE 1.5 – Fonctionnement des menus feuilles : a-d) mode novice, e) mode expert

1.3.3 Guidage du tracé

L'approche du guidage du tracé est une nouvelle approche d'apprentissage des commandes gestuelles qui suit une autre stratégie que les *Marking menus* ; elle permet d'apporter une assistance à l'utilisateur au moment où il trace ses gestes en lui proposant des réponses visuelles sous forme de suggestions de trajectoires à partir de la position actuelle (cf. figure 1.6).
OctoPocus [5] est constitué de deux mécanismes travaillant en parallèle, le premier s'occupe de l'affichage en continu des commandes avec des épaisseurs différentes suivant la correspondance du geste amorcé avec celui suggéré. Le second mécanisme s'occupe de la reconnaissance gestuelle, il va accorder des taux d'appartenance à chaque classe à travers des algorithmes de classification, puis en fixant un seuil, il constitue l'ensemble de classes de commandes qui ressemblent le plus au geste tracé. Ces modèles les plus ressemblants sont ceux transmis au module de retour visuel qui les affiche à l'utilisateur.

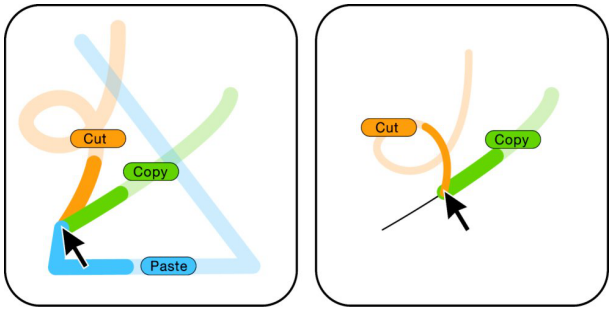


FIGURE 1.6 – Commandes suggérées par OctoPocus [5].

1.4 Limitations

Les menus cités dans cette section, on été conçu dans le but d'apprendre à l'utilisateur un raccourci gestuel pour exécuter les commandes sans faire appel au menu graphique. Cependant la forme des gestes induits par ce type de menus n'a pas été prise en compte et on peut distinguer quelques défauts liés à la forme globale du geste ou la façon de le dessiner. Les *Marking menus* classiques ne présentent à l'utilisateur (en mode novice) que les cibles contenant les libellés à atteindre, sans autre guidage, ce qui lui laisse un large champ de liberté dans le tracé des gestes activant les commandes. L'ensemble de gestes induit comporte une variabilité importante au sein de la même classe. Ceci peut rendre la tâche de classification difficile en créant des erreurs d'ambiguïté entre différentes classes.

Contrairement aux *Marking menus*, les menus fleurs et les menus feuilles de Bailly [3] et [4] imposent à l'utilisateur des trajectoires curvilignes prédéfinies et strictes (figure 1.4.b) afin d'avoir des gestes facilement identifiables, ce qui permet de garantir la forme des commandes gestuelles apprises. Ces trajectoires sont conçues pour favoriser la reconnaissance des gestes induits mais elle sont peu naturelles (elles imposent des détours, trajectoires non directes) pour l'utilisateur en mode novice et donc en mode expert.

Cette famille de menu (*Mareking Menu*) n'est pas développé avec l'objectif principal d'aboutir à une interface purement gestuelle. Elle ne présente aucun mécanisme qui prend en compte la nature particulière des gestes manuscrits : variabilité inertie, aspect naturel ou non, etc.

1.5 Solution proposée

Afin d'éviter les problèmes liés à la conception des formes de gestes, nous proposons de lier la forme des gestes à apprendre, non pas à une relation logique relative à la nature des commandes à réaliser, mais à la forme d'un menu interactif conçu spécifiquement, d'une part pour faciliter l'apprentissage des gestes par l'utilisateur et d'autre part pour optimiser la reconnaissance de ces gestes par le système (gestes discriminants).
sc

Notre approche est inspirée des *Marking menus*. Cependant, notre démarche poursuit un objectif différent : en effet, notre objectif est d'aboutir en mode expert à une interface purement gestuelle, dont les commandes seraient des gestes naturels, qui exploitent autant que possible la grande variété des formes propres à l'écriture manuscrite (lignes, courbes, boucles, points de rebroussement...). Nous nous attacherons donc à développer des menus dédiés, dont la forme et le comportement interactif bien spécifiques permettent d'aboutir à un jeu de gestes adapté à une utilisation naturelle d'interface gestuelle.

Le jeu de gestes proposé suivra une topologie qui évitera des gestes rectilignes sans imposer à l'utilisateur des contraintes fortes de trajectoire, ceci dans le but de faciliter la classification des gestes dans le mode expert en minimisant la variabilité intra-classe de chaque geste.

Chapitre 2

Conception de menu gestuel interactif

Dans ce chapitre nous présentons la conception du menu que nous avons proposé dans ce stage. D'abord, nous commençons par citer un ensemble de propriétés et de contraintes à respecter lors de la conception des menus interactifs. Ensuite, nous détaillons la topologie des menus et la composition des items. Nous présentons aussi les différents modes d'interaction : la première qui permet d'aider l'utilisateur à rechercher l'item souhaité et la seconde qui assure la fluidité des gestes lors d'un passage d'un niveau à un autre. Nous présentons à la fin l'impact des nos choix d'interaction et de topologie sur la forme des gestes.

2.1 Étude préliminaire

Dans un premier temps, nous avons commencé par une étude de la topologie des menus qui permettent à la fois la génération d'un alphabet de gestes facilement identifiables dans la phase expert et le respect d'un ensemble de critères d'ergonomie cités dans [2].

Recherche visuelle

Il existe différentes stratégies pour faciliter la recherche de l'item souhaité par l'utilisateur. La première consiste à **filtrer** les items en n'affichant que ceux fréquemment utilisés par l'utilisateur afin de réduire leur nombre. La deuxième stratégie consiste à **réorganiser** les items en groupes sémantiques ou en suivant une relation hiérarchique, ceci permet à l'utilisateur de cibler d'abord le groupe voulu, puis la commande souhaitée dans le groupe. La dernière stratégie repose sur un mécanisme de **mise en valeur** qui consiste à mettre en premier plan certaines commandes par rapport aux autres en fonction de leur taux d'activation, de leur pertinence... Ceci facilite le parcours visuel de l'utilisateur et par conséquent réduit le temps de recherche.

Temps de pointage

C'est le temps nécessaire pour atteindre le menu souhaité à partir de l'emplacement courant. Ce temps est fonction de la distance de l'item et de la largeur des menus graphiques qui servent à guider l'utilisateur. Pour les menus linéaires, ce temps est fonction de l'emplacement de l'item dans le menu, les derniers items sont placés plus loin du curseur et donc demandent plus de

temps que les premiers. Par ailleurs, les menus circulaires présentent les cibles à une distance constante de l'emplacement du curseur, ce qui permet d'atteindre n'importe quel item en traçant la même longueur du geste et donc ça réduit le temps de pointage.

Transition

La transition est la distance qui sépare la zone de travail courante et le menu, elle doit être petite pour ne pas détourner l'attention de l'utilisateur de son travail. Les menus contextuels respectent bien cette propriété en affichant le menu sur l'emplacement courant contrairement aux barres de menus qui sont dans la plupart des cas éloignées de la zone de travail.

Localité

C'est la distance qui sépare l'emplacement du curseur après avoir choisi la commande de la zone principale de travail. Cette distance ne dois pas être importante pour ne pas perturbé l'utilisateur après avoir choisi son menu. Pour les menus circulaires, cette distance est deux fois plus importante que pour les menus déroulants. Pour remédier à ce problème, une stratégie consiste à dessiner le geste en plusieurs tracés partant toujours depuis le même point et où chaque tracé correspond à un niveau dans la hiérarchie du menu. Ceci permet à ce que l'utilisateur reste toujours proche de la zone de travail quel que soit le nombre de niveaux parcourus.

Tâche de navigation

Dans le mode novice, l'utilisateur navigue dans arborescence des menus pour atteindre les items cibles tout en effectuant des retours arrières. Pour faciliter la navigation, la **prévisualisation** des items des sous-menus permet d'inspecter rapidement les différentes possibilités offertes à l'utilisateur. Afin d'assurer le retour arrière, les items parents doivent être toujours visibles.

Forme des gestes

La forme des gestes est imposée par la topologie du menu. Dans le cadre d'une interaction gestuelle suivant un menu interactif, la forme du menu est complètement décorrélée du sens de la commande. Il faut que les menus n'exigent pas de contraintes d'utilisation fortes sur les trajectoires à prendre afin de rendre la manipulation intuitive et naturelle par rapport à la topologie du menu.

2.2 Topologie du menu

Notre solution a été conçue pour une utilisation sur Tablet PC, avec un large espace d'affichage et un mode d'interaction stylo confortable (l'utilisateur peut poser la paume de sa main sur la surface de saisie). Ce mode de saisie offre une interaction plus naturelle et plus directe comparée à une interaction via la souris.

