Numéro d'ordre 2348

Thèse

présentée devant

l'Université de Rennes I

pour obtenir le grade de

Docteur de l'Université de Rennes I

Mention : Traitement du signal et Télécommunications

par

Sébastien Marcel

Équipe d'accueil :	France Télécom R&D - CNET Lannion (DTL/DLI/TNT)
École doctorale :	Mathématiques, Informatique, Signal et Electronique
Composante universitaire :	Structure et Propriétés de la Matière

Une approche générative neuro-markovienne du traitement de séquences d'images: Application à la reconnaissance statique et dynamique des gestes de la main

Soutenue le 4 octobre 2000 devant la commission d'examen

Composition du jury:

Président	M. Laurent MICLET (Prof., ENSSAT)
Rapporteur	M. James CROWLEY (Prof., INPG)
Rapporteur	M. Samy BENGIO (Dr., IDIAP)
Directeur de Thèse	M. Christian ROUX (Prof., ENST Bretagne)
Examinateur	M. Patrick BOUTHEMY (Dir. Rech., INRIA-IRISA)
Examinateur	M. Daniel COLLOBERT (Dr. d'Etat, France Telecom R&D)

Résumé

La détection et l'analyse des personnes est un problème fondamental en vision par ordinateur et plus spécifiquement dans la conception des interfaces gestuelles. Les interfaces gestuelles basées sur l'image constituent la voie la plus naturelle pour la construction d'interfaces homme-machine évoluées. Ainsi, les machines seraient plus faciles à utiliser en associant la commande gestuelle à la commande vocale.

Cette thèse a pour ambition de présenter des techniques d'apprentissage statistique pour la reconnaissance des gestes de la main, et plus généralement pour le traitement d'images fixes et de séquences d'images animées. En effet, il est nécessaire de distinguer deux aspects des gestes de la main :

- l'aspect statique est caractérisé par une posture de la main dans une image,
- l'aspect dynamique est défini soit par la trajectoire de la main, soit par l'enchaînement de postures de la main dans une séquence d'images.

Les techniques d'apprentissage proposées se basent sur une approche générative des réseaux de neurones artificiels (RN) et des modèles de Markov cachés (HMM). En effet, en apprentissage statistique, on possède presque toujours des données relatives aux observations que l'on souhaite modéliser (les exemples). En revanche, on ne connaît pas toujours le reste de l'univers, c'est-à-dire toutes les observations qui ne sont pas celles que l'on désire (les contre-exemples). Il est donc difficile d'apprendre à séparer les exemples et les contre-exemples. L'approche générative se fonde sur la modélisation de la distribution de probabilité des exemples. L'évaluation de la distance entre une observation inconnue et cette distribution, permet de décider si cette observation est un exemple ou n'en est pas un.

D'un côté, les RN sont très efficaces pour des tâches de reconnaissance de formes, mais sont mal adaptés à la reconnaissance de données de nature temporelle. Nous montrons l'utilisation d'un modèle génératif de RN pour la reconnaissance statique des postures de la main dans des images.

D'un autre côté, les HMM sont bien adaptés à la modélisation des séquences de données temporelles, mais ne permettent pas de discriminer efficacement des séquences de classes distinctes. Il semble donc naturel d'étudier un modèle hybride qui possède les propriétés des RN et des HMM. Ainsi, nous proposons une approche générative neuro-markovienne (IOHMM génératif) pour la reconnaissance dynamique des gestes de la main dans des séquences d'images.

Pour finir, ce mémoire présente un démonstrateur de reconnaissance des gestes de la main (GHOST) qui utilise les techniques décrites.

Mots clés : Réseaux de neurones artificiels, modèles de Markov cachés, IOHMM génératifs, vision par ordinateur, reconnaissance des gestes de la main.

Abstract

People detection and analysis is a challenging problem in computer vision and more specifically in the design of gestural interfaces. Gestural interfaces based on the image is the most natural way for the construction of advanced man-machine interfaces. Thus, machines would be easier to use by associating the gestural command with the vocal command.

The goal of this thesis is to present several techniques of statistical training for the recognition of hand gestures, and more generally for fixed images processing and animated images processing. Indeed, it is necessary to distinguish two aspects of hand gestures:

- the static aspect is characterized by a posture of the hand in an image,
- the dynamic aspect is defined either by the trajectory of the hand, or by the sequence of hand postures in a sequence of images.

Suggested training techniques are based on a generative approach of artificial neural networks (NN) and hidden Markov models (HMM). On one hand, in statistical training, the observations we wish to model (examples) are almost always known. On the other hand, the remainder of the universe, in fact all observations which are not desired (counterexamples), is not always known. It is thus difficult to learn how to separate examples and counterexamples. The generative approach is based on modeling the probability distribution of the examples. The evaluation of the distance between an unknown observation and this distribution makes it possible to decide if this observation is an example or not.

On one hand, NN are very efficient for pattern recognition tasks, but are badly adapted to the recognition of data of temporal nature. We show the use of a generative model of NN for the static recognition of hand postures in images.

On the other hand, HMM are well adapted to the modeling of temporal sequences of data, but do not allow to discriminate efficiently sequences of distinct classes. Thus, it seems natural to study a hybrid model which has the properties of NN and HMM. We propose a neuro-markovian generative approach (generative IOHMM) for the dynamic recognition of hand gestures in sequences of images.

Finally, this thesis introduces a hand gesture recognition system (GHOST) which uses these techniques.

Keywords : Neural networks, hidden Markov models, Generative IOHMM, Computer vision, Hand gesture recognition.

Remerciements

Tout d'abord, je tiens vivement à remercier tous les membres de mon jury de thèse et plus particulièrement M. James Crowley, Professeur à l'Institut National Polytechnique de Grenoble, et M. Samy Bengio, Docteur et Directeur de recherche à l'Institut Dalle Molle d'Intelligence Artificielle Perceptive, pour avoir accepté d'en être rapporteur. Je tiens également à adresser mes plus vifs remerciements à mon directeur de thèse M. Christian Roux, Professeur à l'Ecole Nationale Supérieure des Télécommunications de Bretagne et responsable du département Image et Traitement de l'Information.

Je tiens tout particulièrement à exprimer ma reconnaissance et ma plus profonde gratitude en premier lieu, à Daniel Collobert qui m'a fait l'immense honneur de m'accueillir au sein de son équipe et qui m'a réellement initié aux réseaux de neurones artificiels, et en second lieu à Olivier Bernier dont la présence parfois inopinée dans mon bureau se révélait à chaque fois très enrichissante et dont le soutien dans les méandres de mes équations et de mes formules m'a été du plus grand secours. Daniel et Olivier, merci pour m'avoir donné ma chance, je souhaite ne pas vous avoir déçus.

Ce travail n'aurait, bien évidemment, jamais été possible sans le soutien financier de France Telecom. Je lui exprime toute ma gratitude et je regrette de ne pas avoir pu lui apporter plus.

Je voudrais enfin exprimer ma plus sincère sympathie à tous les membres et ex-membres de l'équipe TNT de France Telecom R&D. A chacun d'eux je réserve un petit mot, merci à Manu pour ses remarques critiques mais constructives et son anglais infaillible, merci à Michel pour ses codes sources de LISTEN qui se sont révélés primordiaux pour mon travail, merci à Raphaël pour le CGM, merci à Vincent pour ... ton flegme et ta clémence (encore pardon d'avoir eu la main un peu lourde !), merci à Fabrice pour m'avoir fait bénéficier de son savoir encyclopédique, merci à Joël pour avoir toujours été présent pour résoudre les problèmes techniques, merci à Pascal et Bernard pour m'avoir laissé du CPU et de l'espace disque, merci à Paul pour avoir déniché sans cesse les informations et les articles les plus intéressants, et enfin merci à Daniel B. dont l'humeur m'a permis de constater l'existence d'un gouffre entre l'homme et la machine.

Je remercie également chaleureusement Elisabeth, Lilian, Véronique, Annabelle et Sylvie pour toutes ces agréables soirées et pour toutes ces parties de billard inoubliables et interminables (pour certaines !!).

En un clin d'oeil, merci à toutes les personnes que j'ai cotoyé ces dernières années, Daniel P. pour m'avoir le premier fait découvrir le monde de la recherche et l'atmosphère si particulier des laboratoires et merci à Mohand B. pour m'avoir donné le goût de la recherche en informatique. Merci ensuite à toi Michel et à toi Gaby, auprès de vous j'ai appris beaucoup sur les autres et sur moi-même, c'est avec fierté que je me rappellerai à jamais de vous.

Pour finir, je remercie pour leur affection et leur soutien indéfectible toute ma famille, Papa, Maman, Papi et Mami. Je pense également à mon frangin, qu'il sache que pendant toutes ces années où il aurait pu se croire fils unique, il avait un frère qui pensait à lui.

Et je garde la meilleure pour la fin. Pour toute sa compréhension pour toutes mes absences et pour toute sa tendresse pour toute sa présence et enfin pour tout son amour, merci Christine.

Table des Matières

1 Introduction **21**

1.1 "Au doigt et à l'œil" . . . 21
1.2 Cadre de l'étude . . . 23
1.3 Organisation du mémoire . . . 24

2 Les gestes de la main **27**

2.1 Communiquer de manière non-verbale . . . 27
2.1.1 Historique . . . 29
2.1.2 Taxonomie des gestes de la main . . . 32
2.1.3 Les gestes de commande . . . 35
2.1.4 Les gestes co-verbaux . . . 35
2.2 Modélisation des gestes . . . 37
2.2.1 La main . . . 39
2.2.2 Le corps . . . 41
2.2.3 Discrétisation de l'espace corporel . . . 43
2.3 Analyse des gestes . . . 44
2.3.1 Analyse basée sur les gants numériques . . . 44
2.3.2 Analyse basée sur l'image . . . 45
2.4 Reconnaissance des gestes . . . 47
2.4.1 Reconnaissance des gestes statiques . . . 47
2.4.2 Reconnaissance des gestes dynamiques . . . 47
2.4.3 Systèmes existants . . . 49
2.5 Conclusion du chapitre . . . 52

3 Les réseaux de neurones et les modèles de Markov cachés **53**

3.1 Les réseaux de neurones . . . 53
3.1.1 Les fondements biologiques . . . 54
3.1.2 Un peu d'histoire . . . 57
3.1.3 Le neurone formel . . . 58

3.1.4 Généralisation du neurone formel 61
3.1.5 Les architectures 62
3.1.6 L'apprentissage dans les réseaux de neurones 66
3.1.7 Résultats théoriques: apprentissage par l'exemple, capacité et généralisation 79
3.2 Les modèles de Markov cachés 85
3.2.1 Introduction 85
3.2.2 Formalisation des observations et du modèle 85
3.2.3 Calcul de $\Pr(\mathbf{o}_1^T \mid \Lambda)$ 87
3.2.4 Maximiser la vraisemblance 89
3.2.5 Trouver une séquence d'états x_1^T optimale 96
3.2.6 Autres aspects des modèles de Markov cachés 98
3.3 Les modèles hybrides 100
3.3.1 Idée générale 100
3.3.2 Quelques exemples de modèles 101

4 Les Input/Output Hidden Markov Models génératifs 103
4.1 Les Input/Output Hidden Markov Models 103
4.1.1 Principe 103
4.1.2 Modélisation 104
4.1.3 Similitude des IOHMM avec des mélanges conditionnels d'experts 107
4.1.4 Apprentissage 109
4.1.5 Discussion 115
4.2 Une approche générative des IOHMM 116
4.2.1 Motivation 116
4.2.2 Modélisation 116
4.2.3 Apprentissage 117
4.2.4 Propriétés discriminantes de l'aspect génératif 122

5 Reconnaissance des gestes de la main en vision par ordinateur 123
5.1 Problématique 124
5.2 Rappels de traitement d'image 125
5.2.1 La couleur 125
5.2.2 Pré-traitements et segmentation 127
5.3 Segmentation 131
5.3.1 Approche géométrique 131
5.3.2 Approche statistique 131
5.3.3 Le modèle des blobs 132

5.3.4 La vraisemblance des blobs . . . 132
5.3.5 Ajustement des blobs . . . 133
5.3.6 Résultats . . . 135
5.4 Reconnaissance statique . . . 136
5.4.1 Reconnaissance des postures de la main . . . 136
5.4.2 Résultats sur notre base d'images de test. . . . 138
5.4.3 Résultats sur une base de Benchmark . . . 139
5.5 Détection des transitions . . . 140
5.5.1 Détection des changements de postures . . . 140
5.5.2 Description des enchaînements de postures . . . 141
5.5.3 Discrimination . . . 142
5.5.4 Résultat de détection sur les bases de séquences . . . 143
5.6 Reconnaissance dynamique . . . 144
5.6.1 Les trajectoires gestuelles . . . 145
5.6.2 Résultats comparatifs de classification . . . 147
5.6.3 Résultats de classification et de rejet . . . 150
5.7 Application à un système de reconnaissance des gestes de la main . . . 153
5.7.1 GHOST: une interface gestuelle mono-utilisateur . . . 153
5.7.2 Un espace corporel centré sur le visage . . . 154
5.7.3 Le principe de fonctionnement de GHOST . . . 156
5.8 Discussion . . . 159

6 Conclusion et perspectives **161**

A La rétropropagation du gradient de l'erreur **163**

B L'algorithme EM **165**
B.1 Convergence de l'algorithme EM . . . 165
B.1.1 Enoncé du problème . . . 165
B.1.2 Montrons la monotonie de g . . . 166
B.1.3 Montrons qu'un extremum local de Q est un extremum local de g 167
B.2 Inégalité de Jensen . . . 168
B.2.1 Propriété . . . 168
B.2.2 Preuve . . . 168
B.3 Formule de Fisher . . . 169

C Application de l'algorithme EM aux HMM **171**
C.1 Algorithme Forward-Backward . . . 171

C.1.1 Etape Forward . . . 171
C.1.2 Etape Forward . . . 171
C.2 Algorithme EM . . . 172
C.2.1 Etape E . . . 172
C.2.2 Etape M . . . 173

D Application de l'algorithme EM aux IOHMM **179**
D.1 Algorithme EM : Etape E . . . 179
D.1.1 Calcul de $\hat{g}_{i,t}$ . . . 179
D.1.2 Calcul de $\hat{h}_{ij,t}$ . . . 180
D.1.3 Calcul de $\hat{\alpha}_{i,t}$ . . . 180
D.1.4 Calcul de $\hat{\beta}_{i,t}$ . . . 181
D.2 Algorithme EM : Etape M . . . 182
D.2.1 Calcul des dérivées partielles $\frac{\partial \varphi_{ij,t}}{\partial s_{ij}}$ . . . 182
D.2.2 Calcul des dérivées partielles de $\frac{\partial Q(\Theta,\hat{\Theta})}{\partial \theta_{kj}}$ . . . 182
D.2.3 Calcul des multiplicateurs de Lagrange pour résoudre $\frac{\partial Q(\Theta,\hat{\Theta})}{\partial w_{ijk}}$ . . . 183
D.2.4 Calcul des multiplicateurs de Lagrange pour résoudre $\frac{\partial Q(\Theta,\hat{\Theta})}{\partial z_{qik}}$ . . . 183

E Application de l'algorithme EM aux GIOHMM **185**
E.1 Algorithme EM : Etape E . . . 185
E.1.1 Calcul de $\hat{g}_{i,t}$ . . . 185
E.1.2 Calcul de $\hat{h}_{ij,t}$ . . . 186
E.1.3 Calcul de $\hat{\alpha}_{i,t}$ . . . 187
E.1.4 Calcul de $\hat{\beta}_{i,t}$ . . . 187

F Les moments de Zernike **189**
F.1 Introduction . . . 189
F.2 Rappel sur les moments . . . 189
F.3 Fonctions définies sur le disque unité . . . 190
F.4 Invariance de forme . . . 191
F.4.1 Invariance en translation . . . 191
F.4.2 Invariance en changement d'échelle . . . 192
F.4.3 Invariance en rotation . . . 194
F.5 Calcul à partir d'images numériques . . . 196
F.5.1 Calcul du centroïde . . . 196
F.5.2 Calcul du rayon d'étendue . . . 197

F.5.3 Calcul des moments 197
F.5.4 Analyse des différentes erreurs 198
F.6 Les moments 201
F.6.1 Les moments géométriques 201
F.6.2 Les moments de Zernike 201
F.6.3 Les moments de Fourier-Mellin 203

G Les séquences de postures de la main **205**

Bibliographie **207**

Liste des Figures

Les gestes de la main **27**

2.1 Expression d'une envergure 28
2.2 "Mon oeil" : Je ne te crois pas ! 28
2.3 Encouragement (en France) ou Insulte (en Iran) 28
2.4 Le clown jongleur 30
2.5 Le singe fumeur 31
2.6 Le pierrot écrivain 31
2.7 Mouvements d'un automate simple 31
2.8 Modèle hiérarchique des gestes 33
2.9 Puissance d'expression gestuelle 34
2.10 Représentation de la dimension temporelle du geste 36
2.11 Les muscles dorsaux de la main 39
2.12 Les articulations de la main 40
2.13 Modèle anthropométrique de la main 40
2.14 Les mouvements des doigts de la main 41
2.15 Un modèle du corps en trois dimensions 42
2.16 Le modèle anthropométrique du corps 42
2.17 Discrétisation de l'espace pour l'emplacement des mains 43

Les réseaux de neurones et les modèles de Markov cachés **53**

3.1 Un neurone biologique 55
3.2 Un réseau de neurones biologique 56
3.3 Une synapse 56
3.4 Le neurone formel 58
3.5 Le neurone formel est un séparateur linéaire 59
3.6 Séparation linéaire sans biais 59
3.7 Séparation linéaire avec biais 59
3.8 La fonction signe 60

3.9 La fonction Heaviside 60
3.10 Le neurone formel général 60
3.11 La fonction sigmoïde 62
3.12 Topologie à connexions totales 62
3.13 Carte topologique de Kohonen 63
3.14 Fonction d'interaction du type Gaussien 63
3.15 Le chapeau Mexicain 63
3.16 Topologie à connexions en couches 64
3.17 Topologie à connexions récurrentes 65
3.18 Apprentissage Hebbien 67
3.19 Le Perceptron de Rosenblatt 69
3.20 Le ET logique est linéairement séparable 70
3.21 Le OU logique est linéairement séparable 70
3.22 Le OU-exclusif n'est pas linéairement séparable 70
3.23 Le Perceptron multicouches 71
3.24 Le problème du Bouc émissaire 72
3.25 Le principe d'une fonction de coût et illustration des minima locaux . . . 72
3.26 Le principe du gradient 73
3.27 La procédure du gradient 73
3.28 Le principe du moment d'inertie 76
3.29 Illustration de la capacité 79
3.30 Illustration de la notion de généralisation 81
3.31 Borne de l'erreur à nombre d'exemples fixe 82
3.32 Borne de l'erreur à capacité fixe 82
3.33 Critère d'arrêt de l'apprentissage 83
3.34 Le modèle du HMM 86
3.35 Illustration du calcul de la variable α 89
3.36 Illustration du calcul de la variable β 92
3.37 Illustration du calcul de la probabilité $\hat{h}_{ij,t}$ 93
3.38 Illustration de l'algorithme de **Viterbi** 96
3.39 Topologie gauche-droite 98
3.40 Représentation spatiale des états 99