En reprenant le principe des *Marking menus*, le mode novice de notre menu permet un guidage du tracé de l'utilisateur par présentation des alternatives qui s'offrent à lui (branches du menu orientées dans différentes directions). L'enchaînement de ces directions, par exploration des niveaux hiérarchiques du menu, aboutit à une trajectoire complexe formant le geste de la commande associée. En mode expert, l'activation de la commande se fait en traçant sans guidage et sans assistance le même geste qui a été appris implicitement dans le mode novice.

La topologie de notre menu induit un ensemble de gestes différenciables tout en restant naturels pour l'utilisateur. Ici, l'utilisateur doit uniquement suivre les branches du menu pour l'activer : la trajectoire est donc guidée implicitement par les menus sans être imposée explicitement. Tout passage en dehors des branches n'induit aucune action ou retour visuel. Ceci va permettre à l'utilisateur d'acquérir d'une manière transparente des gestes qui suivent une topologie bien précise.

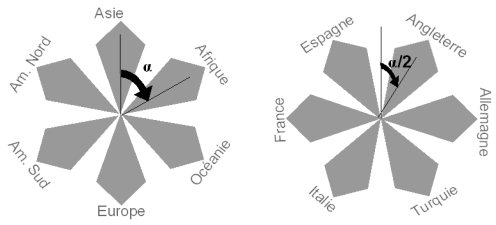
La fluidité de l'utilisation du menu en mode novice, nécessaire à la bonne mémorisation de la commande gestuelle, est assurée par un retour visuel continu et notamment par une gestion particulière des transitions entre les différents niveaux hiérarchiques du menu.

La forme des menus proposés permet d'induire un ensemble de gestes variés (zigzags, boucles, demi-cercles) qui sont facilement différenciables en mode expert. Ces gestes n'ont aucune relation sémantique avec les commandes associées mais sont en relation directe avec la forme des branches parcourues.

2.2.1 Disposition des branches

D'après l'étude de Hopkins [7], les menus circulaires hiérarchiques ne peuvent contenir plus de huit éléments dans le même niveau en raison de la complexité visuelle et de la facilité de recherche. Notre prototype contient au plus six items dans un seul niveau, disposés d'une façon symétrique avec une marge angulaire constante (cf. figure 2.1). En pratique, la profondeur du menu ne peut dépasser 4 niveaux, ce qui fait un nombre total de 1296 commandes possibles. Chaque niveau est constitué d'un ensemble de branches en forme de cerf-volant centrées sur le point de déclenchement du menu. Les labels se situent à l'extrémité de chaque branche dans un rectangle perpendiculaire à sa direction principale.
La marge angulaire α autorisée est égale à 2π divisé par le nombre maximal d'items autorisé (dans notre cas $\alpha = \pi/3$). Afin d'assurer les ruptures dans la trajectoire du geste entre les différents niveaux de hiérarchie, les menus se présentent sous deux configurations différentes obtenues en appliquant une rotation de $\alpha/2$. La figure 2.1.a présente la configuration dans un niveau i. En appliquant une rotation de $\pi/6$ sur le sous-menu, nous obtenons la deuxième configuration possible (figure 2.1.b) qui présente la disposition des branches du niveau $i+1$.

Les deux configurations sont proposées alternativement à chaque changement de niveau hiérarchique, pour imposer des déviations dans le tracé au minimum à $\alpha/2$ par rapport à la branche parente. Ce mécanisme nous permet d'obtenir des formes de gestes non rectilignes d'une façon naturelle en traçant des gestes directs sans poser de contraintes explicites sur la trajectoire.



(a) niveau i (b) niveau i+1

FIGURE 2.1 – Les deux configurations possibles du Menu : menu au niveau i (a) et son sous-menu au niveau i+1 (b)

2.2.2 Découpage en zones

Chaque branche comporte deux zones d'interactivité : *zone de présélection* et *zone d'inertie* (voir figure 2.2). Chaque zone assure un retour visuel continu pour indiquer à l'utilisateur qu'il est sur une trajectoire correcte.

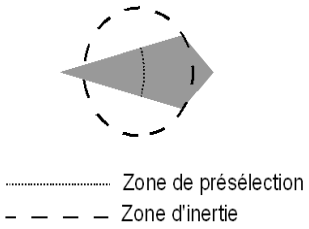


FIGURE 2.2 – Zones d'interaction

Zone de présélection

La première zone est la zone de présélection, elle permet à l'utilisateur de chercher l'item souhaité. Elle est associée à un retour visuel qui consiste à renforcer en permanence la couleur de la branche survolée par le stylo tout en affaiblissant les autres branches (figure 2.3). En franchissant la limite de cette zone, la branche survolée passe à l'état *pré-sélectionnée* alors que les autres ont complètement disparus.

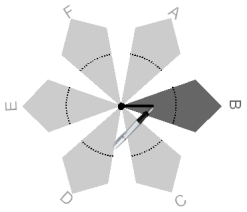


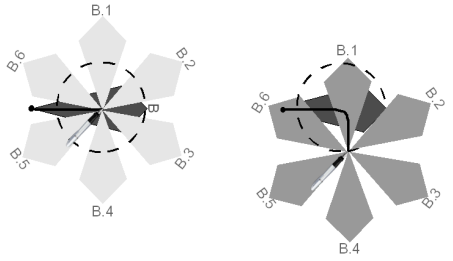
FIGURE 2.3 – Mise en relief de la branche sélectionnée dans la zone de présélection

Zone d'inertie

Une fois dans la zone d'inertie (centrée sur le point de sortie de la zone de présélection), l'utilisateur commence à voir apparaître graduellement les branches du sous-menu à mesure qu'il s'éloigne du centre de la zone (figure 2.4.a). Dans cette zone, le sous-menu suit le mouvement du stylo en restant centré sur la pointe du stylo de l'utilisateur, jusqu'au franchissement de la limite de la zone d'inertie (figure 2.4.b). Cette zone permet d'absorber l'inertie des changements de trajectoire pour améliorer la qualité du retour visuel et permettre à l'utilisateur de faire un geste le plus fluide possible lorsqu'il passe d'un menu à un sous-menu. La fluidité des gestes est importante pour la reconnaissance gestuelle utilisée pour le mode expert.

2.3 Interaction

Le menu offre une interactivité permanente avec l'utilisateur via différents types de retours visuels asservis au mouvement de son tracé pendant l'élaboration de sa commande.



(a) (b)

FIGURE 2.4 – Translation du sous-menu B1-B6 du point d'entrée dans la zone d'inertie (a) au point de sortie (b)

2.3.1 Description pas à pas

Dans la figure 2.5, nous présentons les différentes étapes d'interaction lors de la sélection d'un menu :

- (a) lorsque l'utilisateur clique sur l'écran du Tablet PC, il voit apparaître le menu et l'ensemble des directions possibles ; il se trouve alors par défaut dans la zone de présélection ;
- (b) lorsque l'utilisateur commence à naviguer dans une des directions dans cette zone, la branche choisie "Europe" se renforce en intensité de gris, tandis que les autres branches s'estompent au fur et à mesure qu'il s'éloigne du centre du menu ;
- (c) une fois sorti de la zone de présélection de "Europe", l'utilisateur se trouve dans la deuxième zone d'interaction à savoir la zone d'inertie. À ce moment, seule la branche choisie par l'utilisateur lui est visible ;
- (d) dans la zone d'inertie, les branches du sous-menu "Europe" apparaissent progressivement àmesure que la distance qui sépare la position actuelle du point d'entrée à cette zone. Ces branches du sous-menu suivent le mouvement du tracé et restent centrées sur la position du stylo ;
- (e) franchissement de la zone d'inertie, le sous-menu "Europe" se fixe. La branche du menu parent ("Europe") est validée, et change de forme pour le signifier à l'utilisateur. Un autre cycle de développement de menu débute (retour à l'étape de la figure (a)).

À tout moment, l'utilisateur peut revenir aux menus parents en parcourant les branches en sens inverse.

Description de la courbe d'évolution de la complexité d'affichage

La courbe représente l'évolution de l'intensité des items en fonction du tracé de l'utilisateur. Elle est alignée avec les étapes de sélection du menu afin d'indiquer les l'état des branches à chaque moment. Dans le niveau "affichés", on représente l'ensemble et le nombre des branches présentées à l'utilisateur, par contre, le niveau "caché" représente les items cachés durant la manipulation. Dans la zone de présélection du premier niveau, on remarque qu'au départ on affiche les six branches à l'utilisateur. Ensuite, à force de s'éloigner du centre du menu, cinq branches diminuent en intensité et deviennent de plus en plus transparentes tandis qu'une branche commence à se mettre en relief pour indiquer à l'utilisateur son choix de présélection. Ensuite, dans la zone d'inertie du même niveau, en gardant la même branche présélectionnée, on commence à voir apparaître les six branches du sous-menu sélectionné qui se renforcent progressivement en fonction de la distance qui sépare l'emplacement actuel du centre de cette zone.

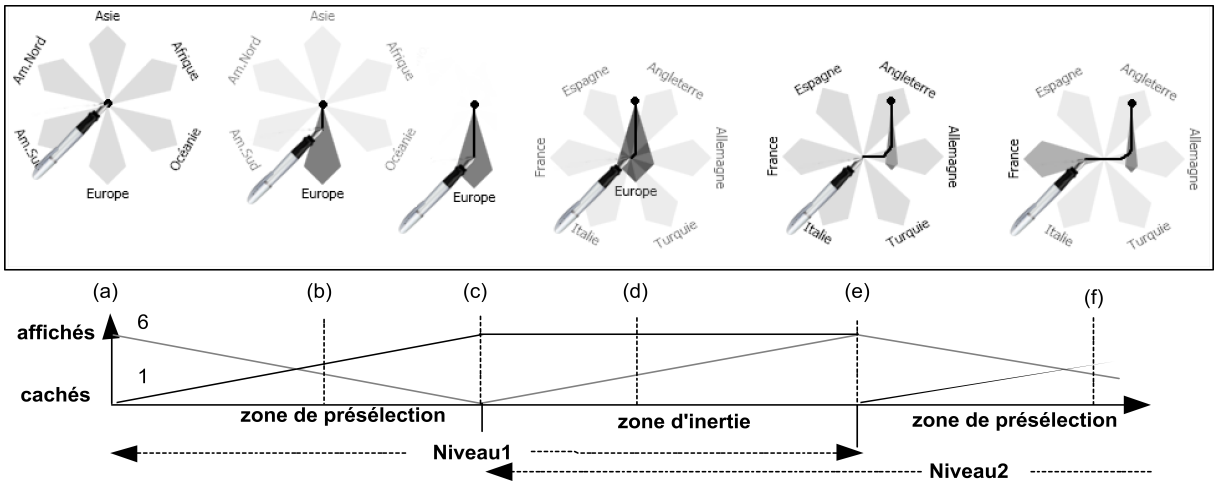


FIGURE 2.5 – Les différentes étapes d'interactions et réponses visuelles proposées dans le mode Novice. La courbe de l'évolution de la complexité d'affichage représente le changement d'intensité des branches affichées en fonction de la progression du tracé dans les zones d'interaction.

2.3.2 Propriétés

L'interaction proposée dans le mode novice permet de répondre aux différentes contraintes d'ergonomie [2]. L'utilisateur peut naviguer entre les différents niveaux tout en autorisant les retours en arrière en parcourant les branches parentes.

Localité

La topologie du menu et le mécanisme d'expansion des niveaux permettent d'afficher les branches du sous-menu dans différentes directions. Cela nous permet de proposer des trajectoires capables de se replier sur elles-mêmes et donc rester proche de la même zone de travail. Ceci nous offre la possibilité de lancer la commande en traçant un geste à tracé unique *"Single-stroke"*.

Temps de recherche

En fonction de l'amorce du tracé, les branches qui ne sont pas concernées par la direction du geste s'estompent en mettant en relief la branche concernée par le tracé. Ceci facilite la lecture des items et réduit la complexité visuelle en cachant le plus tôt possible les autres branches.

Complexité visuelle

Le mécanisme d'interaction permet de réduire au plus vite le nombre d'items affichés en estompillant les branches qui ne suivent pas la direction du tracé. Le nombre de branches affichées est toujours égal au nombre de niveaux parcourus ajouté au nombre des items actuels. On ne permet jamais d'afficher les items de deux niveaux en même temps. Ceci permet à ce menu d'être utilisé même sur les périphériques mobiles à petite taille d'affichage.

2.4 Impact sur la forme des gestes

Le menu proposé dans cette section intervient spécialement dans la forme des gestes induits par la topologie. Les deux configurations et les zones d'interaction permettent d'obtenir un ensemble de gestes variés sans poser de contraintes fortes à l'utilisateur dans la trajectoire

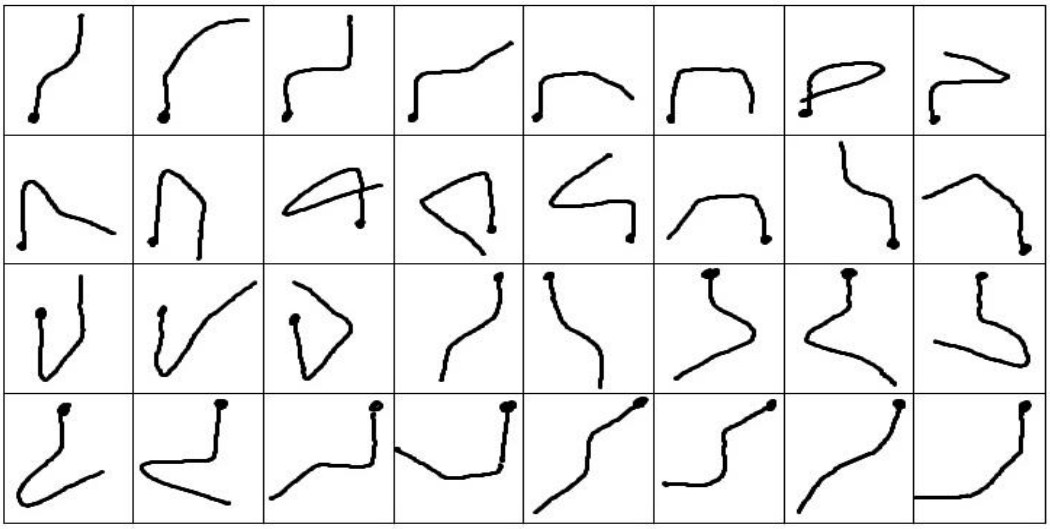


FIGURE 2.6 – Exemples d'un ensemble de gestes induits par la topologie du menu

à prendre. L'alternance des différentes configurations implique des déviations d'un angle d'au moins un $\alpha/2$, ce qui nous permet de distinguer chaque passage d'un niveau à un autre. Cela nous aidera lors de la segmentation du geste en niveaux pour la phase de reconnaissance partielle. Aussi, la zone d'inertie permet d'absorber les différentes variations dues aux transitions d'un niveau à un autre ce qui rend les gestes fluides et directs.

Dans la figure 2.6, nous présentons un ensemble de 32 gestes tracé suivant la topologie du menu présentée dans cette section. Ils illustrent les formes induites par la topologie de menu proposée dans cette section. Ils représentent des gestes associés à des commandes de trois niveaux de la hiérarchie. La variété des formes que présente cet ensemble justifie les choix de disposition des items et les angles choisis, on remarque que cet ensemble contient des formes rectilignes, boucles, zigzags. On peut remarquer aussi la symétrie entre classes qui est due à la disposition des items. Quant à la fluidité, on peut constater que sur tous les gestes les ruptures entre chaque niveau hiérarchique sont bien marquées. par Cet ensemble nous permet de confirmer nos choix de topologie pour la conception des menus, ceci va nous aider dans la phase de reconnaissance de geste en discriminant des formes bien variées.

Chapitre 3

Classification des commandes

Le mode expert de notre système de menu repose sur un moteur de reconnaissance de gestes capable d'identifier n'importe quel geste et de lui affecter ou non la commande associée. Notre menu peut être configurable par l'utilisateur en lui offrant la possibilité d'ajouter, de supprimer ou de modifier des items. Pour cela, nous proposons un mécanisme d'apprentissage capable de reconnaître des nouvelles formes avec peu d'exemples d'apprentissage. Afin de réduire le temps de reconnaissance, nous autorisons aussi au classifieur la possibilité de rejeter les classes au fur et à mesure que le signal est tracé, ceci dans le but de réduire l'espace d'hypothèses concurrentes.

Dans ce chapitre, nous présentons un état de l'art sur la classification gestuelle en citant quelques travaux réalisés dans le domaine. Ensuite nous décrivons l'algorithme utilisé pour la reconnaissance gestuelle basé sur la programmation dynamique. Nous présentons après notre technique de rejet et l'intérêt que cela apporte dans notre classification en élaguant les classes concurrentes. Nous présentons enfin la reconnaissance à la volée qui consiste à reconnaître un geste en utilisant peu d'information sur le signal dans le but de prendre une décision sur la nature de la commande le plus tôt possible.

3.1 Reconnaissance gestuelle

La reconnaissance gestuelle fait appel aux techniques utilisées dans la reconnaissance de formes, c'est un ensemble de techniques visant à identifier des motifs à partir de données brutes pour prendre à la fin une décision sur leur nature. La reconnaissance en-ligne est utilisée lorsque l'échantillon d'encre est constitué d'un ensemble de coordonnées ordonnées dans le temps, il est donc possible de suivre le tracé et de connaître les posés et levés de stylos, la vitesse et la pression exercée sur le dispositif de pointage. La reconnaissance gestuelle tout comme la reconnaissance de forme des gestes s'appuie sur différentes approches : une approche statistique et une autre structurelle.

Une approche statistique consiste à représenter la forme (dans notre cas le geste) par un ensemble de caractéristiques calculées sur des différents niveaux de considération (par point, par tracé ou geste entier). Ensuite, on se basant sur un ensemble d'exemples, on étudie la représentation de chaque classe dans l'espace des caractéristiques pour la construction des modèles ou de frontières de décision. Dans ce type d'approche, on représente l'objet à reconnaître par un vecteur de caractéristiques souvent de grande dimension. Puis il est envoyé en entrée à un

classifieur entraîné tel que les réseaux de neurones ou les SVM (séparateurs à vaste marge).