Les Input/Output Hidden Markov Models génératifs **103**
4.1 Le modèle du IOHMM 104
4.2 Architecture du IOHMM 105
4.3 Les réseaux du IOHMM 106

4.4 Un mélange conditionnel d'experts . . . 107
4.5 La sortie globale du IOHMM . . . 108
4.6 Le modèle du IOHMM génératif . . . 117

Reconnaissance des gestes de la main en vision par ordinateur 123

5.1 Images de type visiophone individuel à analyser . . . 123
5.2 L'oeil . . . 125
5.3 Spectre de la lumière . . . 125
5.4 Modèle de Munsell des trois paramètres physiques de la couleur . . . 126
5.5 Espace TSL . . . 126
5.6 Espace de couleur CIE xyY . . . 126
5.7 Le triangle de Maxwell . . . 127
5.8 Image originale à filtrer . . . 127
5.9 Pixels de teinte chair . . . 127
5.10 Image avant rehaussage . . . 128
5.11 Image après rehaussage . . . 128
5.12 Image avant lissage . . . 129
5.13 Image après lissage . . . 129
5.14 Les contours . . . 129
5.15 Le mouvement . . . 130
5.16 Zones connexes de teinte chair à l'image t . . . 131
5.17 Zones connexes de teinte chair à l'image $t+1$ . . . 131
5.18 Exemple de blobs du visage et de la main sur fond uniforme . . . 135
5.19 Exemple de blobs du visage et de la main sur fond complexe . . . 135
5.20 Variation de la surface d'une zone et d'un blob sur une séquence visage seul . . . 135
5.21 Le modèle génératif contraint . . . 136
5.22 Projection d'un contre-exemple sur l'ensemble des exemples . . . 136
5.23 Distribution de la distance entre l'entrée et la sortie . . . 137
5.24 Exemples d'images de postures de la main extraites de la base HandTNT-B 137
5.25 Jochen-Database . . . 139
5.26 Vocabulaire de gestes de la main . . . 141
5.27 Réduction de la zone à partir du masque binaire de teinte chair, du centroïde et du rayon d'étendue . . . 142
5.28 L'erreur successive sur la valeur des moments pour une séquence Clic met en évidence par des pics d'erreurs les zones de transitions . . . 143

5.29 Trajectoires des gestes déictiques et symboliques dans l'espace corporel centré sur le visage . . . 145
5.30 Histogramme de la sortie du MLP pour les gestes de l'ensemble de test . 147
5.31 Distribution de la sortie globale du IOHMM en fonction de la durée t de la séquence . . . 149
5.32 Topologie du GIOHMM utilisé pour reconnaître les gestes . . . 150
5.33 Probabilité des états associés à un geste déictique . . . 151
5.34 Probabilité des états associés à un geste de rétractation déictique . . . 151
5.35 Exemples de chiffres manuscrits . . . 152
5.36 Le log de la Vlikelihood d'un geste et d'un chiffre manuscrit . . . 152
5.37 Geste de commande . . . 154
5.38 Commande de sélection . . . 154
5.39 Expression d'une envergure . . . 155
5.40 L'espace corporel centré sur le visage dans une image . . . 155
5.41 Modèle anthropométrique du corps . . . 155
5.42 Images à analyser . . . 156
5.43 Représentation des blobs sur le visage et la main . . . 156
5.44 Détection du visage dans les blobs . . . 157
5.45 Représentation de l'espace corporel centré sur le visage . . . 157
5.46 Reconnaissance d'un geste déictique et d'un geste symbolique . . . 157
5.47 L'utilisateur lève la main avec les doigts serrés et l'horloge s'affiche . . . 158
5.48 La messagerie s'affiche . . . 158
5.49 Une vidéo apparaît . . . 158
5.50 L'utilisateur serre le poing et les fenêtres se ferment . . . 158

Les moments de Zernike **189**

F.1 Evaluation des points d'approximation de l'intégrale . . . 198
F.2 Approximation d'une région circulaire discrète . . . 199
F.3 Polynôme radial $R_{pq}(\rho)$ à l'ordre 10 pour toutes ses répétitions q . . . 202
F.4 Polynôme radial $Q_p(\rho)$ à l'ordre 1, 2, 3, 4 et 5 . . . 204

Les séquences de postures de la main **205**

G.1 Exemple d'une séquence Clic . . . 205
G.2 Exemple d'une séquence Stop-Grasp-OK . . . 205
G.3 Exemple d'une séquence No . . . 206
G.4 Exemple d'une séquence Rotate . . . 206

Liste des Tableaux

Reconnaissance des gestes de la main en vision par ordinateur **123**

5.1 Taille des réseaux de neurones CGM utilisés 138

5.2 Résultats obtenus sur la base de test HandTNT-B sur fonds uniformes et sur fonds complexes donnés avec un intervalle de confiance à 95% . . 139

5.3 Résultats obtenus sur la base de Jochen Triesch sur fonds uniformes et fonds complexes donnés avec un intervalle de confiance de 95%. 140

5.4 Les enchaînements de postures utilisés 142

5.5 Résultats obtenus pour les séquences Clic et Stop-Grasp-OK donnés avec un intervalle de confiance à 95% 144

5.6 Résultats obtenus pour les séquences No et Rotate donnés avec un intervalle de confiance à 95% 144

5.7 Description de la base de gestes 146

5.8 Taux de classification pour les HMM discrets, les IOHMM entre **90**% et **100**% de la séquence, et le MLP utilisant les trajectoires interpolées . . . 149

5.9 Taux de classification pour le GIOHMM entre **90**% et **100**% de la séquence 151

5.10 Taux de rejet des chiffres manuscrits par le GIOHMM à la fin de la séquence 153

CHAPITRE 1

Introduction

1.1 "Au doigt et à l'œil"

Mardi, c'est le jour de la semaine que j'aime le plus. J'attends toujours avec autant d'impatience de pouvoir me remettre au travail après ces longs week-ends.

Il est 9 : 00 AM, j'entre dans mon bureau, les lumières s'allument, ma console s'active et une voix féminine se fait entendre:

- "Bonjour, Sam".
- "Bonjour, LINEA", réponds-je alors que je retire ma veste.
- "Que s'est-il passé ce week-end ?".
- "Vous avez reçu de nouveaux messages. Voulez-vous en savoir plus ?".
- "Non, plus tard. Qu'y a-t-il d'autre ?".
- "Les programmes d'apprentissage que vous avez lancés jeudi sont terminés. Voulez-vous voir les résultats maintenant ?".
- "Oui, donne-les moi sur grand écran, je te prie".

Je suis ingénieur Recherche et Développement au département des Techniques Novatrices pour les Télécommunications, un des centres de recherche de GlobalFive OmniTech, le plus gros consortium d'opérateurs téléphoniques de la planète.

J'adore mon job, je travaille sur de nouveaux algorithmes pour les ordinateurs à apprendre de deuxième génération. En effet, grâce à l'essor de la téléphonie mobile, d'Internet et de la robotique au début des années 2000, les techniques d'apprentissage par ordinateur, comme les réseaux de neurones artificiels, ont connu leurs heures de gloire.

A cette époque, les chercheurs ne disposaient pas des outils d'aujourd'hui. De nos jours, les interfaces évoluées à reconnaissance vocale et gestuelle nous permettent de

dialoguer et d'interagir avec des assistants virtuels pour nous aider dans nos tâches quotidiennes.

Les résultats des programmes commencent à s'afficher sur l'écran. Ma journée s'annonce bien remplie. Il me faut un café !

- "Je vous signale que le café est prêt." Si je ne connaissais pas le moteur d'IA de LINEA, je pourrais croire qu'elle lit dans mes pensées. En fait, il ne lui a fallu que quelques jours pour repérer mes habitudes et anticiper mes désirs.

- "Merci, LINEA. Qui l'a préparé ?"

- "C'est Sébastien, il est déjà là depuis une heure. Voulez-vous lui envoyer un message ?"

- "Non merci, j'irai le voir plus tard." Ah, ces thésards ! Les premiers arrivés et les derniers partis.

- "LINEA, éteint la cafetière, je ne veux pas d'un café recuit, j'irai le chercher plus tard. Montre-moi les résultats plus en détail."

Sur le grand écran, chaque programme est représenté par des courbes de performances et des distributions de données. Mon attention est attirée par les résultats d'un des derniers modèles sur lequel je travaille.

- "LINEA, donne-moi plus de détails sur ça." Je désigne du doigt les résultats qui m'intéressent, et aussitôt, ces derniers occupent l'écran. Les performances sont intéressantes. Il me faut en savoir plus. Comment la structure interne du réseau a-t-elle évoluée ?

- "LINEA, visualise la structure du modèle en trois dimensions et passe en mode immersion." Grâce à ce mode d'interaction, je rentre dans mon modèle, je peux avoir une vision globale du système, et d'un geste de la main, obtenir une vue plus précise de la sous-structure du réseau.

- "Excusez-moi de vous déranger. Vous venez de recevoir un nouveau message. Je vous signale qu'il provient de Manu et qu'il vous a déjà laissé un message ce week-end."

- "Bien, mets-moi directement en liaison vidéo avec lui, puis désactive-toi c'est une communication personnelle."

- "Bien, je désactive le mode immersion. La liaison vidéo est établie sur grand écran."

- "Salut, Manu ..."

LINEA voit, entend et exécute. Elle incarne peut-être l'assistante virtuelle du futur. Cette interface personnelle sait être prévenante lorsqu'elle informe que le café est prêt ou qu'elle interrompt Sam dans son travail si un message, peut-être important, vient d'arriver. Mais, LINEA sait aussi s'effacer lorsqu'on le lui demande pour respecter la vie privée.

LINEA est une interface multi-modale qui repose sur des technologies de reconnaissance de la parole, des visages et des gestes. Si les technologies de reconnaissance de la parole sont étudiées depuis longtemps et sont déjà à un stade avancé, en comparaison, la reconnaissance des gestes n'en est qu'à ses balbutiements. Pour imaginer tout le potentiel de la reconnaissance des gestes, il faut avoir à l'esprit tout ce qu'elle recouvre. Les expressions faciales nous informent sur les émotions de la personne, les attitudes corporelles nous renseignent sur ses activités et les mouvements de la main désignent des objets ou illustrent des métaphores verbales. Dans ce mémoire, nous allons nous concentrer exclusivement sur la reconnaissance des gestes de la main.

1.2 Cadre de l'étude

L'équipe TNT[1] de France Télécom Recherche et Développement[2] de Lannion a mis au point LISTEN (Locating Individual Speakers and Tracking ENvironment), un système de suivi de personnes pour des applications de terminaux individuels et de visioconférence "personne libre" [Collobert et al., 1996]. Les visages sont détectés par un réseau de neurones artificiels donnant la position et l'échelle du visage dans l'image. Puis, une caméra et une antenne acoustique se focalisent automatiquement sur le visage de l'utilisateur. La personne est détectée, puis suivie en permanence. Dans ce système, il n'y a pas de commande explicite. LISTEN n'a pas été conçu pour être commandé par l'utilisateur. L'extension aux interfaces homme-machine est toute naturelle et nous amène donc à nous intéresser à l'intégration de la modalité gestuelle au système de suivi actuel.

Nous envisageons un système de visioconférence fonctionnant sur le principe de LISTEN, mais interprétant un geste de la main désignant un objet de la scène (tableau, autre personne, feuille) et effectuant automatiquement la mise au point sur celui-ci. Nous entrevoyons également une application de terminal individuel, où le système détecte et reconnaît des commandes gestuelles afin de déplacer, de modifier ou de déformer des objets virtuels ou distants, et de déclencher ou d'interrompre l'exécution d'un programme.

[1] Techniques Neuromimétiques pour les Télécommunications.

[2] Depuis le mois d'avril 2000, le Centre National d'Etude des Télécommunications a changé de nom. A présent, il se nomme France Télécom R&D.

1.3 Organisation du mémoire

Cette thèse a pour ambition de présenter des techniques d'apprentissage statistique pour le traitement d'images et de séquences d'images. Ces techniques d'apprentissage se basent sur une approche générative des réseaux de neurones artificiels et des modèles de Markov cachés. En effet, en apprentissage statistique, on possède des données relatives aux observations que l'on souhaite modéliser. Par contre, on ne connaît pas toujours le reste de l'univers, c'est-à-dire toutes les observations qui ne sont pas celles que l'on désire. Il est donc difficile d'apprendre à séparer les exemples et les contre-exemples. L'approche générative se fonde sur la modélisation de la distribution de probabilité des données d'entrées. L'évaluation de la distance entre un exemple inconnu et cette distribution permet de décider si cet exemple appartient ou non aux observations. Nous utilisons, d'une part, un modèle génératif de réseau de neurones pour la reconnaissance statique des postures de la main dans des images. Nous proposons, d'autre part, une approche générative neuro-markovienne[3] pour la reconnaissance dynamique des gestes de la main dans des séquences d'images.

Le chapitre 2 introduit la communication non verbale, les gestes en général et les gestes de la main en particulier. Ce chapitre présente en détail l'activité gestuelle humaine et propose une classification des différentes catégories de gestes, ainsi qu'un schéma pour les modéliser.

Le chapitre 3 s'étend sur des techniques d'apprentissage statistique telles que les réseaux de neurones artificiels et les modèles de Markov cachés. Les réseaux de neurones sont principalement utilisés pour des tâches de reconnaissance de formes statiques. Et les modèles de Markov cachés sont mieux adaptés à la modélisation de processus dynamiques. Ce chapitre s'achève sur un état de l'art des techniques hybrides qui combinent les réseaux de neurones artificiels et les modèles de Markov cachés pour exploiter les propriétés des uns et des autres.

Le chapitre 4 présente un modèle hybride à base de réseaux de neurones et de modèles de Markov cachés. Ce modèle est basé sur une approche générative des "Input-Output Hidden Markov Models" pour la reconnaissance de séquences de données temporelles. Une séquence de données d'entrée est proposée au modèle qui produit, de manière synchrone, une séquence de données de sortie et génère une probabilité d'observation de la séquence d'entrée. Pour une tâche de classification, la séquence de sortie produite peut être considérée comme la classe associée à la séquence d'entrée. De plus, la séquence de probabilité d'observation générée peut être utilisée pour rejeter une séquence d'entrée non-conforme au modèle, c'est-à-dire qui n'a pas été apprise.

Le chapitre 5 illustre les applications des techniques présentées pour la reconnaissance des postures de la main dans les images, la détection des changements de postures de la main et la reconnaissance des gestes de la main dans les séquences d'images. Nous utilisons un modèle génératif pour la reconnaissance statique des postures de la main dans des images, et nous proposons un nouveau modèle hy-

[3] à base de réseaux de neurones et de modèles de Markov cachés.

bride pour la reconnaissance dynamique des gestes de la main dans des séquences d'images.

Enfin, le dernier chapitre, le chapitre 6, achève cette thèse. Il présente une synthèse des techniques utilisées et établit les perspectives d'application du modèle génératif neuro-markovien.

CHAPITRE 2

Les gestes de la main

Les ordinateurs sont de plus en plus faciles à utiliser, notamment grâce à des interfaces ergonomiques et intuitives basées essentiellement sur l'écran, le clavier et la souris. L'utilisateur est en contact physique avec les systèmes d'interaction. Il touche son écran tactile pour sélectionner une zone de l'écran, il prend sa souris pour naviguer dans un menu, ou il utilise un raccourci clavier pour accéder plus rapidement à une fonctionnalité. Ces périphériques sont soit limités en vitesse (on préférera le clavier à la souris pour une action rapide) et en facilité d'utilisation (les raccourcis clavier ne sont pas intuitifs et demandent un effort de mémoire). Nous devons donc imaginer des interfaces utilisateurs plus riches, et qui n'imposent pas de contact physique. Ces interfaces avancées pourront être rapidement maîtrisées après un processus d'apprentissage de l'utilisateur associé à une rétroaction avec l'interface.

Une nouvelle forme d'interaction, basée sur des techniques de reconnaissance de la parole, utilise déjà le canal vocal et permet à l'utilisateur de contrôler des systèmes intégrant une fonction d'interprétation. Les machines seraient plus faciles à utiliser si on pouvait les contrôler à travers la langue naturelle ou même à travers les gestes. Ainsi, de nombreuses recherches se sont orientées vers l'étude du canal gestuel, car il est un moyen de communication non-verbal très expressif qui permet une interaction plus naturelle et qui complète le canal verbal. Les interfaces du futur seront des interfaces multimodales, qui intégreront les techniques classiques d'interaction ainsi que celles basées sur la voix et le geste.

Dans ce chapitre, nous nous intéresserons à cette modalité gestuelle en étudiant le rôle qu'elle joue dans la communication humaine, afin de connaître les différents modèles de représentation des gestes. Puis, après un aperçu des différentes techniques de modélisation, d'analyse et de reconnaissance des gestes, nous proposerons les objectifs envisagés et les techniques retenues pour reconnaître les gestes de la main.

2.1 Communiquer de manière non-verbale

Le geste est pour l'homme, un moyen conscient et inconscient de communication non-verbal naturel et très expressif, dont l'aspect dynamique est porteur d'informations.

En effet, l'utilisateur peut exprimer une commande en une seule action, et peut par des variations dans l'aspect dynamique, préciser des paramètres tels que la portée de la commande ou les objets la concernant.

Figure 2.1: Expression d'une envergure

De plus, les gestes sont intuitifs et le plus souvent universels (confirmation ou désapprobation (Figures 2.2 et 2.3), expression d'une localisation, d'une envergure (Figure 2.1) ou d'une distance).

Figure 2.2: "Mon oeil" : Je ne te crois pas !

Figure 2.3: Encouragement (en France) ou Insulte (en Iran)

Utiliser les gestes supprime la surcharge cognitive relative à l'apprentissage d'un nouveau périphérique. Ainsi, le geste offre à de nombreuses applications une interaction directe, plus naturelle, plus concise et plus efficace que la souris.

L'analyse et la reconnaissance des gestes peuvent être utiles dans de nombreux domaines d'applications :

- robotique,
- réalité virtuelle augmentée[1] [Huang et al., 1996],
- interfaces multimodales contrôlées par la voix et le geste [Waibel et Vo, 1993] [Cassell et al., 1999],
- analyse de séquences d'images sportives, de chef d'orchestres ou de chorégraphies [Campbell et Bobick, 1995],
- traduction automatique de la langue des signes [Starner et Pentland, 1995].

2.1.1 Historique

L'étude des gestes à travers le temps

Les gestes ont attiré l'attention des écrivains depuis au moins deux millénaires. L'intérêt originel des gestes était dans la rhétorique. Au premier siècle, Quintilian, spécifia en détail les gestes que les orateurs de son époque utilisaient durant leurs discours. Au XVIII^{ème} siècle, les philosophes sont devenus profondément concernés par l'origine du langage et par les fondements universels de la raison. Un certain nombre de penseurs ont supposé que les premiers langages étaient gestuels. Puis vers 1970, les psychologues établissent un lien entre les gestes et la pensée. Par la suite, les premières catégorisations des gestes font leur apparition.

(Histoire de l'étude du geste extrait de [Kendon, 1982]).

La langue des signes

L'histoire du geste humain est liée à celle de la langue des signes. En effet, le geste est un moyen d'expression et de communication puissant pour les personnes sourdes ou muettes. Platon et Aristote pensaient qu'une personne qui ne parlait pas ne pouvait pas raisonner. Il faut attendre 1776 pour qu'à Paris, l'Abbé Charles Michel de l'Epée [de l'Epée, 1784] fonde le premier Institut destiné à l'éducation des sourds-muets en s'appuyant sur la langue des signes. Aux Etats-Unis, la langue des signes est beaucoup plus active, ainsi qu'en témoigne l'Université de "Gallaudet", exclusivement destinée aux déficients auditifs. Peu de temps après, en 1779, Pierre Desloges, bien qu'étant sourd, écrivit un livre sur la langue des signes et sa structure. A partir de cette époque, on se met à penser que les gestes expriment la pensée humaine, tout comme la langue orale. Enfin en 1791, l'Assemblée Nationale promulgue une loi reconnaissant la légitimité des droits de l'homme pour les personnes atteintes de

[1] Dans les systèmes de réalité virtuelle augmentée, les informations en provenance du monde virtuel sont superposées à celles du monde réel.

surdité. Au XIX^{ème} siècle, les premières études linguistiques apparaissent, mais les oppositions avec la culture orale persistent. En 1968, le droit à la parole est donné aux minorités linguistiques et en 1975, le journal télévisé est traduit dans la langue des signes française sur la 2ème chaîne de télévision. Enfin, en 1992, Emmanuelle Laborit, comédienne sourde reçoit le Molière pour la pièce de théâtre "Les enfants du silence".

(Histoire de la langue des signes extraite partiellement de [Braffort, 1996]).

Figure 2.4: Le clown jongleur

Les automates

L'homme a réalisé des automates depuis la haute antiquité. Il nous en reste des fragments, quelques descriptions, des dessins et des légendes. Mais, si le désir de créer des pièces animées reste constant, le caractère des créateurs d'automates a varié au fil des époques et des civilisations qui les ont marquées, de leurs préoccupations et de leurs rêves. C'est au XVIII^{ème} siècle que l'androïde voit le jour. L'androïde est un automate ayant figure humaine qui présente des fonctionnalités extérieurement semblables à celles de l'homme. Ces créations animées par des mécanismes étaient capables de jongler (Figure 2.4), de dessiner, d'écrire (Figure 2.6), de jouer d'un instrument de musique, voire même d'articuler quelques paroles. Ainsi, si l'androïde a tant fasciné le siècle des lumières, c'est probablement parce que l'homme y voyait le symbole du triomphe de la raison humaine.

Figure 2.5: Le singe fumeur

Figure 2.6: Le pierrot écrivain

Vaucanson, un célèbre créateur d'automates, non content d'imiter les gestes et les attitudes d'un être vivant, tenta de reproduire leurs mouvements internes comme la modulation du souffle chez un joueur de flûte, ou la digestion du canard. Le singe fumeur (Figure 2.5), par exemple, illustre parfaitement le mécanisme du souffle chez le fumeur. Cet automate est capable d'inspirer et d'expirer la fumée d'une cigarette. Vaucanson rêvait de créer un automate qui simulerait la circulation du sang à l'intérieur du corps. Ses travaux et les travaux des autres constructeurs d'automates de la même époque n'ont d'autre but que de recréer la vie.

Figure 2.7: Mouvements d'un automate simple

Les mouvements de presque tous les automates sont inscrits dans un cycle qui se répète sans cesse. La durée d'un cycle complet est un élément important dans l'appréciation d'un automate. Il faut que les gestes soient accomplis à la vitesse voulue, ce qui est assez difficile à déterminer parce que, comme le théâtre, le monde des automates a ses propres conventions. Ainsi, les gestes des personnages se doivent d'être un quart de fois plus lents que chez les êtres vivants. Un autre facteur important pour apprécier un automate est la diversité des mouvements différents que le personnage exécute.

"L'Espagnole au tambourin" (Figure 2.7) est un automate simple. Elle possède quatre mouvements. Tout d'abord, elle penche le buste alternativement à gauche et à droite en un simple mouvement oscillant (sinusoïdal). Dans un second mouvement simultané au premier, elle penche la tête du coté opposé au corps. En troisième lieu, elle lève un peu le bras gauche, mais seulement lorsque le corps se penche à droite. Dans un quatrième temps, elle agite son tambourin par la rotation de son avant-bras droit.

Il faut donc trouver comment animer des tiges et des rouages de métal, comment les ajuster pour obtenir un gracieux mouvement de poignet, pour capter tout le naturel d'un haussement d'épaules, ou faire passer une lueur coquine dans des yeux de verres opaques.

(Histoire des automates et photographies extraites de "L'âge d'or des automates" de Christian Bailly).

L'analyse automatique

L'analyse automatique des gestes est assez récente. On peut dater son apparition avec celle des gants numériques en 1987, destinés essentiellement à des applications de réalité virtuelle. Les recherches se sont basées donc essentiellement sur des périphériques comme le DataGlove [VPL Research, 1987]. Plus récemment, parallèlement à la montée en puissance des ordinateurs et au perfectionnement des caméras, les chercheurs se sont orientés vers une analyse des gestes moins contraignante en utilisant des séquences d'images.

De la simulation à l'analyse automatique, il n'y a qu'un pas : la compréhension de ce que sont les gestes chez l'Homme.

2.1.2 Taxonomie des gestes de la main

Un geste conscient est motivé par l'intention de communiquer ou d'accomplir une certaine tâche (indiquer, rejeter, saisir, dessiner). Il est l'expression physique d'un concept mental [Thieffry, 1981]. Ainsi, dans la société humaine, nous communiquons à l'aide de la parole, mais aussi à l'aide de notre corps, de nos gestes et de nos yeux. Nous utilisons certains de leurs mouvements comme moyen d'expression, pour exprimer ou accentuer une idée, un sentiment ou une attitude. Nous haussons les épaules lorsque nous doutons, nous pointons les choses que nous désirons explorer.

Le geste n'inclut pas seulement l'expression corporelle. En effet, la communication non verbale passe principalement à travers les gestes de la main [Huang et Pavlovic, 1995].

Dans le geste, on s'intéresse donc à la main. Cadoz définit trois fonctions de la main qu'il considère complémentaires et imbriquées [Cadoz, 1994] : la fonction épistémique, la fonction ergotique et la fonction sémiotique.

La fonction épistémique

La main est un organe de perception. La perception tactilo-kinesthésique donne des informations sur la forme, l'orientation, la distance, la dimension des objets par l'intermédiaire du touché et de l'utilisation de mouvements exploratoires. La perception proprioceptive donne des informations sur le poids, les trajectoires, les mouvements des objets.

La fonction ergotique

La main est un organe moteur. Elle interagit avec l'environnement pour le transformer, par exemple, pour déplacer ou déformer un objet.

La fonction sémiotique

La main est un organe d'émission d'informations à destination de l'environnement, c'est-à-dire à destination de la perception visuelle ou tactile d'un ou de plusieurs agents de communication. La communication homme-machine exploite principalement la fonction sémiotique de préférence aux deux autres [Braffort, 1996].

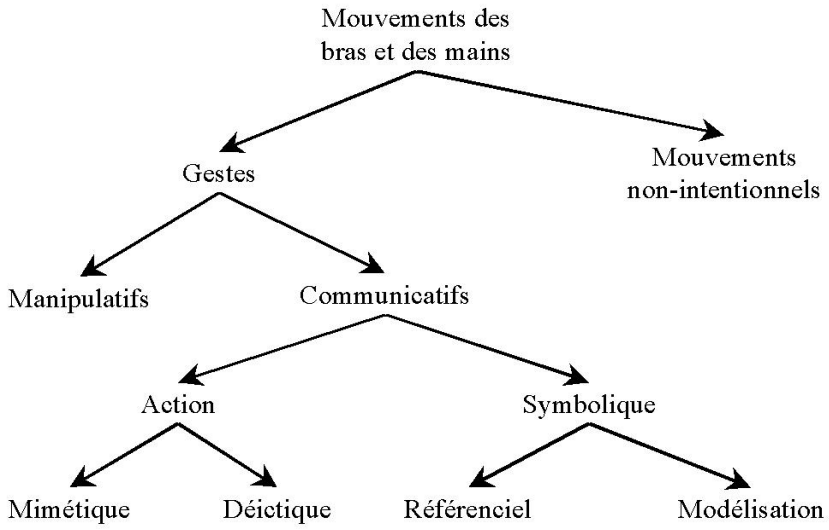


Figure 2.8: Modèle hiérarchique de classification des gestes

Divers auteurs effectuent des distinctions plus fines entre les gestes. McNeill [McNeill, 1992] considère les gestes comme iconiques[2], métaphoriques[3] ou de battement[4], alors que Kendon [Kendon et al., 1986] différencie les gestes autonomes (indépendant de la parole) et les gesticulations (associées à la parole). Plus tard, Quek [Quek, 1994] propose un modèle hiérarchique (Figure 2.8).

On peut distinguer différentes catégories formant une classification du geste humain :

- les gestes de la langue des signes [Moody, 1983],
- les gestes de commande,
- les gestes co-verbaux.

On peut représenter ces trois catégories de gestes sur un axe mesurant la puissance d'expression gestuelle (Figure 2.9).

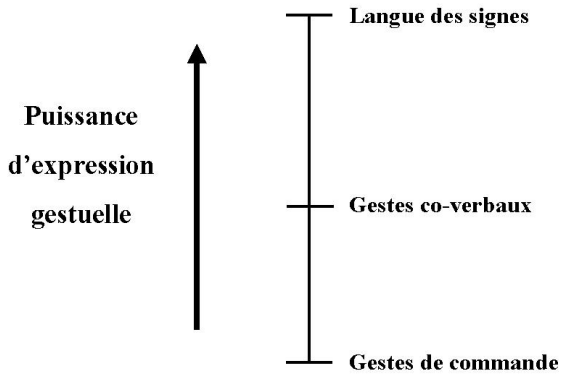


Figure 2.9: Puissance d'expression gestuelle

Les gestes de commande sont destinés à des actes de communication intentionnels dont le sens est simple et adapté au contexte de leur utilisation. Ainsi, on observe souvent une utilisation des gestes de commande dans des activités où le milieu n'est pas propice à la propagation des ondes sonores (plongée sous-marine) et où l'environnement sonore est fortement bruité (la Bourse ou les chantiers de construction).

A mi-chemin, on trouve les gestes co-verbaux[5], qui permettent d'illustrer et de compléter le canal verbal. En effet, le geste et la parole se combinent pour transmettre plusieurs informations simultanément. Par exemple, si le canal gestuel, comme nous allons le voir, est adapté à la transmission d'informations spatiales, il n'en est

[2]Les gestes iconiques sont utilisés pour représenter un objet, une action ou un événement.
[3]Les gestes métaphoriques sont utilisés pour illustrer un concept abstrait.
[4]Les gestes de battement servent à marquer le rythme du discours.
[5]On peut considérer le mouvement des lèvres comme le premier des gestes co-verbaux.

pas de même pour la parole. De même, le geste co-verbal permet de lever des ambiguïtés de certains mots du langage naturel, et à l'inverse, le contenu du message verbal permet de faciliter l'interprétation d'un geste.

Au sommet, on trouve la langue des signes qui constitue une langue à part entière, munie d'une syntaxe et autorisant même la création dynamique de signes non-standards. Les gestes de la langue des signes sont bien plus structurés et donc bien plus complexes que les gestes naturels. Pour simplifier notre travail, nous allons nous concentrer sur les deux autres catégories.

2.1.3 Les gestes de commande

Une interface gestuelle va utiliser des gestes de commande comme moyen d'interaction conversationnelle, mais aussi des gestes de manipulation pour agir sur des objets réels (robotique, télémanipulation) ou des objets virtuels (interfaces graphiques, réalité virtuelle augmentée). Un bon exemple est celui du "Click and Drag" qui, à l'image de l'utilisation de la souris, permet de manipuler des objets graphiques à l'écran. Le geste de commande est indépendant du canal verbal. Pourtant, il peut être utilisé dans un contexte co-verbal ; on parle alors de gestes symboliques ou emblématiques.

2.1.4 Les gestes co-verbaux

Les gestes co-verbaux peuvent être intégrés dans les interfaces multimodales. Associés au canal verbal, ils permettent de communiquer des informations spatio-temporelles. Ils se subdivisent en deux grandes catégories [McNeill, 1992].

- Les gestes symboliques ou emblématiques s'apparentent à des gestes de commande dans un contexte co-verbal. Leur utilisation est propre à des communautés sociolinguistiques (plongeurs, grutiers, etc.).
- Les gestes illustrateurs s'associent au canal verbal. On peut distinguer différentes sous-catégories.
    - Les gestes métaphoriques illustrent un concept abstrait. Par exemple, l'expression "la dimension de mon amour pour toi n'a d'égal que l'infini" (que l'on peut traduire "je t'aime gros comme ça") est accompagnée d'un écartement des bras pour exprimer l'intensité d'un sentiment.
    - Les gestes de battement marquent le rythme du discours. En effet, le geste est souvent lié à la parole pour marquer un point à mettre en évidence ou pour imposer son opinion. Mais, il est aussi l'apanage de la population du Midi de la France (Ne dit-on pas souvent "parler avec les mains" ?).
    - Les gestes déictiques sont des mouvements de pointage à l'aide de la main, du visage ou d'un artefact. La trace de leur mouvement dans l'espace s'apparente généralement à un segment de droite.

- Les gestes iconiques, à la différence des gestes métaphoriques, servent à dimensionner physiquement un objet, une action ou un événement. Par exemple, la phrase "j'ai pêché une truite grosse comme ça" est accompagnée d'un geste indiquant la taille du poisson. Les gestes iconiques peuvent également être subdivisés :
  - Les gestes spatiographiques permettent de désigner des objets par rapport à la position du locuteur (en face, à droite, à gauche).
  - Les gestes orientographiques [Braffort, 1996] permettent de préciser l'orientation d'un objet dans la scène. Par exemple, la phrase "ce tableau sur le mur penche" sera accompagnée d'un mouvement des mains pour indiquer comment aligner le tableau.
  - Les gestes pictographiques permettent de montrer la forme d'un objet à l'aide de primitives géométriques telles qu'un segment de droite, un arc de cercle ou un angle droit.
  - Les gestes kinégraphiques permettent d'illustrer certains types de mouvements à l'aide du déplacement des mains. Par exemple, un locuteur parle d'une route en zig-zag, il va accompagner son discours par un geste de la main effectuant le même zig-zag.

Ainsi, la configuration de la main donne des informations pictographiques, son mouvement ajoute des informations kinégraphiques, l'orientation et l'emplacement de la main procurent respectivement des informations orientographiques et spatiographiques [Braffort, 1996]. Les gestes déictiques et iconiques nous donnent des informations spatiales permettant d'effectuer des tâches de désignation, de localisation et de quantification. De plus, la dynamique du mouvement fournit des informations de répétition, de succession, de continuité ou d'arrêt. Ainsi, les gestes nous donnent des informations temporelles (Figure 2.10).

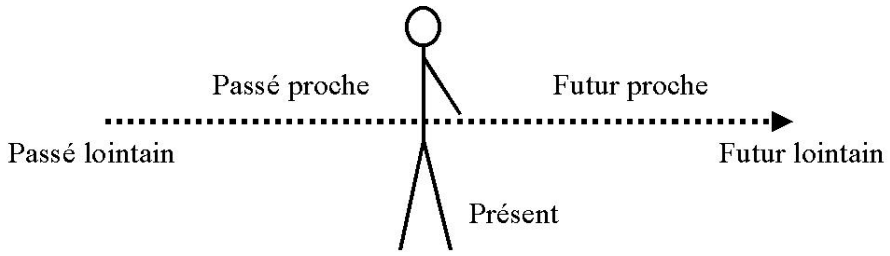


Figure 2.10: Dimension temporelle du geste dans notre culture occidentale

Dans la majorité des cultures, on peut situer le temps sur un axe dont le repère est le locuteur. Généralement, le futur est placé devant, le passé derrière et le présent est à nos pieds. De même, sur un axe frontal parallèle au corps, un événement t_1 qui est placé à gauche d'un événement t_2 indique généralement que t_1 s'est produit avant t_2.

Cependant, toutes les cultures ne représentent pas le temps de la même manière. Dans la culture des aborigènes d'Australie, par exemple, le passé est devant et le

futur est placé derrière. La logique est la suivante : on peut voir le passé car on le connaît, il est donc devant, mais on ne connaît pas le futur, il est donc invisible pour les yeux et par conséquent derrière soi.

Ainsi, on peut choisir les gestes déictiques pour préciser la notion de passé, de présent et de futur en effectuant un geste de pointage. Les gestes spatiographiques positionnent les événements dans le temps (de gauche à droite), les gestes pictographiques indiquent la durée des événements, et les gestes kinégraphiques leurs déroulements dans le temps.

Le geste n'est pas une simple position statique montrant la main sous une certaine configuration. Le geste est également un processus dynamique qui peut s'exécuter de manière localisée dans l'espace corporel (enchaînement de postures) ou s'effectuer par un déplacement plus ample dans cet espace corporel (le déplacement de la main dans l'espace corporel décrit une trajectoire). Il est donc difficile et réducteur de ramener le mouvement à des primitives géométriques simples telles que des droites, des arcs ou des cercles[6], car on perd alors toute l'expressivité d'un geste naturel. La dynamique du mouvement rend le geste imprévisible et de nombreux problèmes se posent, car l'exécution d'un geste peut varier suivant son auteur (problème stylistique) et suivant le contexte, l'humeur ou la fatigue de la personne (problème situationnel) [Morrel-Samuels, 1990].

De nombreux auteurs ont contribué à établir des modèles de représentation du corps et des gestes de la main. Une fois le modèle choisi, il faut déterminer la manière dont les gestes vont être analysés. Puis enfin, il faut mettre en œuvre les techniques de reconnaissance.

2.2 Modélisation des gestes

La réalisation d'un geste de la main peut être vue, comme un processus stochastique dans l'espace des paramètres du modèle gestuel, durant un intervalle de temps approprié [Huang et al., 1996]. En effet, les gestes de la main sont des processus dynamiques qui suivent un schéma caractéristique dans l'espace et dans le temps. On distingue trois phases dans l'accomplissement d'un geste simple [Kendon et al., 1986] :

1. la préparation (mouvement préparatoire à partir d'une position de repos),
2. le noyau (postures, trajectoire gestuelle),
3. la rétraction (mouvement de retour à une position de repos).