La reconnaissance gestuelle a fait ses premières apparitions sur des gestes simples avec un seul tracé, c'est-à-dire qu'on ne lève pas le stylo afin de former un tracé continu. Rubine [14] propose une approche statistique en se basant sur un ensemble de caractéristiques extraites à partir du des gestes illustrés dans la figure 3.1). La classification se fait avec un calcul linéaire de la somme des caractéristiques pondérée avec des poids calibrés pendant la phase d'apprentissage. L'apprentissage consiste à modifier les poids des caractéristiques suivant leur contribution à la discrimination. La classe choisie est celle qui maximise la somme :

$$v_c = w_{C0} + \sum_{i=0}^{f} w_{ci} f_i$$

où f_i est une composante du vecteur de caractéristiques et w_{ci} est le poids associé à cette caractéristique pour la classe C_i.

Pour les gestes iconiques présentés dans la figure 3.2 (composés de plusieurs tracés), Willems[16] a proposé plusieurs ensembles de caractéristiques sur le geste tracé calculés sur différents niveaux de considération du tracé (par point, par tracé ou le geste entier). Il a utilisé deux types de classifieurs : les réseaux de neurone et SVM et a obtenu un taux de classification égale à 99% en utilisant le SVM.

L'approche structurelle consiste à décomposer la forme en objets primitifs et à définir une description de l'organisation de ces objets les uns par rapport aux autres. Une première approche structurelle consiste à définir des modèles structurels de chaque geste dans la phase d'apprentissage et à faire une correspondance entre le modèle en entrée et ceux de la base de données.

FIGURE 3.1 – Exemples de gestes à simple tracé [14]

3.2 Description de l'algorithme de classification (DTW)

L'utilisation du système en mode expert fait appel à un moteur de reconnaissance de gestes qui ne manifeste aucune interaction graphique lors du tracé du geste. Une fois le geste tracé, on lui applique des opérations de prétraitement (normalisation, ré-échantillonnage), puis il est transmis au classifieur. Le classifieur détermine alors la classe correspondant au geste effectué et cette commande est exécutée.

Pour valider notre approche et justifier la topologie des menus, nous avons évalué le taux de bonne classification des gestes tracés. Pour cela, nous avons développé un classifieur basé sur un algorithme de *DTW (Dynamic Time Warping)* [12],[13].

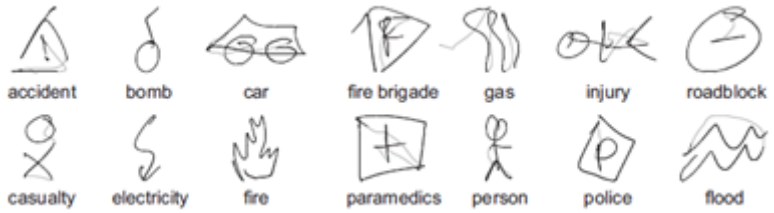


FIGURE 3.2 – Exemples de gestes multi tracés de la base de données NicIcon tracés par un seul scripteur [16].

Le DTW est une approche de classification basée sur un calcul d'une mesure de similarité entre des prototypes qui modélisent les classes et l'objet à identifier. Cette technique est capable de comparer deux courbes en cherchant une correspondance optimale entre elles en exploitant l'information spatiale et temporelle. Ainsi, un chemin de correspondance est créé regroupant les différentes combinaisons des points mis en correspondance des deux courbes. La distance entre les deux courbes est la somme des distances de l'ensemble des points.

3.2.1 Calcul du chemin de correspondance

DTW est une méthode qui recherche un appariement optimal entre deux séries temporelles sous certaines restrictions. Les séries temporelles sont déformées par transformation non-linéaire de la variable temporelle, pour déterminer une mesure de leur dissimilarité, indépendamment de certaines transformations non-linéaires du temps.

Étant donnée deux séries temporelles $P = (p_1, p_2 \ldots, p_N)$ et $Q = (q_1, q_2 \ldots, q_M)$, l'algorithme DTW construit le chemin de correspondance en faisant la correspondance point à point ou point à plusieurs points i et j si les conditions suivantes sont vérifiées :

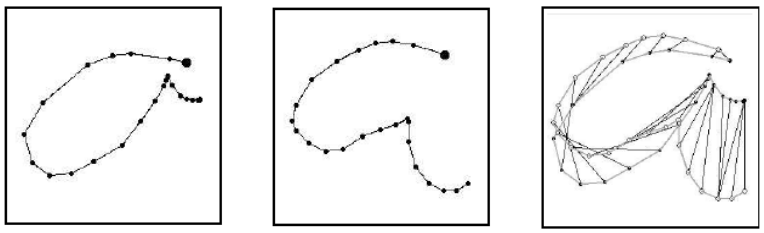


FIGURE 3.3 – Illustration du résultat d'un calcul d'appariement entre deux séries temporelles correspondant à deux caractères : (a) et (b) sont les courbes en entrée. (c) l'appariement du DTW chaque point de la première courbe est associé avec celui qui lui correspond dans la deuxième courbe en fonction des conditions du DTW[12]

Condition de continuité

C'est la condition principale du DTW : elle permet de décider le facteur d'appariement accordé pour une correspondance non linéaire. Pour cela si N_1 et N_2 sont les nombres de points

de deux courbes, alors le i^{eme}point de la première courbe et le j^{eme} point de la seconde peuvent être associés si :

$$\frac{N_2}{N_1}i - cN_2 \leq j \leq \frac{N_2}{N_1}i + cN_2$$

où c représente le degrés d'appariement autorisé pour assurer la continuité, il prend une valeur dans l'interval 0..1 Dans la figure 3.4, les croix représentent les couples de points qui ne peuvent pas être mis en correspondance. En même temps, les cases vides représente la largeur autorisée pour apparier les points des deux courbes.

9	×	×	×	×	×	×				•
8	×	×	×	×	×					
7	×	×	×	×						
6	×	×	×							
5	×	×								×
4	×								×	×
3								×	×	×
2							×	×	×	×
1						×	×	×	×	×
0	•				×	×	×	×	×	×
	0	1	2	3	4	5	6	7	8	9

FIGURE 3.4 – Construction de matrice d'appariement pour deux courbes avec : $N_1 = N_2 = 10$ et $c = 0.3$

Condition de limites

Cette condition force l'algorithme de DTW à faire correspondre les deux premiers et les deux derniers points des deux courbes, ceci se fait que la condition de continuité soit vérifiée ou non. Dans la figure 3.4, les deux points (0,0) et (9,9) représentent les limites imposées par l'algorithme.

Monotonicité

Cette condition oblige l'algorithme DTW à ne pas faire de retour arrière lors de l'appariement, cela signifie que si un point i de la première courbe est associé avec un point j de la seconde courbe, alors il n'est plus possible de faire des correspondances entre des points de première courbe $> i$ avec des points $< j$ de la seconde courbe et inversement ($< i$ et $> j$).

3.2.2 Algorithme

La programmation dynamique est une approche qui permet d'obtenir la solution optimale à un problème de minimisation d'un certain critère d'erreur. Afin de calculer le chemin optimal, notre algorithme proposé (Algorithme 1) DTW est basé sur un calcul de l'appariement qui commence toujours du point (0,0) des deux séries jusqu'au point (N,M) et avancer toujours en incrémentant soit l'indice i ou j soit les deux en même temps. Les conditions présentées précédemment sont respectées, notamment la condition de limite puisque on commence toujours depuis (0,0) jusqu'à (N,M). Puisque on ne peut qu'incrémenter les indices pour l'appariement, on n'autorise aucun retour arrière pour les deux indices, ce qui vérifie la condition de monotonicité.

Algorithm 1 DTW

Require: $P = (p_1, p_2 \ldots, p_N)$ et $Q = (q_1, q_2 \ldots, q_M)$

```
TotalDist=Dist(0,0)
i=0, j=0
while i < N and j < M do
    ProchainPoint= Min( Dist(i,j+1),
                        Dist(i+1,j),
                        Dist(i+1,j+1)))
    TotalDist+=Dist(ProchainPoint)
end while
return TotalDist
```

Algorithm 2 Décision

Require: Prototypes$\{P_1, P_2 \ldots, P_L\}$ et Requête :Q

```
DistMin=+∞
for tous protypes P_I do
    DTWDist=DTW (P_I, Q)
    if DTWDist<DistMin then
        DistMin=DTWDist
        Classe=P_I
    end if
end for
return Classe
```

3.2.3 Sélection des prototypes

L'un de principaux avantages du DTW est qu'il ne nécessite pas de phase d'apprentissage. Les prototypes sont sélectionnés parmi les gestes de la base en faisant un calcul du plus proche voisin. en choisi aléatoirement un sous-ensemble de gestes par classe et on prend le geste qui minimise la distance avec tous les autres. Ce mécanisme de sélection ne nécessite pas une phase d'apprentissage ce qui permet de rendre les menus plus dynamiques, c'est-à-dire que l'utilisateur peut ajouter, supprimer ou modifier l'emplacement des commandes comme il le souhaite.

3.2.4 Classification

Dans l'algorithme 2, on présente la classification qui se fait en comparant un objet en entrée avec tous les prototypes de la base de donnée. L'algorithme choisit à la fin le prototype qui minimise la distance cumulée du DTW.