Les propriétés de ce schéma sont universelles et peuvent être utilisées pour décrire la plupart des gestes de la main (statiques, dynamiques localisés et dynamiques en déplacement), excepté les gestes de battement.

[6]Certains gestes se comportent même comme des sinusoïdes dans le temps [Cohen et al., 1996]. En effet, de nombreux gestes sont des mouvements oscillatoires que les humains utilisent dans des situations critiques ou potentiellement dangereuses.

Il est nécessaire de différencier les gestes statiques, les gestes dynamiques localisés et les gestes dynamiques non-localisés. Les gestes statiques de la main sont caractérisés par la posture de celle-ci qui est déterminée par une configuration particulière des doigts et de la paume. Les gestes dynamiques sont caractérisés par l'enchaînement de postures ou par l'allure générale du mouvement.

Pour les gestes statiques, le noyau du schéma de Kendon est formé par une posture de la main. La phase de préparation a permis d'amener cette dernière dans un espace de travail et la phase de rétractation permettra alors à la main de se relâcher et de quitter l'espace de travail. Il en est de même pour les gestes dynamiques localisés, à ceci près, que le noyau est formé par une séquence de postures de la main. Pour les gestes dynamiques non-localisés, le noyau est formé par la trajectoire accomplie pendant le geste et éventuellement par la séquence de postures exécutées durant le mouvement.

Ainsi, des problèmes se posent tels que la segmentation temporelle des gestes par rapport aux mouvements non-intentionnels ou la détermination de l'intervalle d'exécution d'un geste. En effet, il faut pouvoir distinguer les gestes dynamiques isolés, les gestes dynamiques connectés et les gestes dynamiques enchaînés [Braffort, 1996]. Les gestes isolés sont les plus simples à reconnaître car ils sont segmentés naturellement par des vides gestuels. Les gestes connectés sont des gestes exécutés les uns à la suite des autres sans recouvrement. Les gestes enchaînés sont tels que la fin d'un geste influence le début d'un nouveau et inversement. Il est tout à fait possible de comparer ce phénomène avec celui de la coarticulation pour la parole. En effet, lorsqu'un locuteur parle, les phonèmes (éléments sonores du langage) peuvent être modifiés par le prochain phonème qui sera prononcé.

Pour simplifier certains de ces problèmes, un schéma de gestes comprenant des règles pour la segmentation a été établi [Quek, 1994].

- Les gestes sont des mouvements en trois phases. A partir d'une position de repos, ils commencent par un déplacement lent. Puis, ils continuent en une phase augmentant en vitesse. Enfin, ils se terminent par un retour rapide à une position de repos. On retrouve le schéma de Kendon.
- Les mains ont une configuration particulière durant l'exécution du mouvement.
- Les mouvements lents entre des positions de repos ne sont pas des gestes.
- Les gestes des mains sont restreints à un certain espace de travail.
- Les gestes statiques de la main ont besoin d'une période de temps finie pour être reconnus. Les mouvements répétitifs peuvent être des gestes.

La résolution du problème de la segmentation pour les gestes dynamiques peut se faire par différentes techniques de détection de caractéristiques de la trajectoire comme les pauses [Cagin, 1993] ou les points d'inflexions et de rebroussement [Sagawa et al., 1992].

2.2.1 La main

La main possède des facultés d'adaptation et une grande dextérité [Sturman, 1992]. Son caractère naturel la rend avantageuse dans des situations d'urgences qui réclament une réaction instinctive. Ses facultés d'adaptation font référence à sa facilité à pouvoir passer doucement et rapidement d'une fonction à une autre. Sa dextérité lui permet de réaliser une tâche complexe de façon optimale. La main est un objet hautement déformable qui doit ses qualités à l'association des muscles (Figure 2.11) et des nombreuses articulations reliant les os de son squelette (Figure 2.12).

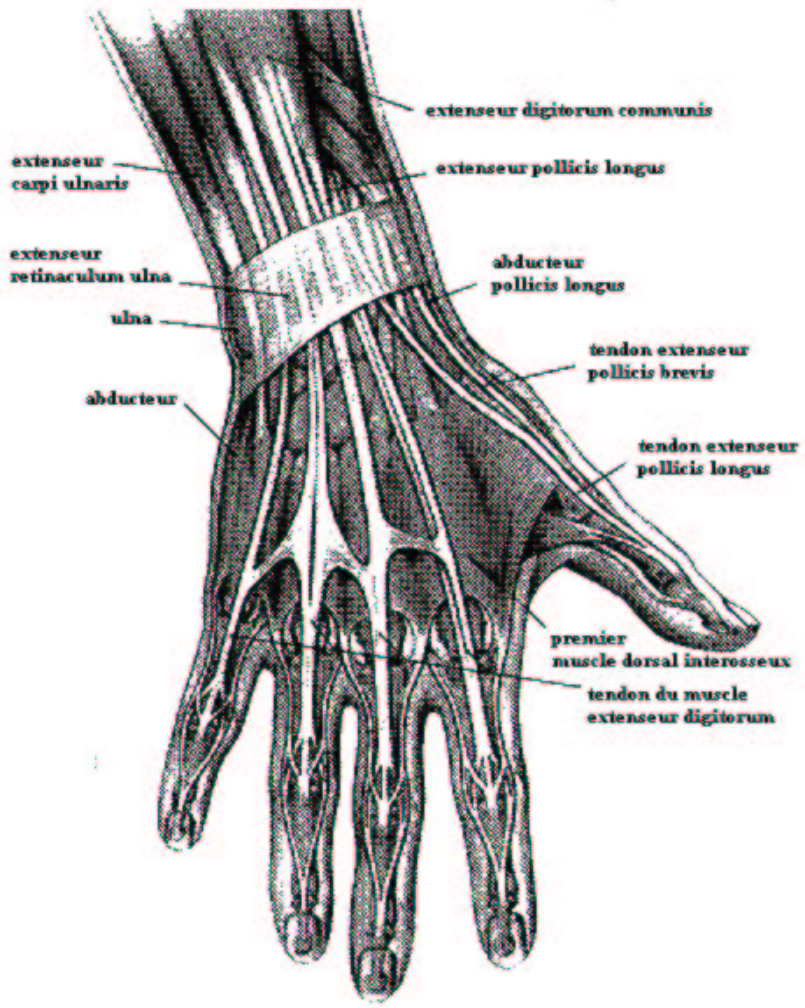


Figure 2.11: Les muscles dorsaux de la main

La main possède, selon les auteurs, entre 27 et 29 degrés de liberté [Rehg, 1995] [Sturman, 1992].

Les nombreuses interconnexions et interactions des muscles et des tendons donnent une grande complexité au mouvement.

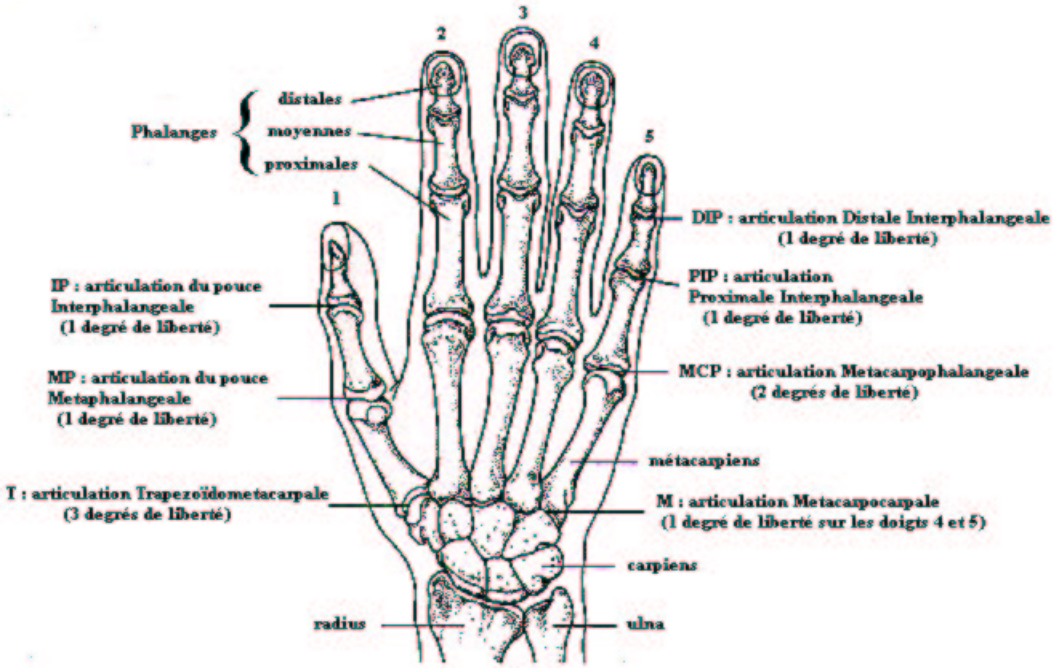


Figure 2.12: Les articulations de la main [Napier, 1980]

La plus grande partie de la masse musculaire est liée à l'avant-bras avec de longs tendons qui transmettent la force aux doigts. La main est ainsi un excellent compromis de légèreté, de flexibilité et de force.

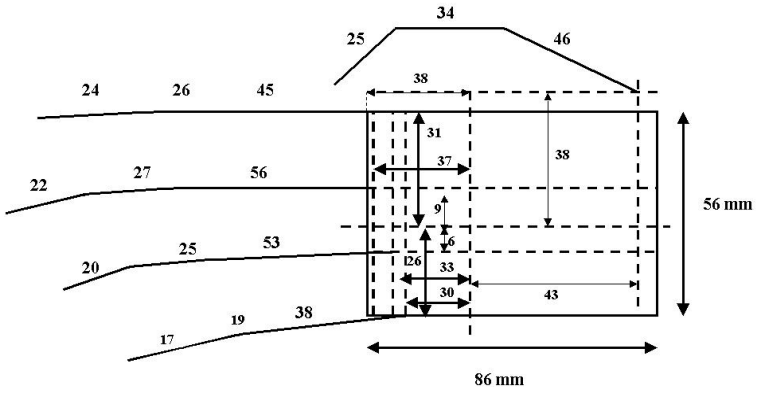


Figure 2.13: Modèle anthropométrique de la main

On connaît la longueur des segments (Figure 2.13), ainsi que les angles associés aux articulations qui respectent certaines constantes et certaines contraintes dynamiques [Rehg, 1995].

Les doigts de la main exécutent des mouvements de flexion, d'extension, d'abduction et d'adduction (Figure 2.14). Le pouce est désolidarisé de la paume. En effet, les trois degrés de liberté de l'articulation trapézoïdométacarpale[7] permettent au pouce d'effectuer une rotation longitudinale en opposition avec les autres doigts.

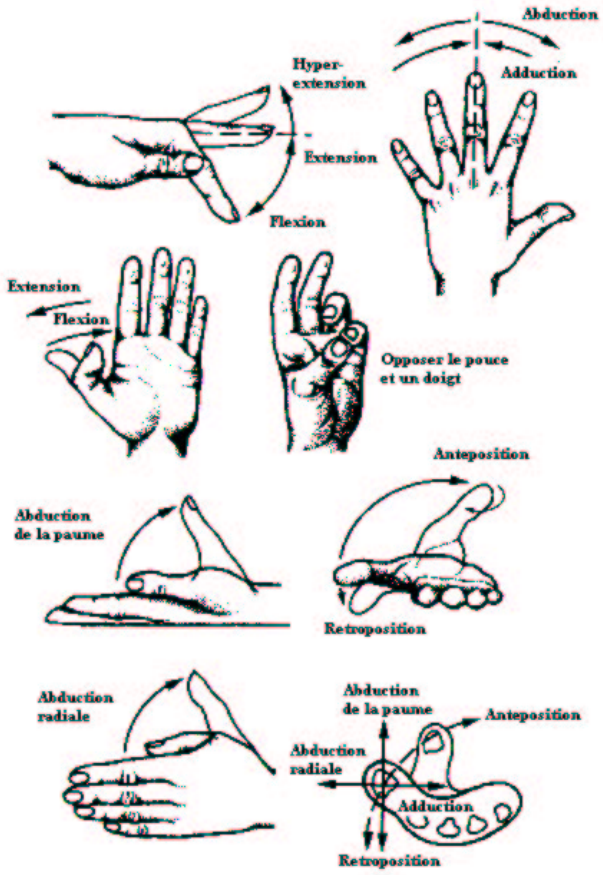


Figure 2.14: Les mouvements des doigts de la main

Les gestes ne se limitent pas à ceux de la main. En effet, les gestes corporels ont aussi une place importante dans le canal gestuel. Ils complètent les gestes de la main et ils donnent souvent des informations sur l'état émotionnel et cognitif du sujet, à travers les émotions visibles sur le visage ou l'activité relative au changement de posture.

2.2.2 Le corps

Les gestes des mains ont tendance à être centrés par rapport au corps [Sturman, 1992]. Les représentations du corps les plus utilisées sont des modèles en trois

[7] L'articulation trapézoïdométacarpale T est située à la base du pouce (Figure 2.12).

dimensions (Figure 2.15), le plus souvent pour des applications de synthèse d'images et d'animations par ordinateur [Magnenat-Thalmann et Thalman, 1990].

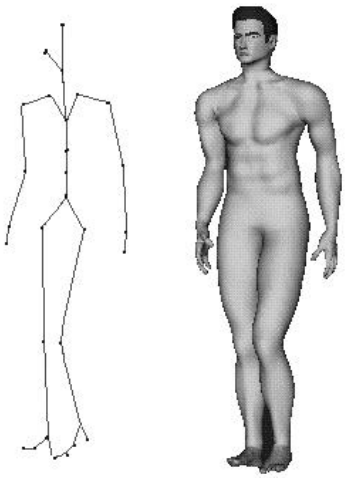


Figure 2.15: Un modèle du corps en trois dimensions

Cependant, on peut se satisfaire de modèles plus simples en deux dimensions. On peut ainsi schématiser le corps par la longueur des segments de son squelette (Figure 2.16) qui sont exprimés en fonction de la hauteur totale du corps. Les proportions sont issues d'études anthropométriques [Winter, 1979].

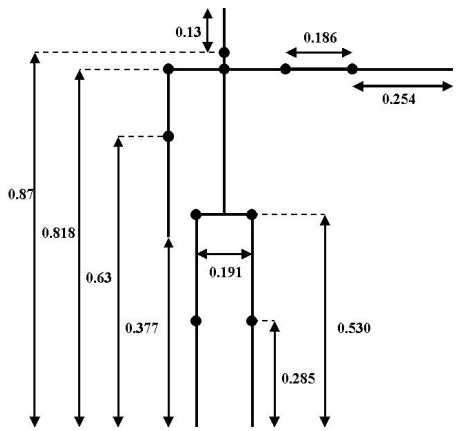


Figure 2.16: Le modèle anthropométrique du corps : les longueurs des segments sont exprimés en proportion par rapport à la hauteur totale du corps

L'utilisation d'un tel modèle pourrait permettre d'analyser certaines postures corporelles (assis, debout, baissé), mais il pourrait aussi nous aider à rechercher les mains et à interpréter leurs gestes.

2.2.3 Discrétisation de l'espace corporel

Certains auteurs proposent de découper le voisinage du corps suivant des zones où les gestes évoluent [McNeill, 1992], afin de réaliser une discrétisation de l'espace de représentation des paramètres gestuels. En effet, tous les gestes ne sont pas effectués aux même endroits et de la même manière dans l'espace corporel. On peut construire cet espace en prenant le centre du corps comme centre du repère, l'axe de symétrie du corps comme axe principal et les distances maximales que les mains peuvent atteindre comme bornes.

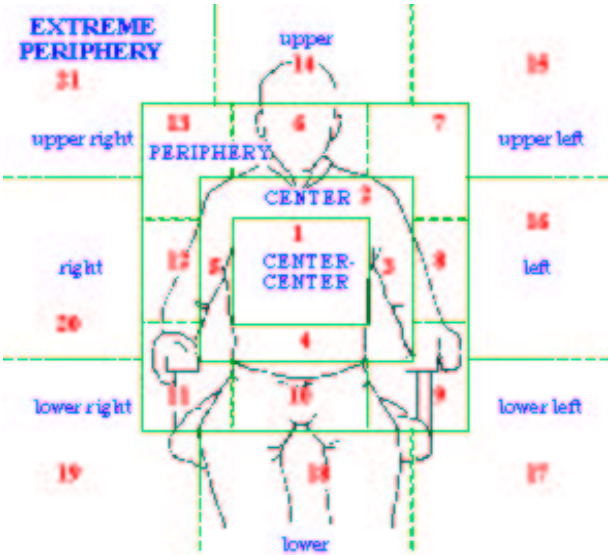


Figure 2.17: Discrétisation de l'espace pour l'emplacement des mains

L'espace corporel est découpé en zones rectangulaires numérotées et concentriques autour du buste (Figure 2.17). Le "center-center", que nous traduirons par **hyper-centre**, est positionné sur la poitrine. Le **centre**, légèrement plus grand que l'**hyper-centre**, inclut les épaules. La périphérie englobe le **centre** pour contenir le visage et les mains au repos. L'extrême périphérie étend la périphérie à la distance maximale que les mains peuvent atteindre.

Cette discrétisation de l'espace est une première étape vers la segmentation des gestes statiques. En effet, afin de pouvoir détecter efficacement l'intention de l'utilisateur de s'adresser au système, on supposera que toute commande reconnue ne doit résulter que d'un effort intentionnel de l'utilisateur [Baudel et Braffort, 1993]. Lors d'un tel effort, les mains se déplacent dans une zone particulière de l'espace corporel, nommée fenêtre active [Marcel et al., 1999]. De plus, les postures reconnues dans la fenêtre active doivent former un vocabulaire bien précis de commandes intuitives qui ont peu de chance d'être réalisées par inadvertance, et qui réclament un effort bref et faible de l'utilisateur pour ne pas le fatiguer. En effet, un geste de la main fait intervenir les muscles du bras, et ce dernier ne dispose pas toujours de points d'appuis.

Ainsi, en se restreignant à une fenêtre active et en définissant un vocabulaire,

on limite le syndrome d'immersion, c'est-à-dire l'impossibilité pour l'utilisateur d'interagir dans le monde réel lorsque l'ordinateur capte et interprète tous ses mouvements. Pour illustrer ce syndrome, imaginons un utilisateur interagissant avec un ordinateur doté d'une fonction d'interprétation des gestes, soudain il s'arrête et effectue un mouvement non-intentionnel car une personne est entrée dans la pièce et lui a parlé. Hélas, le mouvement non-intentionnel est interprété par la machine et peut provoquer alors un incident.

2.3 Analyse des gestes

L'analyse des gestes doit prendre en compte la dimension spatiale et temporelle. Elle doit mettre en œuvre des techniques d'extraction de caractéristiques, pertinentes par rapport à la provenance des données et adaptées à la méthode de reconnaissance choisie. On peut distinguer deux types d'interfaces gestuelles : les interfaces instrumentales et les interfaces gestuelles pures [Baudel et Braffort, 1993]. Les interfaces instrumentales analysent la trace laissée par le geste à l'aide d'une souris ou d'une tablette et les interfaces gestuelles pures interprètent le geste au moyen d'un gant numérique ou d'une caméra.