3.3 Techniques de rejet

Le rejet est la capacité de ne pas classifier un échantillon afin d'éviter de faire des erreurs. Il existe deux causes principales qui peuvent causer ces erreurs : la première est quand le classifieur hésite entre deux classes possibles, et la seconde est quand l'échantillon à reconnaître est trop différent des échantillons qui ont servi à son apprentissage. Deux types de rejets sont définis, un pour chaque type d'erreur : le rejet d'ambiguïté et le rejet de distance ou d'ignorance.

Le rejet d'ambiguïté permet d'augmenter la fiabilité du classifieur en rejetant les échantillons que le classifieur risque d'associer à des mauvaises classes. C'est-à-dire que le classifieur les

identifie comme proche des frontières de décision. Une solution à ce problème est de définir un couloir autour des frontières de décisions et rejeter tous les échantillons qui s'y trouvent. Dans la figure 3.5.a [11] utilise un réseau de neurones à fonctions à base radiales (Radial Basis function network) pour discriminer trois classes en définissant des zones de rejets autour de la surface séparatrice.

Le rejet de distance est la délimitation des connaissances du classifieur ; c'est-à-dire que le classifieur n'est capable de classer que ce qu'il a appris, il doit rejeter tout échantillon qui n'appartient à aucune classe. Dans la figure 3.5.b, le classifieur n'est capable de reconnaître que trois classes, tout en rejetant une quatrième qu'il n'a pas rencontrée dans la phase d'apprentissage.

Le mécanisme de rejet nous apporte une amélioration dans la qualité des réponses visuelles rendues à l'utilisateur. Il va rejeter tout geste n'appartenant pas à l'ensemble des gestes prédéfinis par le système.

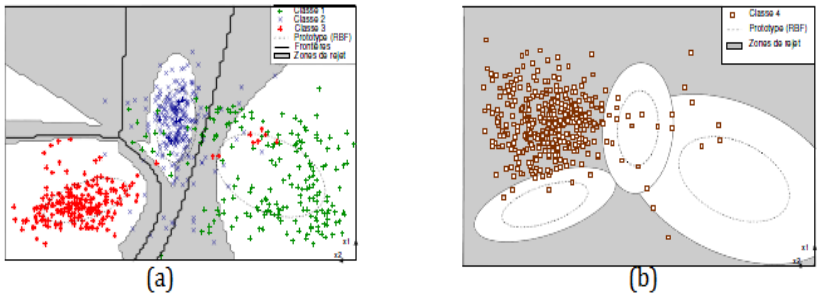


FIGURE 3.5 – Visualisation des zones de rejet : a) zone de rejet de confusion. b) zones de rejet de distance [11].

3.3.1 Calcul du seuil de rejet

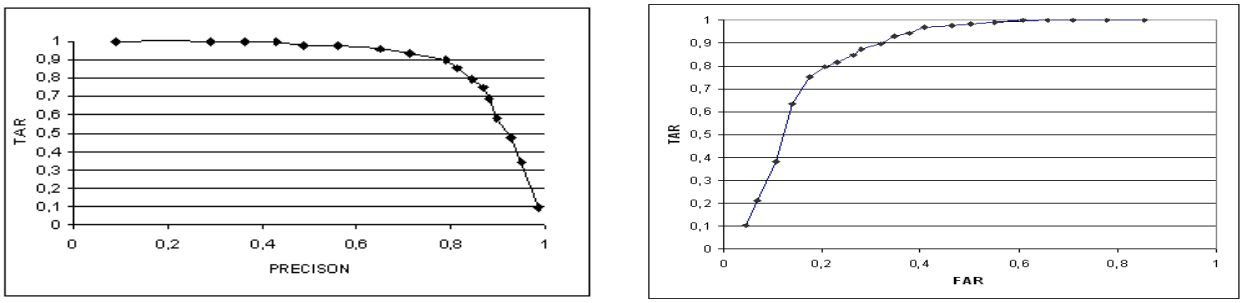
L'utilisation du rejet nécessite généralement un seuil qui permet d'accepter ou de refuser un échantillon. Pour déterminer ce seuil, les courbes ROC (*Receiver Operating Characteristic*) offrent une représentation visuelle sur les performances du seuil utilisé. Pour construire une courbe ROC, il faut disposer de deux bases de données : une base contenant les exemples positifs N_E et une base contenant les exemples négatifs N_R. Le but du rejet est de d'accepter tous les exemples positifs et de rejeter tous les exemples négatifs. Puisque cet objectif n'est généralement pas atteignable, il faut alors faire un compromis entre le taux d'acceptation et de rejet. Ceci est réalisable en calculant la proportion des exemples positifs acceptés par le classifieur (TAR : *True Acceptance Rate*) et ceux refusés par erreur (FRR : *False Reject Rate*) et la proportion des exemples négatifs rejetés (TRR : *True Reject Rate*) et ceux acceptés par erreur (FAR : *False Acceptation Rate*). La précision est la proportion de l'ensemble des exemples positifs bien classés sur l'ensemble total des bien classé.

		Acceptés		Rejetés	
	Nombre	Nombre	Mesure	Nombre	Mesure
Exemples	N_E	N_E^A	$TAR = N_E^A/N_E$	N_R^E	$FRR = N_R^E/N_E$
Contre-exemples	N_R	N_A^R	$FAR = N_A^R/N_R$	N_R^R	$TRR = N_R^R/N_R$

FIGURE 3.6 – Les différentes mesures pour évaluer le rejet de distance

Il existe différentes types de représentations des courbes ROC :

- TAR/Précision : cette représentation permet d'évaluer la capacité à ne reconnaître que les exemples positifs, la solution optimale correspond au point (1,1) (figure 3.7.b, c'est-à-dire un rappel de 100% avec une précision de 100% ;
- TAR/FAR : montre à quel point le classifieur est capable de rejeter les exemples négatifs tout en acceptant les positifs. La solution optimale se correspond au point (0,1)(figure 3.7.b).



(a) courbe ROC TAR/Précision (b) courbe ROC TAR/FAR

FIGURE 3.7 – Quelques représentations des courbes ROC pour l'évaluation des performances du choix d'un seuil de rejet.

Notre approche

Dans notre système de menu, nous allons alléger le travail du classifieur en réduisant le nombre de classes concurrentes. Notre technique de rejet consiste à éliminer les modèles qui ne suivent pas le tracé à chaque niveau. L'idée est qu'une fois les distances qui séparent les prototypes de la requête déterminée, on calcule un seuil relatif aux résultats obtenus, et on élimine toute classe dont la distance est supérieure à ce seuil. Le seuil est fixé par un facteur ρ utilisé pour éliminer les classes dont le score est supérieur à ρ fois le plus petit score obtenu dans une étape i.
L'élimination des classes signifie la réduction des hypothèses possibles pour le classifieur, ce qui entraîne une diminution importante dans le temps de calcul des distances. L'utilisateur ne sera donc pas perturbé pas le temps d'attente du classifieur, la réponse sera donnée instantanément à la fin de son tracé.

3.4 Reconnaissance à la volée

L'un des objectifs de notre stage est de pouvoir reconnaître le geste au moment où il est tracé. La reconnaissance à la volée permet de prendre des décisions intermédiaires sur la classe d'appartenance du geste sans qu'il soit fini.

Uchida [15] propose une approche de reconnaissance prédictive basée sur une combinaison de classifieurs. Le geste est découpé en fenêtres séquentielles, il y a autant de classifieurs que de fenêtres. Chaque classifieur s'occupe de la classification du vecteur de caractéristiques de sa fenêtre associée. Le résultat de la reconnaissance à un instant t pour une classe C est égale à la

somme de tous les résultats de fenêtres précédentes pondérée avec des poids fixés dans la phase d'apprentissage en utilisant une technique de propagation des poids.
Les auteurs ont validé leur approche dans le cadre de la reconnaissance de caractères en-ligne. La base de données est constituée de 500 exemples d'apprentissage des chiffres de 0 à 9 et de 300 autres de tests. Les performances sont évaluées pour la discrimination de chaque paire c'est-à-dire 0-1, 0-2, 0-3... Les performances obtenues montrent que la propagation des poids lors de la phase d'apprentissage améliore le taux de bonne classification (par exemple, un taux de 90% est atteint à l'instant t=23 avec la méthode de propagation des poids, par contre le même taux est atteint à l'instant t=37 mais sans propagation des poids).

Une autre approche de reconnaissance à la volée utilisée dans la reconnaissance gestuelle corporelle proposée par Uchida[10] se basant sur la programmation dynamique. Elle consiste à associer chaque frame du signal en entrée avec une autre frame du prototype de la base de données pour construire un chemin d'appariement en utilisant le calcul récurrent suivant suivant :

$$g_{c,1}(\tau) = 3d_{c,1}(\tau)$$

où $d_{c,t} = \|I_\tau - R_{c,t}\|$ est la distance entre une frame τ en entrée avec une frame t d'une classe de la base de données. Pour le reste des frames, on associe deux frames en utilisant la fonction suivante :

$$g_{c,t}(\tau) = min \begin{bmatrix} g_{c,t-1}(\tau-1) + 3d_{c,t}(\tau) \\ g_{c,t-1}(\tau-2) + 2d_{c,t}(\tau-1) + d_{c,t}(\tau) \\ g_{c,t-2}(\tau-1) + 3d_{c,t-1}(\tau) + 3d_{c,t}(\tau) \end{bmatrix}.$$

Intégrer ce mécanisme dans la reconnaissance gestuelle signifie une interaction continue entre le système et l'utilisateur. Il pourra donc voir apparaître des résultats de ce qu'il a fait et décidera simultanément du chemin à prendre.

Mise en œuvre

Nous avons mis en œuvre la stratégie de reconnaissance à la volée en apportant quelques modifications sur l'algorithme 1 DTW. En effet, nos travaux sont inspirés de ceux d'Uchida [10] qui a utilisé un algorithme de classification basé sur DTW.

L'idée est de découper les prototypes en un nombre de frames égale au nombre de niveaux du menu.

Puis appliquer à chaque frame du geste à identifier un appariement avec la frame correspondante

Le classifieur choisira à la fin la classe qui minimise la distance de l'appariement parmi tous les modèles.

$$C_t = Min(dist(P_{i,t}, Q_t) \qquad 0 < t < T, 0 < i < N$$

avec C_t est la classe rendu par le classifieur à la frame t. $P_{i,t}$ représente la frame t du prototype i. Q_t la frame t du geste requête.

3.5 Conclusion

Dans cette section, nous avons présenté le moteur de reconnaissance de gestes basé sur l'algorithme DTW qui a l'avantage de ne pas avoir le besoin d'une phase d'apprentissage pour la reconnaissance des gestes. Cet algorithme mesure la similarité entre les prototypes représentant les classes avec les exemples en entrée. On a aussi présenté les propriétés à ajouter au classifieur : à savoir le rejet des classes pour minimiser l'espace des hypothèses en élaguant les classes qui ne

ressemblent pas à l'amorce du tracé dès que l'utilisateur trace un niveau du menu. La validation de notre approche en faisant des expériences sur sa capacité d'identifier les gestes en rejetant le plus de classes improbables possibles confirmeront nos solutions proposées dans cette section.

Chapitre 4

Expérimentations et résultats

Dans la section suivante, nous allons introduire les différents outils utilisés pour l'implémentation du menu. Ensuite, nous présentons le prototype. et la compagne de saisie réalisée pour la collecte d'une base de données afin de mener les expériences de classification. Nous présentons à la fin les résultats obtenus dans la classification des gestes.

4.1 Mode Novice

4.1.1 Outils utilisés

Le menu a été conçu dans le langage C# en utilisant la technologie *Windows Presentation Foundation*(WPF). WPF combine interfaces d'application, graphismes en 2D, 3D et gestion des multimédias. Son moteur de rendu vectoriel utilise l'accélération matérielle des cartes graphiques modernes. Cela rend l'interface plus rapide, évolutive et indépendante de la résolution [1]. Le but de WPF est de fournir un modèle de programmation unique pour le bureau et pour les applications sur le Web, beaucoup plus élaboré que le modèle classique de Windows. Il fournit des éléments d'interface graphique : fenêtres, boutons, champs de texte, menus, listes... Leur description se fait en XAML *eXtensible Application Markup Language*.
XAML est un langage à balise utilisé pour le développement des applications extensibles en ajoutant des composants. XAML permet donc d'intégrer des commandes, de définir des propriétés des éléments et de leur associer des événements [2].

WPF sépare l'apparence d'une interface utilisateur de son comportement. L'apparence est généralement spécifiée dans le XAML, le comportement est mis en œuvre dans un langage de programmation comme le C# ou Visual Basic.

Par ailleurs, puisque nous travaillons sur un tablet PC, la gestion de l'encore électronique est assurée en utilisant la boite à outils de *Microsoft* : *Tablet PC SDK* qui offre la gestion des applications à base d'interaction manuscrite (stylet, doigt).

1. http ://www.wpftutorial.net
2. http ://www.xaml.fr

4.1.2 Prototype

Le prototype réalisé illustre l'ensemble des propriétés citées dans la section 2. Il est constitué de trois niveaux représentant une hiérarchie géographique : continents → pays → villes. La figure 4.1 illustre l'interface qui contient le menu. Elle contient une zone d'affichage du menu, et un panel regroupant quelques options d'affichage. Dans la zone d'affichage le menu est développé dans son troisième niveau en passant par les items "Europe", "Espagne" et en affichant les items représentant les villes du sous-menu "Espagne".

Nombre de branches

Pour ce prototype, nous avons fixé le nombre de pétales à 6 par niveau, ce qui nous donne un angle de $\pi/3$ séparant chaque pétale. Ce nombre a été fixé pour minimiser la complexité d'affichage et réduire le temps de recherche en facilitant la lecture des items. En utilisant cette configuration, nous augmentons aussi les déviations imposées implicitement à l'utilisateur lors du passage d'un niveau à un autre.

Taille des menus

Chaque pétale est dessiné dans une rectangle de (largeur*hauteur=50*30) pixels, ce qui implique que menu s'étale sur une surface d'un cercle de rayon égale à la largeur d'un pétale (50 pixels).

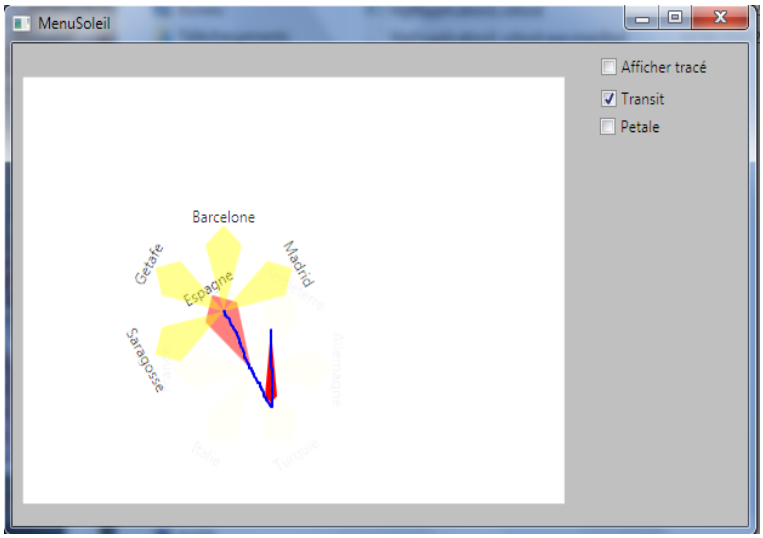


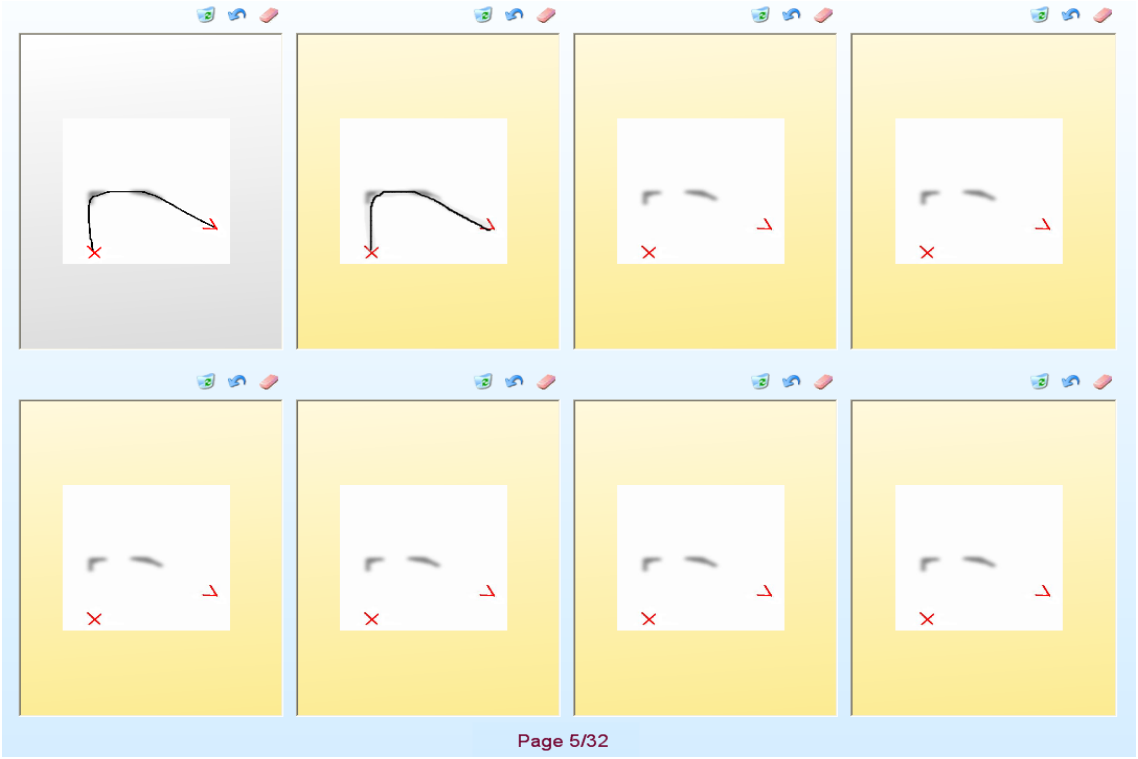
FIGURE 4.1 – Capture d'écran de l'application du prototype

4.2 Mode Expert

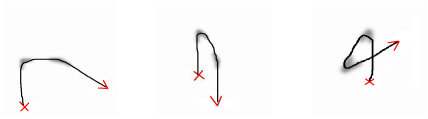
L'évaluation de notre système de menu dans le mode expert nécessite un ensemble d'utilisateurs qui ont déjà passé la phase d'apprentissage des trajectoires, ce qui peut demander un temps important. Pour la validation de l'algorithme de classification, nous avons mené une campagne de saisie en reconstituant un environnement qui permet à l'utilisateur de tracer des gestes dans le mode expert. À la fin, nous avons obtenu une base de données regroupant un échantillon significatif de gestes potentiels que des utilisateurs ont réalisé en situation comparable à une utilisation en mode expert du système.