Nous allons nous placer dans le cadre des interfaces gestuelles pures qui constituent pour nous un intérêt majeur. En effet, on considère que seuls les capteurs tels qu'un gant numérique ou une caméra nous permettent d'exploiter le canal gestuel pour réaliser des interfaces homme-machine évoluées.

2.3.1 Analyse basée sur les gants numériques

Des périphériques spéciaux à base de gants ont été développés pour analyser la configuration de la main. Une des applications est la manipulation et l'exploration des mondes virtuels. L'utilisateur est muni d'un casque, d'un gant numérique et se retrouve immergé dans un monde virtuel. Le gant permet au système de repérer la main dans l'espace et de la reconstruire dans la réalité virtuelle. Ainsi, le sujet peut interagir avec la scène virtuelle, le plus souvent en saisissant des objets, en appuyant sur des boutons ou en déplaçant des jauges. Les gants numériques utilisent des capteurs mécaniques ou optiques qui traduisent les mouvements de flexion et d'abduction des doigts en des signaux électriques. La position et l'orientation de la main sont données par des capteurs électro-magnétiques ou acoustiques. Il en existe de toutes les sortes :

- Digital Data Entry Glove [Grimes, 1987],
- VPL DataGlove [VPL Research, 1987],
- Exos Dexterous HandMaster (Little Inc, 1987),
- Power Glove (Mattel Inc, 1989),

- Virtex CyberGlove [Kramer et Leifer, 1989],
- Space Glove [Industries, 1991].

Le gant le plus utilisé est le DataGlove de VPL Research. Il comporte seize degrés de liberté (dix informations de flexion et six informations de position).

Cependant, la fréquence d'acquisition des gants numériques reste faible et produit des données relativement bruitées. De plus, l'utilisateur reste relié physiquement par des câbles à l'ordinateur, ce qui a pour effet de dénaturer les gestes qu'il exécute. Les gants numériques sont encombrants et complexes à connecter. C'est pourquoi, différentes techniques basées sur le traitement et l'analyse de l'image ont été introduites pour enrichir l'aspect naturel de l'interaction.

2.3.2 Analyse basée sur l'image

L'analyse de la posture du corps, ou des gestes de la main, basée sur l'image est la voie la plus naturelle pour la construction d'interfaces gestuelles homme-machine, mais elle est la plus difficile à concevoir. L'image est une modalité non intrusive pour la reconnaissance des gestes [Campbell et al., 1996]. Les nouvelles technologies permettent le traitement d'images en temps réel. Ainsi, l'interaction basée sur l'image devient possible. Pourtant, cette approche soulève de nombreux problèmes techniques : segmenter le corps ou la main du reste de la scène, analyser les postures et suivre l'objet segmenté dans une séquence d'images.

Segmentation de l'image

Pour simplifier la segmentation, certains systèmes utilisent des marqueurs passifs (points colorés), des marqueurs actifs (diodes lumineuses) ou la couleur de la peau. Lorsque le sujet est immobile, que la caméra est fixe et que le fond est stable et uniforme, on peut alors localiser les marqueurs dans l'image. De plus, la main possède une teinte particulière qui peut s'avérer discriminante. Ainsi, on utilise des techniques à base d'histogrammes de couleurs [Saxe et Foulds, 1996] ou des tables de correspondances chromatiques (HSI[8] ou RVB[9]) basées sur des pixels de teinte chair. Cependant, ces techniques de localisation sont limitées par la variabilité de l'empreinte de la couleur de la peau en fonction des conditions d'illumination.

Analyse de la posture

Une approche pour analyser les postures et les gestes, est d'effectuer une reconstruction en trois dimensions. Un modèle 3D est mis en correspondance avec une ou plusieurs images pour estimer les paramètres (orientations, angles des articulations). Cependant, le problème d'occultation limite cette technique lorsque

[8] teinte, saturation et intensité.
[9] rouge, vert et bleu.

l'on ne dispose que d'un seul capteur, ce qui peut être partiellement résolu par l'utilisation de plusieurs caméras. Les modèles en trois dimensions utilisés peuvent être volumétriques ou squelettiques.

Les modèles volumétriques sont utilisés pour décrire l'apparence visuelle des objets en animation par ordinateur, et ainsi, pour pouvoir effectuer une analyse par la synthèse du suivi et de la posture [Koch, 1993] [Wren et al., 1997]. Dans ces modèles, on utilise des structures géométriques, telles que des cylindres, des sphères ou des ellipses, pour représenter la forme de certaines parties du corps (doigts, bras, avant-bras, jambes) [Pentland et al., 1996] [Downton et Drouet, 1991] [Etoh et al., 1991] [Gravila et Davis, 1995] [O'Rourke et Badler, 1980]. L'estimation des paramètres de ces modèles est complexe. En effet, il faut pouvoir estimer les paramètres initiaux et les mettre à jour au cours du temps. Cette méthode est très coûteuse et difficilement implémentable en temps réel. De plus, la connaissance de la posture exacte de la main, par exemple, est inutile pour des gestes de communication [Quek, 1994], mais est bien adaptée pour des gestes de manipulation.

Les modèles squelettiques se basent sur les caractéristiques morphologiques et bio-mécaniques [Tubiana, 1981] [Thompson, 1981] pour représenter les segments et les angles articulatoires [Ahmad, 1994] [Rehg et Kanade, 1993] [Rehg et Kanade, 1995] [Vaillant et Darmon, 1995].

Une autre approche d'analyse est basée sur une mesure de similarité avec les éléments d'un ensemble de postures ou de gestes pré-établis. De nombreuses méthodes utilisent des modèles 2D déformables de la main ou du corps [Lanitis et al., 1995] [Kervrann et Heitz, 1995] [Cootes et al., 1995] [Gravila, 1996], dans lesquels on ajuste et on associe un motif avec un modèle existant à l'aide d'une fonction d'interpolation. Ces motifs sont généralement constitués d'une suite de points provenant de l'extraction des contours, de la courbure de l'extrémité des doigts [Cipolla et al., 1993] [Davis et Shah, 1994] ou du calcul des filtres de Gabor en des points particuliers de l'image [Triesch et Malsburg, 1996] [Triesch et Malsburg, 1998]. Les modèles déformables peuvent procurer des informations suffisantes pour les gestes de manipulation et de communication, mais ils nécessitent une grande résolution de l'image et ont un coût de calcul élevé. D'autres méthodes sont des techniques de corrélation d'une transformation de l'image telle qu'une décomposition en vecteurs propres et l'accumulation des informations spatio-temporelles. Ces techniques sont bien plus faciles à calculer, mais elles donnent des informations moins précises, ce qui les rend moins adaptées pour exploiter des gestes de manipulation.

Suivi

Nous avons précisé qu'un geste est un processus autant temporel que spatial. Ceci implique de pouvoir connaître à chaque instant, c'est-à-dire dans chaque image, la position de la main précédemment reconnue, afin d'établir sa trajectoire. Il existe différentes techniques pour prédire le prochain emplacement de la main ou pour estimer les paramètres d'un modèle 3D, en utilisant un modèle dynamique [Rehg et Kanade, 1993] [Kuch et Huang, 1995] ou des filtres de Kalman [Vaillant et Darmon,

1995] [Pentland et Liu, 1995] [Wren et Pentland, 1998] [Martin et al., 1998].

2.4 Reconnaissance des gestes

La reconnaissance des gestes consiste à réaliser la classification des données issues de la phase d'analyse. La reconnaissance des actions gestuelles (gestes dynamiques) nécessite une invariance dans le temps, alors que de son côté, la reconnaissance des postures doit être invariante à l'échelle et à l'orientation. La reconnaissance des gestes naturels est difficile car les gestes ont une très grande variabilité d'une personne à l'autre. De plus, la reconnaissance des gestes souffre de problèmes de coarticulation. Par exemple, le mouvement que l'auteur du geste envisage d'exécuter après un geste peut influencer la fin du geste courant. De même, lors de l'exécution d'un geste dynamique localisé (enchaînement de postures), une posture transitoire non-intentionnelle peut être reconnue, ce qui pourrait empêcher la reconnaissance.

2.4.1 Reconnaissance des gestes statiques

La reconnaissance des postures peut se baser sur des modèles d'action gestuelle (historique des mouvements dans une image [Essa et Pentland, 1995] [Bobick et Davis, 1996] formé par l'accumulation du mouvement), des contours [Deriche, 1987] [Shen et Castan, 1986] [Kass et al., 1997], des fonctions de taille [Uras et Verri, 1995] [Frosini, 1991], des moments mathématiques [Teague, 1980] (signatures [Bröckl-Fox, 1995], moments de Zernike [Hunter et al., 1995], filtres orientables [Freeman et Adelson, 1991]), des vecteurs propres [Moghaddam et Pentland, 1995] ou des coefficients de Gabor [Triesch et Malsburg, 1996]. Les moments de Zernike [Schlenzig et al., 1994b] ont des magnitudes invariantes à la rotation et les histogrammes d'orientation [Freeman et Roth, 1995] sont invariants aux conditions d'éclairage. Mais, ces techniques ont besoin d'une résolution de l'image assez grande pour être discriminantes.

La reconnaissance de posture peut se faire par un arbre de décision [Cagin, 1993] et par une recherche du maximum de vraisemblance [Rubine, 1991] où une mesure de distance est utilisée pour évaluer le taux de corrélation d'un vecteur de caractéristiques avec des exemples de postures [Darrell et Pentland, 1992]. La reconnaissance de posture peut également être réalisée par des réseaux neuromimétiques [Kramer et Leifer, 1989]. Cependant, cette technique est peu utilisée car le temps d'apprentissage est élevé. Pourtant, elle obtient de très bons résultats, et peut travailler sur des images de petite taille.

2.4.2 Reconnaissance des gestes dynamiques

Les caractéristiques utilisées pour reconnaître les gestes sont nombreuses. On utilise généralement la trace du mouvement, l'accélération, la vitesse cartésienne, la vitesse polaire ou angulaire [Campbell et al., 1996], la longueur de la boîte englobante, la longueur totale parcourue ou l'angle de départ [Rubine, 1991]. Ces caractéristiques,

plus ou moins invariantes, sont extraites dans une fenêtre de temps fixe ou variable pour déterminer le type de geste.

Déformation dynamique du temps (Dynamic Time Warping)

Cette méthode utilise des techniques de programmation dynamique pour aligner temporellement deux séquences gestuelles de durée différente [Campbell et al., 1996]. La programmation dynamique essaye de résoudre une tâche de correspondance en recherchant le coût minimal d'un chemin dans un graphe où les noeuds représentent généralement les états traversés par un geste [Wilson et Bobick, 1995].

Les réseaux de neurones

La reconnaissance de gestes peut se faire par réseaux de neurones artificiels. Lorsque les gestes sont de taille fixe (postures) ou que les caractéristiques gestuelles sont de longueur finie, il convient, le plus souvent, d'utiliser un perceptron multicouches [Marcel, 1999] [Harling, 1993]. D'autres auteurs préféreront utiliser des cartes topologiques de Kohonen [Boehm et al., 1994]. Cependant, les gestes ne sont pas toujours de longueur fixe, leur durée peut être très variable. On utilise alors des réseaux de neurones récurrents [Murakami et Taguchi, 1991] ou des TDNN (Time Delay Neural Networks) [Waibel et Vo, 1993].

Les Modèles de Markov Cachés ou Hidden Markov Models

Les modèles de Markov cachés (HMM) sont très utilisés en reconnaissance des gestes [Starner et Pentland, 1995] [Schlenzig et al., 1994a]. Ce sont des processus doublement stochastiques [Rabiner, 1989] constitués d'états cachés, de distributions de transition entre les états et de distributions d'émission pour émettre des observations. On attribue un modèle de HMM par geste à reconnaître, puis on cherche à déterminer les paramètres de ces modèles par apprentissage. L'apprentissage des HMM consiste à calculer les probabilités de transition et les probabilités d'émission par optimisation du maximum de vraisemblance à travers un ensemble d'exemples d'apprentissage. Pour cela, on utilise des procédures de maximisation de la vraisemblance comme l'algorithme de Baum-Welch [Baum, 1972]. Lors de la reconnaissance, on cherchera lequel des modèles de HMM a la plus grande probabilité d'avoir généré le geste observé.

Mais auparavant, il faut pouvoir modéliser les observations qui, en reconnaissance des gestes, sont les composants élémentaires d'un geste (postures, trajectoires) représentés par des vecteurs de caractéristiques. Pour modéliser les observations, on utilise des mélanges de gaussiennes [Moghaddam et Pentland, 1997], des réseaux de neurones ou des techniques de quantification de vecteurs (moments géométriques, moments de Zernike, valeurs propres).

2.4.3 Systèmes existants

Les systèmes à base de gants numériques

Une implémentation d'un monde virtuel combinant les gestes de la main, la parole, le son et des images stéréoscopiques existe [Codella et al., 1992]. Ce système simule une pièce contenant des objets non rigides qui peuvent être créés ou déplacés en temps réel par deux utilisateurs.

Glove-talk [Fels et Hinton, 1993], une interface comprenant la main d'un utilisateur et un synthétiseur vocal, utilise cinq réseaux de neurones artificiels pour reconnaître un vocabulaire de deux cent trois gestes de la main associés à des mots.

Le VirtualPanelArchitecture [Su et Furuta, 1993] combine un panneau de contrôle matériel et logiciel. Les utilisateurs peuvent respectivement tourner, bouger et pointer des boutons, des ascenseurs et des écrans virtuels.

Le système de pointage VirtualEnd-Effector [Wang et Cannon, 1993] est utilisé pour entraîner et diriger des robots grâce aux gestes de la main. Il utilise des réseaux de neurones basés sur une transformation du squelette.

Le système GIVEN [Figueiredo et al., 1993] (Gesture-based Interaction in Virtual Environments) introduit des techniques permettant à l'utilisateur de saisir et d'entourer des objets virtuels et procure ainsi une interaction plus précise. L'utilisation de capteurs tactiles a été proposée pour augmenter la sensibilité dans les espaces virtuels.

Un des concepts les plus récents est une alternative entre les systèmes multimédias et les systèmes de réalité virtuelle [Krueger, 1993]. Le système représente un environnement de travail virtuel contenant des objets et des outils de contrôle virtuels. Ces derniers sont manipulés à partir d'un support de travail réel par différents utilisateurs collaborant au même projet.

Dans le système CHARADE [Baudel et Beaudouin-Lafon, 1993], les gestes de la main sont utilisés pour contrôler la navigation dans un hypertexte. Il fonctionne en temps réel et reconnaît six gestes de commande. Tous ces gestes comprennent trois phases (posture de départ, phase dynamique, posture finale). La discrimination entre les différentes commandes est faite sur les deux premières phases. Cependant, le vocabulaire des postures est complexe à réaliser, ce qui ne facilite pas l'utilisation du système.

Les systèmes basés sur l'image

- Les systèmes utilisant des modèles 3D de la main :
  Une approche utilisée en reconnaissance des gestes de la main consiste à reconstruire un modèle tridimensionnel de la main humaine. Le modèle est mis en correspondance avec des images. Puis, les paramètres correspondant à l'orientation et aux angles des articulations sont estimés et classés.

  Il existe un système de suivi des membres humains [Downton et Drouet, 1991]

dans une séquence d'images, utilisant un modèle humain cylindrique articulé. Les images sont obtenues sur fond noir uniforme à partir d'une caméra monochrome. La correspondance du modèle avec l'image est basée sur une projection perspective du modèle cylindrique et sur la comparaison avec les contours obtenus dans l'image. Le système fournit alors une information cinématique des angles des articulations.

Un modèle cylindrique a également été réalisé au laboratoire ATR Research [Etoh et al., 1991]. Le système utilise des caméras stéréoscopiques et un algorithme qui partitionne un objet en un ensemble hiérarchique de cylindres pour modéliser la main humaine. Les points de contours extraits de l'image sont utilisés pour trouver l'axe et la dimension associés au cylindre.

DigitEyes, un système complet de reconnaissance des gestes de la main, utilise un modèle cinématique cylindrique 3D doté de 27 degrés de libertés [Rehg et Kanade, 1993]. Les extrémités des doigts sont extraites d'une image stéréoscopique sur fond restreint et sont utilisées comme caractéristiques pour effectuer la correspondance avec le modèle. Le suivi des caractéristiques et l'estimation des paramètres du modèle se font par minimisation de l'erreur résiduelle. Le système a été testé pour une application de souris graphique 3D avec une seule caméra et pour l'estimation des angles des articulations de la main avec des caméras stéréoscopiques.

Un système d'analyse des gestes de la main basé sur un modèle squelettique en trois dimensions a été développé par Lee et Kunii [Lee et Kunii, 1993]. Ce modèle est doté de 27 degrés de liberté et incorpore des contraintes issues de la cinématique de la main pour réduire l'espace de recherche des paramètres du modèle. Le système a été utilisé pour l'analyse de 16 symboles de la LSA[10]. Il produit de très faibles erreurs, mais le temps de calcul est très élevé.

De même, Kuch a formalisé un modèle de la main avec 26 degrés de liberté incluant six contraintes cinématiques de la main [Kuch et Huang, 1995]. Ce modèle peut suivre des gestes complexes de la main dans une grande séquence d'images provenant d'une seule caméra. Le système a été utilisé pour le suivi des gestes de la LSA et comme "Virtual Gun".

- Les systèmes utilisant des marqueurs ou des gants colorés :
  La main humaine possède une forme hautement non-convexe et essayer de détecter sa configuration à partir d'images est très difficile. Une des techniques est donc de placer des marqueurs colorés distincts sur la main et plus généralement sur les extrémités des doigts.

  Torige et Kono ont conçu un système qui indique la direction de la main en mouvement [Torige et Kono, 1992]. Ils utilisent des caméras stéréoscopiques, des gants noirs et des marqueurs colorés sur l'épaule, le coude, le poignet et l'extrémité des doigts. Ils calculent la position des doigts et les paramètres de mouvement pour contrôler un robot manipulateur.

[10] Langue des Signes Américaine.

Davis et Shah [Davis et Shah, 1993] se basent sur les marqueurs des extrémités des doigts. Le système calcule les trajectoires des marqueurs sur plusieurs images de fond uniformément noir, puis il les utilise pour déterminer le début et la fin d'un geste. Le système réalise une segmentation temporelle de sept gestes prédéfinis de la main à quatre images par seconde.

Maggioni [Maggioni, 1993] utilise les images d'une seule caméra et un gant dont certaines régions sont colorées différemment. Le système calcule plusieurs paramètres géométriques et les utilise pour estimer la position et l'orientation de la main.

Cipolla, Okamoto et Kuno [Cipolla et al., 1993] utilisent un gant, dont les extrémités des doigts sont marquées, pour déterminer le mouvement de rotation et de translation de la main afin de changer la position et l'orientation d'un objet virtuel.