4.2.1 Description de la base de données

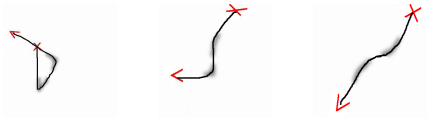
La base de données pour l'évaluation du classifieur contient 32 gestes associés à des commandes sur 3 niveaux de menus (cf. figure 4.2). Les gestes sont tracés sur un Tablet PC suivant des images indiquant la trajectoire à suivre et des zones grisées floues figurant les zones d'inertie afin d'assurer la fluidité des gestes sur l'interface illustrée dans la figure 4.2.a. Les gestes ont été saisis par 10 personnes dont l'âge varie entre 18 et 45 ans. Chaque utilisateur a tracé 10 exemples de chaque geste, ce qui fait un nombre total de 3200 exemples dans la base de données.



(a) Interace de saisie utilisée pour la collecte de la base de données



(b) (c) (d)



(e) (f) (g)

FIGURE 4.2 – Extrait des 32 modèles de la base de données

4.2.2 Expériences et résultats

Reconnaissance de gestes

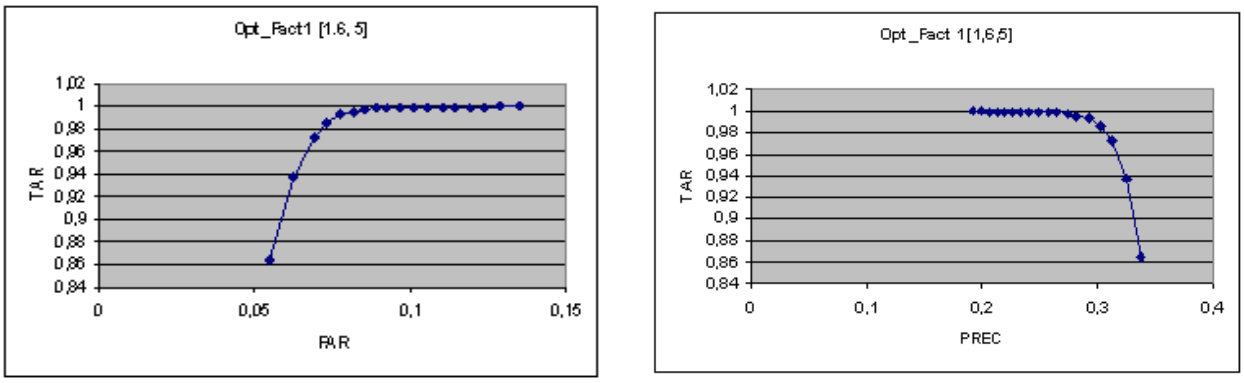
En travaillant sur la base de données que l'on vient de présenter et l'algorithme de reconnaissance gestuelle, nous avons réussi à atteindre un excellent taux de bonne classification avec un score allant jusqu'à 99.97%. Dans le tableau 4.5, nous détaillons les taux de reconnaissance pour chaque classe, on peut remarquer que dans la plupart des classes la classification est parfaite, sauf pour quelques unes où a eu quelques erreurs.

Rejet

Rappelons que le rejet que l'on a implémenté consiste à éliminer les classes à chaque transition d'un niveau à un autre. Pour ce, nous avons utilisé les courbes ROC afin de calibrer le bon seuil qui élimine le maximum de fausses classes le plus tôt possible tout en gardant la meilleure. L'évaluation consiste à mesurer le taux de rejet à chaque niveau et à vérifier en parallèle que la reconnaissance n'a pas été affectée.
Étant donné que notre base de données contient des gestes constitués de 3 niveaux, le facteur d'élimination ρ a été calculé pour chaque niveau.

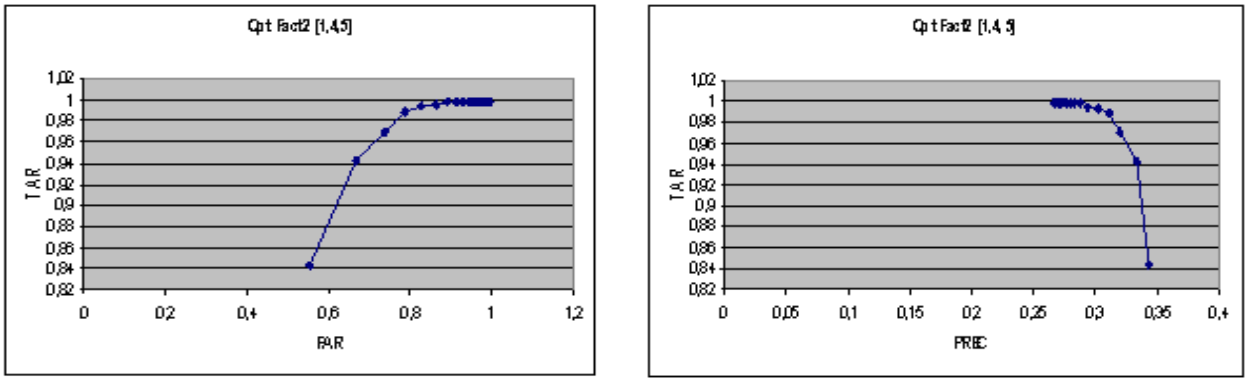
- pour la première transition nous avons fait varier ρ_1 dans l'intervalle [2..5] avec un pas de 0.2, les résultats sont illustrés dans les courbes ROC de la figure 4.3. Ces résultats signifient que le classifieur est capable de supprimer en moyenne 28 classes sur 32 seulement avec un tiers du tracé. Ceci simplifie la reconnaissance dans la deuxième transition en n'ayant pus que 4 classes en concurrence dans le classifieur ;

- à partir des résultats obtenus dans la première transition nous avons fixé ρ_1 à 3 qui est associé à un TAR=0.998% et FAR=0.089%. La valeur de ρ_2 a été prise entre]1..5] avec un pas de 0.2. Le rejet dans cette transition n'élimine presque pas de classe (figure 4.4). Ceci est dû d'une part au fait qu'il ne reste que peu de classes en concurrence (en moyenne 4) et d'autre part que les classes restantes partagent le même premier tiers du geste.



(a) (b)

FIGURE 4.3 – Courbes ROC pour le premier niveau



(a) (b)

FIGURE 4.4 – Courbes ROC pour le deuxième niveau

Reconnaissance prédictive "à la volée"

Un autre aspect, concerne la prédiction des trajectoires. Nous essayons d'anticiper la décision du classifieur et par conséquent la prochaine destination de l'utilisateur en suivant le mouvement du geste.

Pour évaluer le taux de prédiction de notre classifieur, nous avons utilisé les connaissances *à priori* des formes des gestes et les connaissances sur les directions qui forment le tracé. En exploitant les résultats intermédiaires obtenus à chaque niveau, nous avons considéré que la prédiction de la prochaine direction est réussie si la direction en question figure dans la réponse du classifieur (même si la classe proposée ne correspond pas à la classe recherchée). Les résultats obtenus (tableau 2) montrent que la prédiction du deuxième niveau est toujours assurée tandis que la prédiction du troisième niveau présente quelque difficultés liées à la segmentation du gestes tracé puisque nous avons découpé le geste en trois segments de tailles égales (ce qui signifie que le découpage n'est pas précis).

Conclusion

Dans cette section, nous avons présenté les différentes expérimentations réalisées durant ce stage. Les résultats obtenus confirment les bonnes propriétés de notre systèmes de menu notamment la topologie de menu et le type d'interaction qui nous ont garanti la variété des formes de gestes. Cependant il reste quelques améliorations à apporter pour la reconnaissance à la volée en segmentant mieux les gestes.

	Niveau 1		Niveau 2		Niveau 3	
Classe	Nombre de classes en concurrence	taux prédiction du 2ème niveau(%)	Nombre de classes en concurrence	taux prédiction du 3ème niveau(%)	Nombre de classes en concurrence	taux reconnaissance(%)
1	32	100	2	91.17	2	100
2	32	100	2	30.88	2	100
3	32	100	6	20.28	6	98.55
4	32	100	6	33.82	6	100
5	32	100	6	26.47	6	100
6	32	100	6	04.34	6	100
7	32	100	6	14.70	6	100
8	32	100	6	07.24	6	100
9	32	100	2	55.07	2	100
10	32	100	2	88.40	2	100
11	32	100	2	71.01	2	100
12	32	100	2	55.88	2	100
13	32	100	2	40.57	2	97.10
14	32	100	2	37.68	2	97.10
15	32	100	2	60.86	2	100
16	32	100	1	72.46	1	100
17	32	100	3	19.11	2	98.52
18	32	100	3	31.88	3	100
19	32	100	3	39.13	3	100
20	32	100	2	56.52	2	100
21	32	100	3	55.22	3	98.50
22	32	100	2	38.23	2	98.52
23	32	100	3	52.94	3	100
24	32	100	3	44.11	3	98.52
25	32	100	3	05.79	3	100
26	32	100	3	63.23	3	100
27	32	100	3	21.73	3	100
28	32	100	3	30.30	3	100
29	32	100	2	94.11	2	100
30	32	100	3	11.59	2	100
31	32	100	2	79.10	2	100
32	32	100	3	47.82	3	100

FIGURE 4.5 – Résultats obtenus avec la base de données, la première et la deuxième colonne contiennent le nombre de classes restant dans la reconnaissance et le temps de prédiction du niveau suivant. La dernière continent le temps de reconnaissance de chaque classe.

Chapitre 5

Perspectives

5.1 Travail futur

5.1.1 Évaluation de l'apprentissage

Nous allons étendre nos expérimentations à la validation du passage d'un utilisateur du mode novice au mode expert. L'objectif sera d'évaluer l'acceptabilité et l'efficacité des menus pour la tâche d'apprentissage des gestes. Nous étudierons le temps requis à l'utilisateur pour passer du mode novice au mode expert et nous évaluerons en parallèle l'acceptabilité et l'efficacité des menus en établissant des formulaires d'évaluation au près des utilisateurs. Nous estimerons aussi la qualité de la reconnaissance des gestes dans des conditions réelles et notamment l'impact de la zone d'inertie dans la forme du geste graphique lors du passage d'un niveau à un autre.

5.1.2 Combinaison mode expert et novice

En fonction des décisions prises sur l'amorce de tracé, on peut aussi combiner mode expert et mode novice dans le cas où l'utilisateur n'a appris qu'une partie du geste. Dans ce cas, il trace en mode expert ce qu'il a appris, puis il patiente, le temps de basculer en mode novice pour voir apparaître le reste des menus compatibles avec le geste amorcé.

5.1.3 Prédiction

La prédiction permettra de prendre une décision sur la nature de la commande souhaitée par l'utilisateur le plus tôt possible et permettra de passer de la reconnaissance du geste à l'interprétation des paramètres de la commande associées. Elle consiste à attribuer une classe d'appartenance à une amorce de geste le plus tôt possible. Nous essayerons dans nos travaux futurs de prolonger l'aide de l'apprentissage des gestes du mode novice en prédisant la trajectoire du geste amorcé en mode expert, ceci dans le but d'identifier au plus tôt la commande ciblée par l'utilisateur. L'un des principaux problèmes rencontrés dans cette phase est le fait que les prototypes partagent la même forme de l'amorce du tracé [10], le classifieur aura beaucoup de mal à distinguer entre eux.

Conclusion

Nous avons présenté dans ce rapport une nouvelle approche de menu dérivée des *Marking menus* dédiée à l'apprentissage du vocabulaire de gestes pour les interfaces purement gestuelles. Il fonctionne en deux modes différents : un mode novice pour aider l'utilisateur à mémoriser les gestes et un mode expert pour permettre à l'utilisateur de tracer son geste sans assistance d'un menu graphique.

En mode novice, nous proposons une topologie de menu qui induit un ensemble de gestes dont la forme est variée, sans poser de contraintes trop perturbantes. L'interaction permanente permet aussi d'assister l'utilisateur dans son geste pour atteindre les sous-menus souhaités. Elle dépend de deux zones différentes : la première est la zone de présélection qui permet à l'utilisateur de choisir son menu en estompant les items non concernés par le geste pour faciliter la lecture de l'item souhaité. La deuxième est la zone d'inertie qui permet à l'utilisateur d'enchaîner d'un niveau à un autre d'une façon naturelle et sans attendre l'expansion du sous-menu. La topologie du menu et l'interaction proposées induisent un ensemble de gestes fluides et directs.

Dans le mode expert, nous avons mis en œuvre un moteur de reconnaissance de gestes, il permet d'associer à chaque geste une commande spécifique. Afin d'avoir des menus dynamiques dans le nombre et l'emplacement des commandes, nous avons utilisé un algorithme DTW capable modéliser les classes par des prototypes calculés à partir de peu d'exemples. Ceci nous autorise donc à accorder à l'utilisateur la possibilité d'ajouter, de supprimer ou de modifier les items du menu. Nous avons aussi implémenté un mécanisme de rejet capable d'élaguer un nombre important de classes lors de la reconnaissance au moment où le geste est tracé dans le but de minimiser le temps de d'identification du geste.

Les perspectives permettent d'envisager une évaluation des capacités du menu pour l'apprentissage des commandes gestuelles à plus grande échelle et dans un cas d'utilisation plus réel.

Bibliographie

[1] Abdullah Almaksour, Eric Anquetil, Solen Quiniou, and Mohammed Cheriet. Evolving fuzzy classifiers : Application to incremental learning of handwritten gesture recognition systems. In *Proceedings of the 20th International Conference on Pattern Recognition*, 2010.

[2] Gilles Bailly, Eric Lecolinet, and Laurence Nigay. Quinze ans de recherche sur les menus : critères et propriétés des techniques de menus. In *IHM '07 : Proceedings of the 19th International Conference of the Association Francophone d'Interaction Homme-Machine*, pages 119–126, New York, NY, USA, 2007. ACM.

[3] Gilles Bailly, Eric Lecolinet, and Laurence Nigay. Flower menus : a new type of marking menu with large menu breadth, within groups and efficient expert mode memorization. In *AVI '08 : Proceedings of the working conference on Advanced visual interfaces*, pages 15–22, New York, NY, USA, 2008. ACM.

[4] Gilles Bailly, Anne Roudaut, Eric Lecolinet, and Laurence Nigay. Menus leaf : enrichir les menus linéaires par des gestes. In *IHM '08 : Proceedings of the 20th International Conference of the Association Francophone d'Interaction Homme-Machine*, pages 169–172, New York, NY, USA, 2008. ACM.

[5] Olivier Bau and Wendy E. Mackay. Octopocus : a dynamic guide for learning gesture-based command sets. In *UIST '08 : Proceedings of the 21st annual ACM symposium on User interface software and technology*, pages 37–46, New York, NY, USA, 2008. ACM.

[6] J. Callahan, D. Hopkins, M. Weiser, and B. Shneiderman. An empirical comparison of pie vs. linear menus. In *CHI '88 : Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 95–100, New York, NY, USA, 1988. ACM.

[7] Don Hopkins. The design and implementation of pie menus. *Dr. Dobb's J.*, 16(12) :16–26, 1991.

[8] Stéphane Huot and Eric Lecolinet. Archmenu et thumbmenu : contrôler son dispositif mobile «sur le pouce '. In *IHM '07 : Proceedings of the 19th International Conference of the Association Francophone d'Interaction Homme-Machine*, pages 107–110, New York, NY, USA, 2007. ACM.

[9] Gordon Kurtenbach and William Buxton. The limits of expert performance using hierarchic marking menus. In *CHI '93 : Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems*, pages 482–487, New York, NY, USA, 1993. ACM.

[10] Akihiro Mori, Seiichi Uchida, Ryo Kurazume, Rin-ichiro Taniguchi, Tsutomu Hasegawa, and Hiroaki Sakoe. Early recognition and prediction of gestures. In *ICPR '06 : Proceedings of the 18th International Conference on Pattern Recognition*, pages 560–563, Washington, DC, USA, 2006. IEEE Computer Society.

[11] Harold Mouchère. *Étude des mécanismes d'adaptation et de rejet pour l'optimisation de classifieurs : Application à la reconnaissance de l'écriture manuscrite en-ligne.* PhD thesis, INSA, 2008.

[12] Ralph Niels. *Dynamic Time Warping.* PhD thesis, Radboud University Nijmegen, 2005.

[13] Louis Vuurpijl Ralph Niels. Using dynamic time warping for intuitive handwriting recognition. In *Advances in Graphonomics, proceedings of the 12th Conference of the International Graphonomics Society (IGS2005)*, pages 217–221, Nijmegen, THE NETHERLANDS, 2007.

[14] D. Rubine. Specifying gestures by example. In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, pages 329–337. ACM New York, NY, USA, 1991.

[15] Seiichi Uchida and Kazuma Amamoto. Early recognition of sequential patterns by classifier combination. In *ICPR*, pages 1–4, 2008.

[16] Don Willems, Ralph Niels, Marcel van Gerven, and Louis Vuurpijl. Iconic and multi-stroke gesture recognition. *Pattern Recogn.*, 42(12) :3303–3312, 2009.