- Les systèmes utilisant les propriétés de l'image :
  Ces systèmes se basent sur l'extraction de certaines caractéristiques associées à des images de postures de mains. Les techniques d'analyse peuvent travailler simplement sur des moments géométriques de l'image (moments de Hu [Ming-Kuel, 1962], moments de Fourier-Mellin [Arsenaud et Sheng, 1986], moments de Zernike [Teague, 1980] Annexe F) ou avec des réseaux de neurones artificiels. Cependant, ces méthodes n'ont pas pour but d'estimer les paramètres exacts de la main comme les angles des articulations, leur objectif est d'effectuer soit un suivi de la main, soit une classification de la posture.

  Darrell et Pentland ont développé un système fonctionnant à dix images par seconde en utilisant des ensembles de vues [Darrell et Pentland, 1993]. Chaque geste de la main est représenté par son propre ensemble de différentes vues. Ces dernières sont ensuite mises en correspondance avec les gestes d'une séquence d'images en utilisant des techniques de corrélation temporelle (Dynamic Time Warping).

  Segen [Segen, 1993] utilise des techniques d'extraction de contours à partir de simples silhouettes pour discerner en temps réel dix postures distinctes.

  Ahmad et Tresp [Ahmad et Tresp, 1993] proposent un réseau de neurones pour classifier la posture d'une main. Celle-ci est décrite par les coordonnées polaires des extrémités des doigts et le centre de la main dans l'image. Ce réseau est conçu de telle sorte qu'il effectue une classification avec des caractéristiques d'entrées manquantes ou incertaines. Dans le cas où toutes les caractéristiques sont présentes, il obtient un taux d'erreur de 5%. Ahmad a également développé un système temps réel de suivi tridimensionnel de main en environnement complexe [Ahmad, 1994]. Il se base sur une segmentation d'histogramme et sur trois moments géométriques pour extraire et localiser dans l'image les extrémités des doigts et les angles des articulations.

  Starner et Pentland [Starner et Pentland, 1995] utilisent les paramètres géométriques d'une main uniformément colorée. Puis, un modèle de Markov caché à

cinq états effectue la classification de certains gestes de la Langue des Signes Américaine.

Schlenzig, Hunter et Jain utilisent des moments de Zernike comme caractéristiques [Schlenzig et al., 1994b]. Ceux-ci sont calculés à partir d'images de postures de la main sur un fond uniforme. Un modèle de HMM effectue la reconnaissance de six gestes de la main à la cadence de $\frac{1}{2}$Hz.

Kjeldsen [Kjeldsen, 1994] a conçu un système pour le contrôle d'une interface d'ordinateur multi-fenêtrée à l'aide des gestes de la main. Il utilise une segmentation d'histogramme, un réseau de neurones pour la différentiation des postures et un ensemble de règles pour la reconnaissance des gestes. Freeman et Roth [Freeman et Weissman, 1995] ont proposé un système simple et rapide de reconnaissance des postures à partir d'histogrammes d'orientations locales. Le système est robuste aux variations locales d'illumination, mais nécessite un fond noir uniforme.

2.5 Conclusion du chapitre

L'étude des gestes est un domaine assez ancien, mais son analyse au travers d'outils modernes, comme les gants numériques ou les caméras, est plus récente. La nomenclature des gestes est bien établi, et les techniques d'analyses et de reconnaissance sont nombreuses.

Nous avons choisi d'analyser les gestes de la main en nous basant sur les images provenant d'une caméra vidéo. En effet, l'image est une modalité non-intrusive pour la reconnaissance des gestes. En utilisant l'image, nous devons être capables de reconnaître les gestes de la main aussi bien statiques (postures de la main), que dynamiques (enchaînements de postures, trajectoires dans un espace corporel).

Le chapitre suivant, introduit des techniques statistiques d'apprentissage par ordinateur pour effectuer de la reconnaissance de formes dans les images et de la reconnaissance de séquences de données temporelles.

CHAPITRE 3

Les réseaux de neurones et les modèles de Markov cachés

Dans ce chapitre, nous abordons l'apprentissage statistique et deux modèles de machines à apprendre : les réseaux de neurones artificiels et les modèles de Markov cachés.

Les réseaux de neurones artificiels, d'une part, sont inspirés du neurone biologique et de l'organisation du cerveau humain. Nous présentons, tout d'abord, les fondements biologiques des réseaux de neurones artificiels que nous approfondissons ensuite d'un point de vue plus formel. Puis, nous introduisons la théorie de l'apprentissage et la notion de généralisation pour les réseaux de neurones artificiels en particulier et pour les machines à apprendre en général. Les réseaux de neurones artificiels sont principalement utilisés pour la reconnaissance de données statiques et ne sont pas naturellement adaptés à la reconnaissance de données séquentielles.

Les modèles de Markov cachés, d'autre part, ne sont pas inspirés d'un modèle biologiquement plausible. Ils reposent sur une chaîne d'états discrets et sont particulièrement bien adaptés à la modélisation des processus dynamiques. Nous présentons le modèle formel des modèles de Markov cachés, puis nous détaillons l'apprentissage statistique du point de vue de la maximisation de la vraisemblance.

Ce chapitre s'achève sur la présentation de modèles hybrides à base de réseaux de neurones artificiels et de modèles de Markov cachés.

3.1 Les réseaux de neurones

"Les capacités de traitement de l'information manifestées par les êtres vivants, même les plus simples, sont actuellement hors de portée des systèmes informatiques traditionnels : un ordinateur ne sait pas voir, entendre ou reconnaître des odeurs. Contrôler le vol d'un avion par ordinateur requiert des développements de programmes très complexes et très coûteux. Une mouche en fait autant et contrôle son vol grâce à un volume cérébral microscopique" (Extrait de [Blayo et Verleysen, 1996]).

Dans la première partie de ce chapitre nous introduisons les fondements nécessaires pour aborder les réseaux de neurones artificiels : les fondements biologiques, le neurone formel, les architectures de réseaux et les mécanismes d'apprentissage.

3.1.1 Les fondements biologiques

De tous temps, les hommes ont été fascinés par leur corps et leur anatomie. Dans le but de prendre réellement conscience d'eux-mêmes ou de chercher une réponse aux questions fondamentales : qui sommes nous ? d'où venons nous ? sommes nous les seuls à penser et à nous émouvoir ? Certains rationalistes ont tenté d'établir la résidence de l'âme dans le corps. Déjà, les chirurgiens de l'Ancien Empire égyptien constataient des relations entre les lésions cérébrales et les troubles du comportement. Cependant, les égyptiens ne semblaient pas avoir réellement compris les implications de ces observations. Pour les Hébreux, Homère ou même Aristote, c'est le coeur qui est la source de l'intelligence et des sentiments, pas le cerveau. Au II^{ème} siècle, les expériences de Galien démontrent que le cerveau joue bien un rôle central dans la commande du corps et dans l'activité mentale. Galien évoque, avec la notion de pneuma[1], le concept d'"organe de l'âme". Avec la renaissance, les chercheurs et les philosophes construisent des descriptions de plus en plus poussées de la morphologie cérébrale dont on reconnaît la complexité. Pourtant, la doctrine officielle est l'immatérialité, l'immortalité et l'unicité de l'âme. Même Descartes qui compare le corps humain à une machine, ne le contredit pas. Au XVII^{ème} siècle, la thèse de l'immortalité de l'âme disparaît progressivement des ouvrages consacrés aux sciences du cerveau.

"Il aura fallu presque trois millénaires pour retrouver la pensée des atomistes[2] grecs dans sa simplicité originelle et pour que celle-ci s'exprime en toute liberté" [Changeux, 1983]. Les premières descriptions modernes du cerveau apparaissent. On compare ce dernier avec l'ordinateur (machine cybernétique) pour introduire la notion de codage interne de l'information. On sait que l'encéphale de l'homme contient, dans l'organisation anatomique de son cortex, des représentations du monde qui l'entoure (images mentales). Il est aussi capable de construire de nouvelles représentations et de les utiliser dans ses calculs.

De récents progrès technologiques ont permis d'établir des cartes d'activités cérébrales en utilisant des caméras à positrons et des traceurs radioactifs. Néanmoins, même si les chercheurs ont pu déterminer le rôle de certaines zones corticales, et même s'ils ont pu décrire le fonctionnement exact du neurone biologique, ils n'ont pas encore totalement établi le lien entre les deux.

[1] Selon Galien, le pneuma circule dans les nerfs et met ainsi en relation cerveau, organes des sens et organes moteurs.

[2] Un atomiste est un partisan de l'atomisme : philosophie matérialiste qui considère l'univers comme formé d'un nombre fini d'atomes associés en combinaisons fortuites et purement mécaniques.

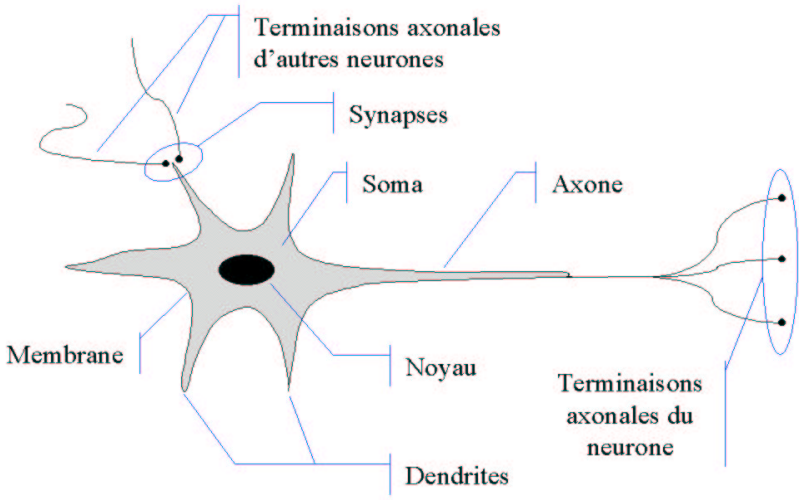


Figure 3.1: Représentation d'un neurone biologique.

Le système nerveux repose sur un nombre gigantesque d'unités élémentaires massivement interconnectées : les neurones (Figure 3.1). Le cerveau humain en possède de 10^{10} à 10^{12} et chacun d'entre eux est connecté en moyenne à 10^3 autres neurones. Le neurone est une cellule vivante. Il en existe de différentes formes (pyramidales, sphériques ou étoilées), mais tous ont une structure commune.

- la membrane : elle sépare l'intérieur du neurone (cytoplasme) de l'extérieur.
- le soma : c'est le corps du neurone et comme toutes les autres cellules du corps humain, sauf les hématies (globules rouges), il contient un noyau.
- le noyau : il détient l'information génétique qui, d'une certaine façon, contient le plan de construction complet de l'organisme. Et pourtant, le code génétique ne détermine pas précisément toutes les interconnexions entre les neurones, mais seulement l'organisation générale du cerveau. En effet, les quelques 100 000 gènes, approximativement, présents chez l'homme ne permettent pas de coder le nombre astronomique de synapses.
- les dendrites : ce sont des prolongements de la membrane du neurone qui recoivent des signaux d'autres neurones. Les dendrites mesurent de $0,2$ à 10 microns de diamètre et une dizaine de microns de longueur. Il n'y a pas de contact direct entre les membranes des neurones connectés : cette liaison est assurée par une synapse.
- l'axone : c'est la fibre nerveuse principale, le seul prolongement de la membrane qui se sépare en une chevelure de filaments à destination des dendrites d'autres neurones pour propager le signal.
- les terminaisons axonales : ce sont les extrémités de l'axone en contact avec les dendrites, via la synapse.

- les synapses : elles assurent le passage du signal des terminaisons axonales vers les dendrites, via généralement un processus électrochimique.

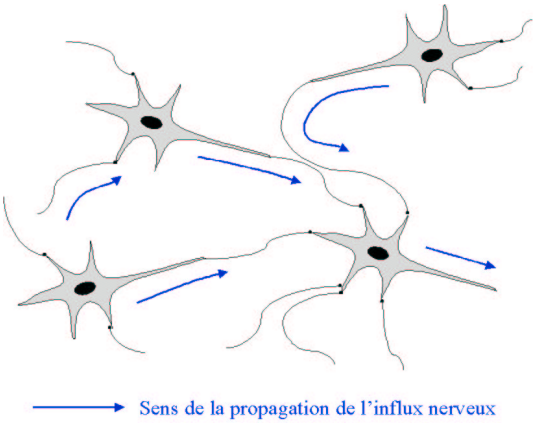


Figure 3.2: Représentation d'un réseau de neurones biologique.

Le neurone effectue une intégration spatiale et temporelle des signaux électriques en provenance d'autres neurones (Figure 3.2) via les dendrites. Les synapses jouent le rôle de modulateur (excitateur ou inhibiteur) entre une extrémité de l'axone et une dendrite.

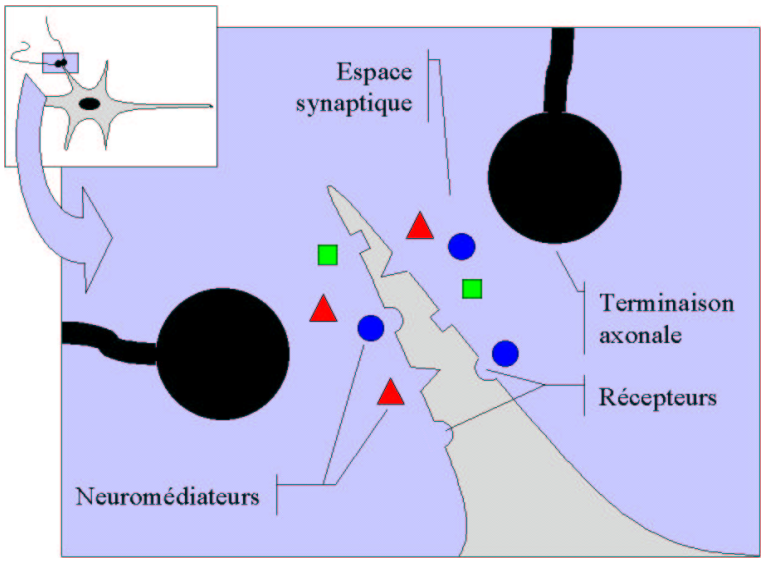


Figure 3.3: Représentation de la synapse.

A ce stade, la propagation du signal nerveux s'effectue grâce à un mécanisme électrochimique. Lorsque l'influx nerveux atteint les terminaisons synaptiques de l'axone, des molécules (neuromédiateurs) se diffusent dans l'espace synaptique pour se fixer sur des récepteurs (Figure 3.3). Un déséquilibre ionique se produit entre l'intérieur et l'extérieur de la cellule et un potentiel électrique se forme dans la

membrane du neurone. Si le potentiel électrique du neurone est assez élevé alors il se décharge le long de la fibre nerveuse principale : l'axone. Le signal propagé dans l'axone est une série d'impulsions dont la fréquence est d'environ 100 Hz et dont la vitesse de propagation est d'environ 80 m/s. Le lecteur intéressé par l'anatomie neuro-biologique du cerveau, son fonctionnement ou son histoire, trouvera des explications passionnantes et très détaillées dans [Changeux, 1983].

L'étude du cerveau a énormément progressé, elle a dépassé la scolastique[3] médiévale et l'obscurantisme[4] des siècles qui ont suivi. Cependant, même si la science nous a appris beaucoup sur le cerveau, elle nous a également appris à rester modeste devant l'extrême complexité de cet organe qui échappe encore pour l'essentiel, à notre compréhension.

Nous sommes encore très loin de pouvoir concevoir des cerveaux artificiels, et même si nous en étions capables, il faudrait concevoir des algorithmes également très complexes pour manipuler les flux d'informations. C'est pourquoi, nous ne parlerons plus de neurones biologiques. Lorsque nous parlerons de neurone ou de réseau de neurones, il ne s'agira que de leurs versions artificielles, bien plus simples.

3.1.2 Un peu d'histoire

En 1943, McCulloch et Pitts [McCulloch et Pitts, 1943] proposent le modèle du neurone formel inspiré des observations neurophysiologiques et anatomiques du neurone biologique. Les neurones constituent des unités simples (automates linéaires à seuils) mais qui assemblées sont capables de résoudre des fonctions logiques. En 1949, Donald Hebb propose une règle d'apprentissage pour les réseaux de neurones artificiels [Hebb, 1949]. En 1962, Rosenblatt avec le **perceptron** [Rosenblatt, 1962] et Widrow avec l'**adaline** [Widrow et Hoff, 1960] proposent un modèle de réseau à une couche et une méthode pour modifier les connexions. Rosenblatt démontre la convergence d'un algorithme itératif d'adaptation des poids pour un **perceptron** monocouche pour apprendre certaines fonctions logiques. En 1969, Minsky and Papert mettent en évidence les limitations du **perceptron** [Minsky et Papert, 1969], ils apportent un éclairage théorique sur certains problèmes impossibles à résoudre avec ce modèle, comme le OU-exclusif (XOR).

Durant 16 ans, les chercheurs vont se détourner de l'approche connexionniste pour lui privilégier une approche symbolique. Ainsi, l'Intelligence Artificielle (IA) symbolique a connu ses heures de gloire et a permis de grandes avancées comme celles des systèmes experts. Cependant, les chercheurs se sont vite rendu compte des limites de l'IA symbolique pour modéliser le fonctionnement du cerveau, qui n'est manifestement pas basé sur une logique symbolique. Bien heureusement, certains chercheurs sont restés fidèles à l'approche connexionniste et ont poursuivi leurs recherches, notamment dans le domaine des mémoires associatives.

[3]Enseignement philosophique et théologique propre au moyen âge, fondé sur la tradition aristotélicienne interprétée par les théologiens.

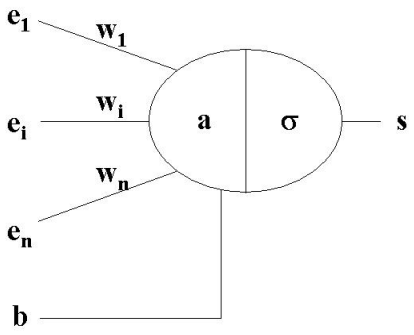
[4]Attitude d'opposition à l'instruction, à la raison et au progrès.

En 1982, Hopfield présente un modèle de réseau de neurones entièrement connecté et doté d'un formalisme plus riche [Hopfield, 1982]. Hopfield introduit la notion de fonction d'énergie et associe la mémoire à des attracteurs dans les systèmes dynamiques. Par ailleurs, "la machine de Boltzmann" [Ackley et al., 1985] est le premier modèle à lever les limitations du **perceptron** par la méthode du recuit simulé. Cependant, l'avancée la plus marquante est l'algorithme de rétropropagation découvert tout d'abord par Werbos en 1974 [Werbos, 1974], puis parallèlement par Rumelhart en 1986 [Rumelhart et al., 1986], Parker en 1985 [Parker, 1985] et Le Cun en 1985 [LeCun, 1985]. Cette approche, bien que non biologiquement plausible, lève la limitation du **perceptron** de Rosenblatt. Cette méthode est destinée à des modèles multicouches statiques (MLP) (sans récurrences).

Depuis, l'algorithme de rétropropagation a été étendu à des réseaux de neurones admettant des connexions récurrentes [DeVries et Principe, 1992] [Frasconi et al., 1992] [Leighton et Conrad, 1991] [Sastry et al., 1994] [Tsoi et Back, 1994] [Wan, 1993] [Williams et Zipser, 1989]. Les algorithmes de rétropropagation dynamique permettent de modéliser les processus dynamiques tels que les séries temporelles. Les algorithmes de rétropropagation reposent sur le principe de minimisation d'une fonction de l'erreur commise en sortie du réseau et effectuent un calcul et une descente du gradient pour trouver le minimum de cette fonction coût. La preuve de la convergence des algorithmes de rétropropagation dynamique vers un minimum global a été établi par [Srinivasan et al., 1994]. D'un autre côté, les limitations théoriques de la descente du gradient ont été démontrées par [Bengio et Simard, 1994].

3.1.3 Le neurone formel

Le neurone formel de McCulloch et Pitts [McCulloch et Pitts, 1943] est un opérateur binaire effectuant la somme pondérée des valeurs d'entrées par des coefficients synaptiques à valeurs réelles. Ces coefficients peuvent être positifs ou négatifs, on parlera alors respectivement de poids ou de synapses excitatrices ou inhibitrices.



$$a = \sum_{i=1}^{n} w_i e_i + b \qquad (3.1)$$

$$s = \sigma(a) \qquad (3.2)$$

Figure 3.4: Le neurone formel de Mc Culloch et Pitts.

- **e** est le vecteur d'entrée du neurone, il représente soit l'ensemble des signaux en provenance de l'environnement, soit les sorties d'autres neurones.

- **w** est un vecteur dont le rôle est de simuler les pondérations synaptiques à l'aide de poids associés respectivement à chaque entrée du neurone.

Le neurone formel est un séparateur linéaire (Figure 3.4, Equation 3.1) qui réalise une partition de l'espace des entrées en deux sous-espaces séparés par une droite (Figure 3.5). Cette partition est telle que les réponses données par le neurone, pour deux vecteurs d'entrée appartenant respectivement à ces deux sous-espaces, soient différentes.

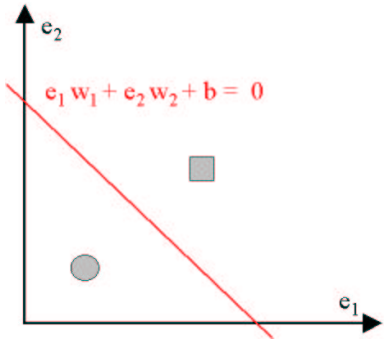


Figure 3.5: Le neurone formel est un séparateur linéaire.

- *b* est le biais du neurone. Sans un biais, les droites séparatrices passeraient toutes par 0 (Figure 3.6), ce qui pourrait rendre impossible la séparation de deux classes distinctes. Pour éviter cela, on ajoute un terme pour décaler la séparatrice et ainsi pouvoir effectivement réaliser la séparation des deux classes (Figure 3.7). Le plus souvent, on considère le biais comme le poids associé à une entrée de valeur constante (+1 ou −1).

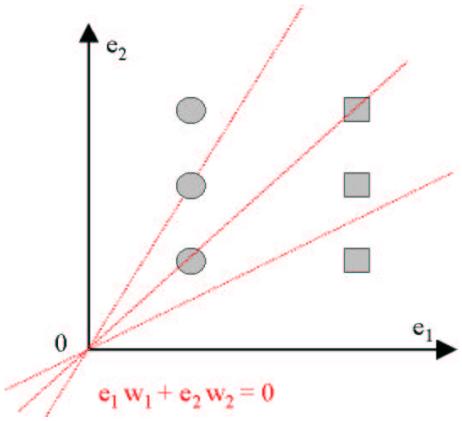


Figure 3.6: La séparation linéaire des deux classes est impossible sans biais.

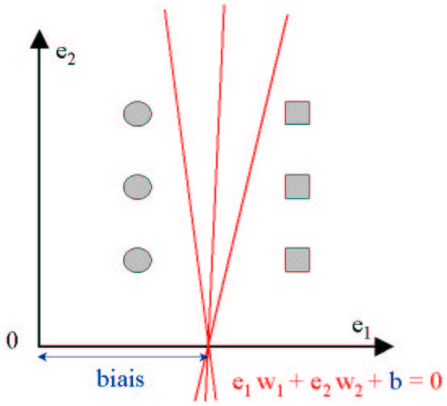


Figure 3.7: La séparation linéaire des deux classes devient possible avec un biais.

- *a* est une variable qui représente le potentiel du neurone. Cet état interne du neurone (activité) est défini par une somme des entrées pondérées par le vecteur de poids (Equation 3.1).

- *s* est une variable qui représente la sortie du neurone envoyée vers d'autres unités. Elle est calculée en appliquant une fonction mathématique à l'activité du neurone (Equation 3.2).

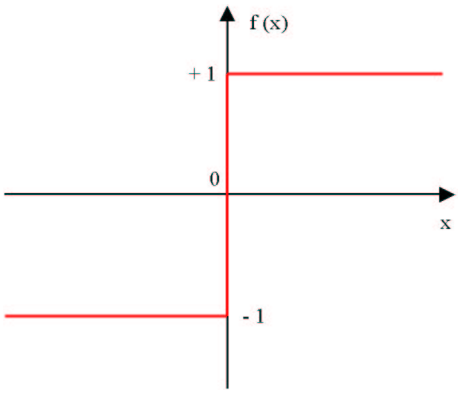


Figure 3.8: La fonction signe.

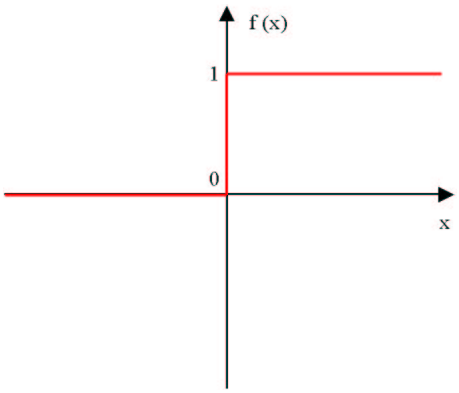


Figure 3.9: La fonction Heaviside.

- σ est la fonction d'activation (ou de transfert) qui est appliquée à l'activité. Il peut s'agir par exemple de la fonction signe (Equation 3.3) ou de la fonction de Heaviside (Equation 3.4).

$$\sigma(a) = \begin{cases} +1 & : \quad a \geq \theta \\ -1 & : \quad a < \theta \end{cases} \qquad (3.3)$$

$$\sigma(a) = \begin{cases} +1 & : \quad a \geq \theta \\ 0 & : \quad a < \theta \end{cases} \qquad (3.4)$$

Pour résumer, le neurone formel effectue une somme des entrées e_i pondérées par des poids w_i. Si cette somme pondérée dépasse un certain seuil θ alors le neurone est actif, sinon le neurone est inactif.

Evidement, il s'agit là d'une simplification grossière du neurone biologique. La modulation synaptique est représentée par un produit et l'intégration du signal est ramenée à une somme et à une fonction mathématique σ. De plus, le neurone formel accomplit une transformation statique, ne représentant que les propriétés spatiales du neurone biologique en ne prenant pas en compte l'aspect temporel du traitement du signal. Ce modèle est, en apparence, surprenant de simplicité. Mais nous allons voir que sa généralisation va permettre des architectures de réseaux de neurones complexes et de nouveaux algorithmes pour travailler sur ces modèles.

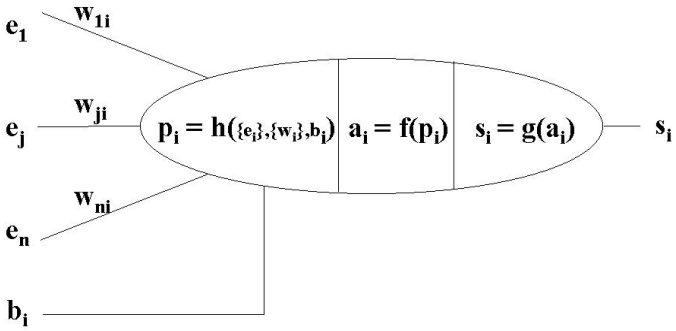


Figure 3.10: Le neurone formel général.

3.1.4 Généralisation du neurone formel

En 1986, Rumelhart [Rumelhart et al., 1986] définit le modèle général d'un neurone formel (Figure 3.10) et de leur assemblage en réseau.

De même que le modèle de McCulloch et Pitts, le modèle général du neurone formel possède : un vecteur d'entrée, un vecteur de poids associé au neurone, un biais, un état d'activation et une sortie. Un réseau de neurones artificiels se construit en commençant par définir une unité i:

- on définit tout d'abord la nature des entrées (binaires ou à valeurs réelles) qui caractérise soit l'environnement, soit les sorties d'autres neurones. On considère ce dernier comme un ensemble de valeurs d'entrée **e**. Suivant les modèles, on dispose également d'un ensemble de sorties désirées **d** associées respectivement à chaque vecteur d'entrée.

- de même, on définit la nature des sorties qui, elles aussi, peuvent être binaires ou réelles.

- on détermine l'entrée totale $p_i = h(.)$ définie par le prétraitement effectué sur les données (affine, polynomial, booléen, linéaire). Elle est généralement définie par une somme pondérée des entrées (Equation 3.5).

$$p_i = \sum_{j=1}^{n} w_{ji} e_j + b_i \tag{3.5}$$

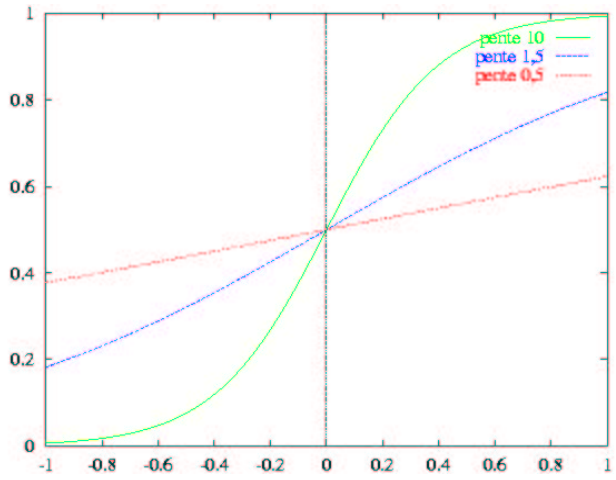
- on applique une fonction d'activation $f(.)$, appelée aussi fonction de transfert du neurone, qui définit son état interne a_i tel que $a_i = f(p_i)$. Les modèles de réseau de neurones sont caractérisés par la fonction de transfert qu'ils utilisent : fonction binaire à seuil (Equation 3.3, Figure 3.8) ou (Equation 3.4, Figure 3.9), fonction sigmoïde (Equation 3.6, Figure 3.11), fonction tangente hyperbolique ou toute autre fonction croissante et impaire. Cette fonction permet d'introduire une non-linéarité dans les réseaux de neurones.

- puis, la fonction de sortie $g(.)$ calcule la sortie du neurone à partir de son état d'activation. Généralement, on considère la fonction $g(.)$ comme la fonction identité, ainsi $s_i = a_i = f(p_i)$. La sortie s_i va contribuer aux calculs des états d'activations d'autres neurones en aval.

Une fois toutes les cellules spécifiées, on les assemble en réseau:

- suivant les modèles de réseaux, les cellules sont classées en plusieurs catégories. Dans les MLP, on en distingue trois types: les neurones d'entrée, les neurones de sortie et les neurones cachés.

- à ce stade, il est nécessaire de préciser un patron de connectivité, c'est-à-dire un schéma d'interconnexions entre les neurones. Ce schéma détermine une architecture particulière : complet, en couche, avec boucles ou avec connexions récurrentes.



$$f(x) = \frac{1}{1 + e^{-\beta x}} \quad (3.6)$$

β: pente de la sigmoïde

Figure 3.11: La fonction sigmoïde avec différentes pentes.

3.1.5 Les architectures

"Les réseaux de neurones sont constitués d'un assemblage de constituants élémentaires. Chacun de ces éléments effectue un traitement simple, mais pourtant l'ensemble fait émerger des propriétés globales dignes d'intérêt."

(Extrait de [Blayo et Verleysen, 1996])

L'architecture d'un réseau de neurones (RN) définit la disposition des neurones au sein du réseau par la structure des connexions entre les unités. Nous allons distinguer trois grandes catégories de structures.

Les structures à connexions complètes

Chaque unité du réseau de neurones est connectée à toutes les autres (Figure 3.12), y compris elle-même. Cette architecture a été introduite par [Hopfield, 1982] pour modéliser les systèmes dynamiques.

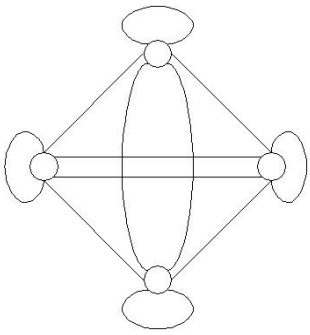


Figure 3.12: Topologie à connexions totales.

Les cartes topologiques de Kohonen

Il existe une catégorie de réseau de neurones dont la topologie est particulière, et n'a rien de commun avec les modèles précédents. Il s'agit du réseau de Kohonen (ou cartes topologiques de Kohonen) (Figure 3.13) [Kohonen, 1995]. Il faut préciser que les neurones des cartes topologiques de Kohonen ne sont pas des neurones formels comme ceux décrits plus haut.

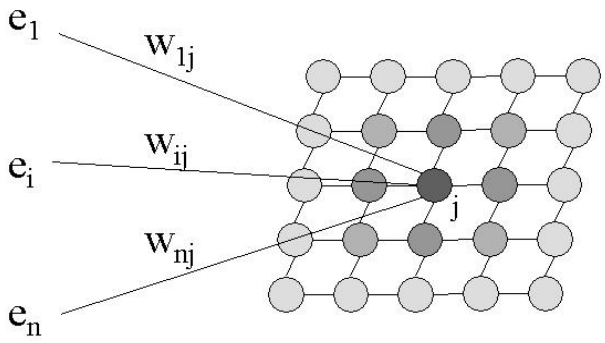


Figure 3.13: Carte topologique de Kohonen.

Ce modèle possède deux couches de connexions de nature différente. La première couche permet de relier les entrées avec tous les neurones de la carte. La seconde couche spécifie une structure d'interconnexions latérales, définissant ainsi une fonction d'interaction entre un neurone et ses voisins. Cette fonction d'interaction peut être représentée par une gaussienne (Figure 3.14) ou un "chapeau mexicain" (Figure 3.15). Plus on s'éloigne d'un neurone, plus l'interaction est faible et plus on se rapproche, plus l'interaction est forte.

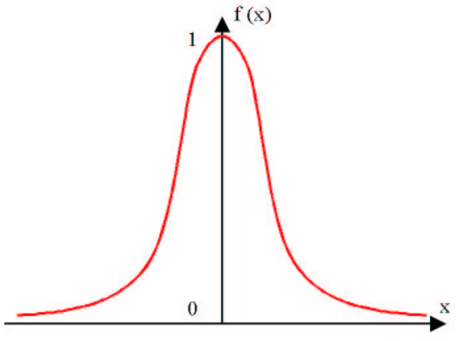


Figure 3.14: Fonction d'interaction du type Gaussien.

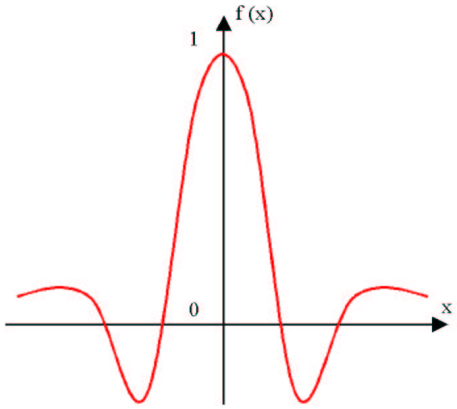


Figure 3.15: Fonction d'interaction du type chapeau Mexicain.

Les structures organisées en couches

Les neurones sont organisés en couches et l'information se propage de couche en couche (Figure 3.16). On distingue la couche d'entrée dont les unités reçoivent l'information, la couche de sortie dont les neurones fournissent l'information traitée et la ou les couches intermédiaires, appelées couches cachées dont les neurones sont exclusivement en relation avec les neurones d'entrées, de sorties ou d'autres neurones cachés.

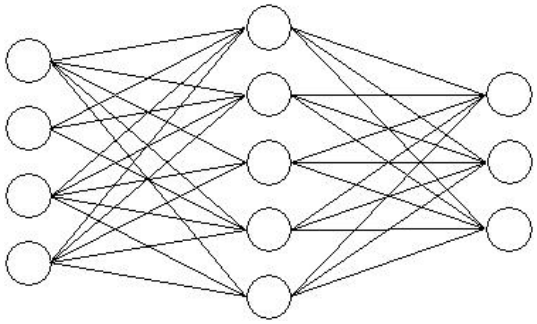


Figure 3.16: Topologie à connexions en couches.

On peut encore différencier plusieurs catégories de RN organisés en couches: RN à couches à connexions totales, RN à couches à connexions partielles, RN à couches à connexions partagées et RN à couches à connexions récurrentes.

- dans les RN à couches à connexions totales, tous les neurones de deux couches consécutives sont connectés entre eux.

- dans les RN à couches à connexions partielles, les neurones d'une couche ne sont connectés qu'à une partie des neurones de la couche suivante.

- dans les RN à couches à connexions partagées, on impose la contrainte que certaines connexions entre les neurones de deux couches consécutives sont identiques.

- les RN à couches à connexions récurrentes : jusqu'à présent, nous n'avons vu des modèles de RN où l'information se propage des neurones d'entrée vers les neurones de sortie de manière unidirectionnelle. Il est possible de définir des modèles qui permettent à l'information de revenir en arrière (Figure 3.17). Dans ce cas, le RN dispose de connexions récurrentes et devient ainsi plus adapté à la modélisation de séquences temporelles.

Les modèles de réseau de neurones organisés en couches sont parmi les plus utilisés et les plus efficaces. De plus, ils possèdent comme nous allons le voir de nombreux algorithmes pour les entraîner. Ils ont également fait l'objet de nombreuses études théoriques.

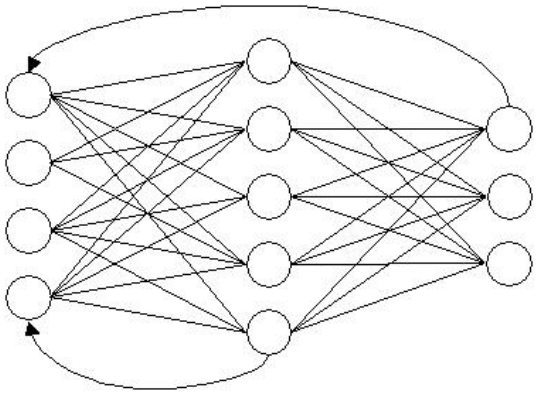


Figure 3.17: Topologie à connexions récurrentes.

Nous venons de voir trois grandes catégories de structures de connexions pour les RN. Cependant, il est tout à fait envisageable d'imaginer des approches hybrides qui combinent des réseaux de structures différentes.

Le fonctionnement du réseau fait évoluer les unités et les connexions. On parle de *dynamique des unités* et de *dynamique des connexions*:

- la dynamique des unités dépend de la règle de transition utilisée par les réseaux de neurones. L'état du système est défini par l'ensemble des activations de tous les neurones. Ces dernières sont déterminées par des règles de propagation (synchrones ou asynchrones) de l'information. Lorsque la propagation est synchrone, la propagation de l'information est réalisée en parallèle dans toutes les unités. Lorsque la propagation est asynchrone, on attend que tous les signaux soient arrivés à un neurone pour calculer son activation et poursuivre la propagation du signal vers les neurones en aval.

- la dynamique des connexions dépend de la règle d'apprentissage qui modifie les connexions ou leurs structures.

Généralement, les modifications ne portent que sur la valeur des connexions et la fonction de transfert est choisie a priori. La dynamique des connexions s'effectue postérieurement à la dynamique des unités. En effet, pour que les poids soient modifiés, le réseau doit avoir fourni l'information en sortie. Il faut donc à présent décrire un ensemble de règles d'apprentissage pour modifier les comportements d'un réseau de neurones. Plus formellement, un réseau de neurones réalise une fonction qui est définie par ses poids. De ce point de vue, apprendre c'est chercher la fonction qui modélise le mieux les données.

Nous allons à présent introduire les principes fondamentaux de l'apprentissage et de la généralisation pour les réseaux de neurones en particulier, mais aussi pour les machines à apprendre en général.

3.1.6 L'apprentissage dans les réseaux de neurones

"L'apprentissage, c'est-à-dire la capacité d'acquérir de nouveaux comportements ou d'en modifier d'anciens par expérience, est une propriété de presque tous les animaux, y compris ceux que l'on peut difficilement qualifier d'intelligents."

(Extrait de [Kandel et Schwartz, 1991]).

Idée générale

L'apprentissage est un processus d'acquisition de connaissances sur l'environnement. Il est lié à la mémoire et à la perception. Pour Kandel, il est défini comme la capacité d'acquérir de nouveaux comportements ou d'en modifier par l'expérience. Historiquement, voyons comment l'apprentissage a été mis en évidence.

C'est dans le domaine de la psychologie que l'apprentissage a été le plus étudié. Pavlov a développé la notion de conditionnement et Tolman a développé son explication en montrant qu'il n'y avait pas toujours de relation directe entre comportement et stimulation. Skinner a proposé des méthodes de conditionnement pour étudier les effets du renforcement ou de la punition. Il ressort de ces études que toutes les modifications comportementales postulent l'existence de la mémoire. La mémoire recouvre deux fonctions distinctes: l'acquisition d'information et l'extraction de l'information (le rappel). Le lien entre l'acquisition et le rappel se nomme l'engramme. Si l'engramme se manifeste par un changement de comportement, c'est qu'il se fonde sur une modification des structures cérébrales [Blayo et Verleysen, 1996]. Il y aurait donc une relation entre le changement de comportement et la modification des propriétés de la synapse et du neurone.

La mémoire reposerait sur deux processus: la mémoire à court terme et la mémoire à long terme. Le stockage de l'information dans une mémoire plutôt qu'une autre est lié à l'intérêt de cette information. Quels sont les processus mis en jeu pour le codage de l'information dans la mémoire ? Un élément de réponse a été apporté par le physiologiste D. Hebb en 1949, qui proposa un modèle simple pour expliquer ces modifications.

La règle de Hebb

Cette forme d'apprentissage est l'une des plus anciennes, elle a été proposée par Hebb en 1949 [Hebb, 1949]. En voici la formulation neurobiologique:
"*Quand une cellule A excite par son axone une cellule B et que, de manière répétée et persistante elle participe à la genèse d'une impulsion dans B, un processus de croissance ou un changement métabolique a lieu dans l'une ou dans les deux cellules de sorte que l'efficacité de A à déclencher une impulsion dans B est, parmi les autres cellules qui ont cet effet, accrue.*"

On peut résumer la formulation de Hebb de la manière suivante, pour un réseau de neurones artificiels :

- si les deux neurones situés de part et d'autre de la synapse (Figure 3.18) sont actifs simultanément (synchrones), alors on augmente le poids.
- si les deux neurones ne sont pas synchrones, la synapse décroît ou est éliminée.

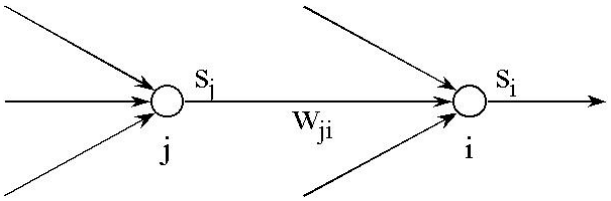


Figure 3.18: La règle d'apprentissage de Hebb.

Mathématiquement, ceci se traduit pour des sorties binaires ± 1 par :

$$w_{ij}^{t+1} = w_{ij}^{t} + \alpha\, s_i\, s_j$$

- s_i est la sortie du neurone i.
- w_{ij}^t est le poids de la connexion entre le neurone i et le neurone j à l'instant t.
- α est une constante qui représente la vitesse d'apprentissage. On l'appelle également le pas d'apprentissage.

Le règle de Hebb illustre l'apprentissage par renforcement. Elle ne tient compte que de l'activité entre deux unités et il n'est pas possible de guider l'apprentissage. Toutefois, il existe d'autres formes d'apprentissage.

Apprentissage supervisé et non supervisé

Les réseaux de neurones sont des mémoires associatives. Elles sont soit auto-associatives ou soit hétéro-associatives. Les mémoires auto-associatives associent des vecteurs d'entrée à eux-mêmes (diabolo[5], ou réseaux d'Hopfield). Les mémoires hétéro-associatives permettent d'effectuer les associations entre deux ensembles de vecteurs différents (perceptrons et perceptrons multicouches). On distingue alors différentes stratégies d'apprentissage :

- apprentissage supervisé: il s'agit d'un mode d'apprentissage qui est guidé. Pour chaque entrée, on fournit au réseau une sortie désirée, qui permet de mesurer l'erreur commise en sortie, afin de modifier le comportement du réseau pour diminuer cette erreur. L'algorithme d'apprentissage supervisé le plus courant est la rétropropagation du gradient de l'erreur pour les perceptrons multi-couches.

[5]Un diabolo est un réseau de neurones dont la tâche est de reconstruire, à l'identique en sortie, le vecteur présenté en entrée.

- apprentissage non-supervisé: ce mode d'apprentissage laisse le modèle s'adapter aux données. On considère que toute l'information nécessaire se trouve dans les données d'entrée. Les réseaux de Kohonen sont utilisés par exemple en apprentissage non supervisé pour effectuer des tâches simples de catégorisation.

- apprentissage par renforcement: à mi-chemin entre le mode supervisé et le mode non-supervisé, on trouve l'apprentissage par renforcement. Dans ce mode d'apprentissage, on ne donne pas explicitement les sorties désirées, mais on constate seulement la réussite ou l'échec de la tâche, puis on récompense ou bien on puni le réseau.

Dans le cadre de cette thèse, nous allons étudier uniquement les algorithmes supervisés pour les réseaux de neurones. Ainsi, un réseau de neurones à apprentissage supervisé peut réaliser trois tâches disctinctes :

1 classification : Soit $\mathcal{C}$ un ensemble de classes et $\mathbf{x}$ un vecteur d'entrée. Le problème de classification consiste à attribuer un exemple $\mathbf{x}$ à une classe donnée C_i.

2 régression : Soit $g(.)$ une application non-linéaire inconnue mais telle que $d = g(\mathbf{x})$ où $\mathbf{x}$ est le vecteur d'entrée et d est un scalaire. On cherche à construire un réseau de neurones qui approxime la fonction $g(.)$ grâce à un ensemble de couples d'exemples d'entrée-sortie $(\mathbf{x}_1, d_1) \ldots (\mathbf{x}_P, d_P)$ où P est le nombre de couples d'exemples.

3 prédiction : On veut construire un réseau de neurones qui effectue la prédiction d'un échantillon $x(t)$ d'un signal, étant donné les M échantillons passés de ce signal $x(t-1), x(t-2), \ldots, x(t-M)$. En réalité, la prédiction est un cas particulier de la régression. Mais le plus souvent, il est convenu de la différencier.

La règle du Perceptron de Rosenblatt

En 1958, Rosenblatt [Rosenblatt, 1958] propose une machine à apprentissage supervisé qu'il nomme le **perceptron** (Figure 3.19). Il se compose de:

- une **rétine**: elle est constituée de cellules à états binaires qui représentent les entrées du **perceptron**.

- une **couche de cellules d'association**, chacune de ces cellules est connectée à une ou plusieurs cellules de la rétine. En réalité, ce sont des prédicats booléens qui réalisent une extraction de primitives sur des sous-ensembles de la rétine.

- une ou plusieurs **cellules de décision**: ce sont des unités linéaires à seuil. Chacune d'elle est connectée à des cellules d'association. Seules ces connexions sont modifiées lors de l'apprentissage.

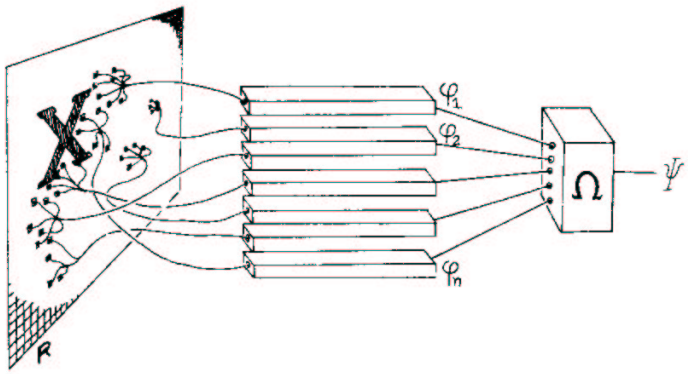


Figure 3.19: Le Perceptron de Rosenblatt [Minsky et Papert, 1969].

La modification des connexions entre les neurones de la couche d'association et les neurones de la couche de décision s'effectue grâce à la règle delta (Equation 3.7).

$$w_{ij}^{t+1} = w_{ij}^{t} + \alpha(d_j - s_j)x_i \tag{3.7}$$

où d_j et s_j représentent respectivement la sortie désirée et la sortie calculée, et x_i représente une cellule d'association.

Rosenblatt a démontré [Rosenblatt, 1962] que si on utilise un **perceptron** pour résoudre un problème donné et qu'il admet une solution, c'est-à-dire une matrice de poids qui résolve le problème, alors l'apprentissage par la règle du **perceptron** convergera nécessairement vers cette solution.

La règle de Widrow-Hoff

Cette règle publiée en 1960 est une variante de la règle du **perceptron** étendue à des valeurs réelles et qui minimise l'erreur en sortie [Widrow et Hoff, 1960]. Ici, le perceptron n'est plus un perceptron de Rosenblatt, il n'y a pas de rétine et pas de cellule d'association. La formule d'apprentissage est la suivante :

$$w_{ij}^{t+1} = w_{ij}^{t} + \alpha(d_j - a_j)s_i$$

où $(d_i - a_i)$ représente la différence entre la sortie désirée et la valeur d'activation de l'unité concernée.

Par la suite, Minsky et Papert [Minsky et Papert, 1969] montrèrent que les problèmes linéairement non-séparables, comme le problème du OU-exclusif, ne possèdent pas de solution pour un **perceptron** monocouche.

Le problème du OU-exclusif (XOR)

Le problème du OU-exclusif permet d'illustrer la notion de séparabilité linéaire. Un problème de classification est dit linéairement séparable s'il est possible de séparer

l'espace des solutions avec un hyperplan de telle sorte qu'aucune classe ne soit de part et d'autre de l'hyperplan.

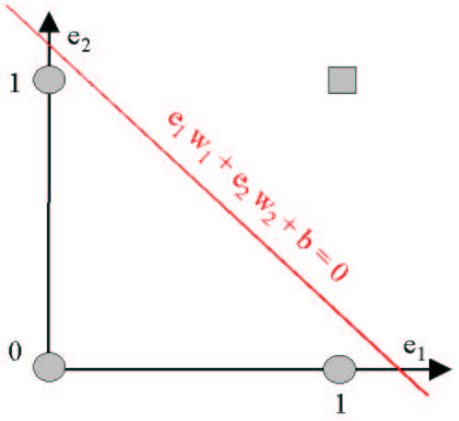


Figure 3.20: Le ET logique est linéairement séparable.

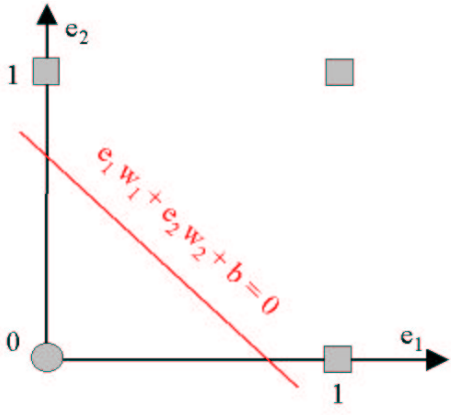


Figure 3.21: Le OU logique est linéairement séparable.

Il est très simple de trouver des solutions évidentes pour séparer linéairement un ET logique (Figure 3.20) et un OU logique (Figure 3.21) à l'aide d'un neurone à deux entrées. Prenons un neurone formel avec deux entrées e_1 et e_2. Ce neurone possède un biais b et deux poids w_1 et w_2 respectivement associés aux entrées e_1 et e_2. Pour le ET logique, on peut prendre $w_1 = 1$, $w_2 = 1$ et $b = 1.5$. Pour le OU logique, on peut prendre $w_1 = 1$, $w_2 = 1$ et $b = -0.5$. Par contre, il est impossible de trouver des valeurs pour w_1, w_2 et b telles qu'il existe une droite qui sépare les ronds et les carrés pour la figure 3.22 représentant le XOR.

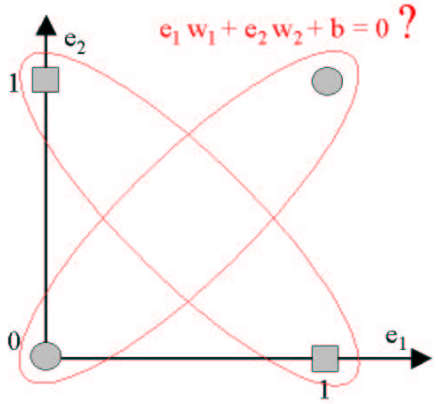


Figure 3.22: Le OU-exclusif n'est pas linéairement séparable.

Pour résoudre le problème du XOR, il faut une couche supplémentaire de neurones cachés. On dispose alors d'un **perceptron** multicouches.

Le perceptron multicouches

Le **perceptron multicouches** (MLP) est un réseau de neurones constitué d'une couche d'entrée, d'une ou de plusieurs couches cachées et d'une couche de sortie. Seuls les neurones de deux couches consécutives sont connectés (Figure 3.23).

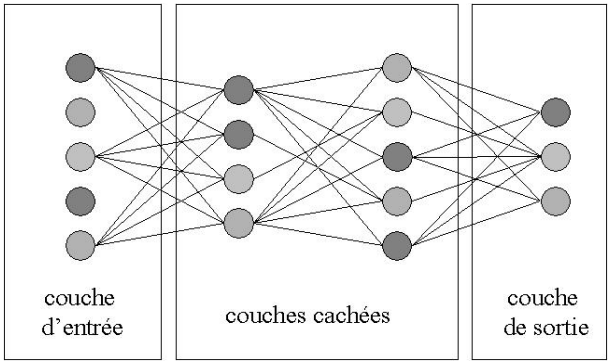


Figure 3.23: Le modèle du Perceptron multicouches.

Décrivons de manière plus formelle ce modèle. Soit l le numéro de la couche considérée, le perceptron multicouches est décrit par :

- N_l est le nombre de neurones de la couche l,
- L est le nombre de couches et le numéro de la couche de sortie,

$$s_j^l = f(a_j^l) \qquad (3.8) \qquad\qquad a_j^l = \sum_{i=1}^{N_{l-1}} w_{ij}^l s_i^{l-1} \qquad (3.9)$$

- s_j^l est la sortie du $j^{\text{ème}}$ neurone de la couche l,
- a_j^l est l'activation du $j^{\text{ème}}$ neurone de la couche l,
- $f(x)$ est la fonction sigmoïde (Equation 3.6).

Malheureusement, la règle delta ne permet pas de connaître les modifications à apporter aux connexions de ces neurones. C'est le problème du "Bouc Emissaire" ou Credit Assignment Problem.

Le problème du "Bouc Emissaire" (Credit Assignment Problem)

On dispose donc d'un **perceptron** multicouches (MLP). Il se pose alors un nouveau problème. En effet, dans le modèle du perceptron monocouche, nous avons une règle pour modifier les connexions entre la couche d'association et la couche de sortie. Cependant, dans un MLP nous n'avons pas l'équivalent de la règle delta pour modifier les connexions entre les neurones des couches cachées (Figure 3.24).

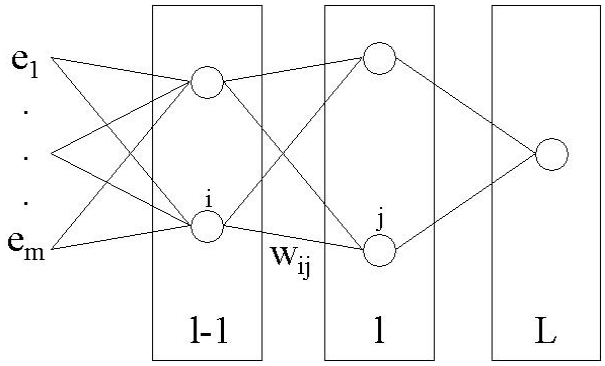


Figure 3.24: Le problème du Bouc émissaire : Comment modifier les poids des couches internes du réseau ?

En effet, comment déterminer l'erreur dans les neurones cachés ? De quelle manière contribue-t-elle à l'erreur commise dans les cellules de sorties ?

Pour résoudre le problème du "Bouc Emissaire", nous devons reformuler le problème en introduisant la notion de fonction de coût. On cherche donc une fonction de coût qui soit minimale lorsque, pour tous les exemples, l'erreur entre la sortie obtenue et la sortie désirée est minimale ou nulle, et qui soit dérivable et continue. Le plus souvent, on choisit la fonction de coût des moindres carrés $\sum_p (\mathbf{d}_p - \mathbf{s}_p)^2$ où p est un exemple d'apprentissage. Puis, on applique une procédure de minimisation à cette fonction de coût pour rechercher les paramètres de poids optimaux.

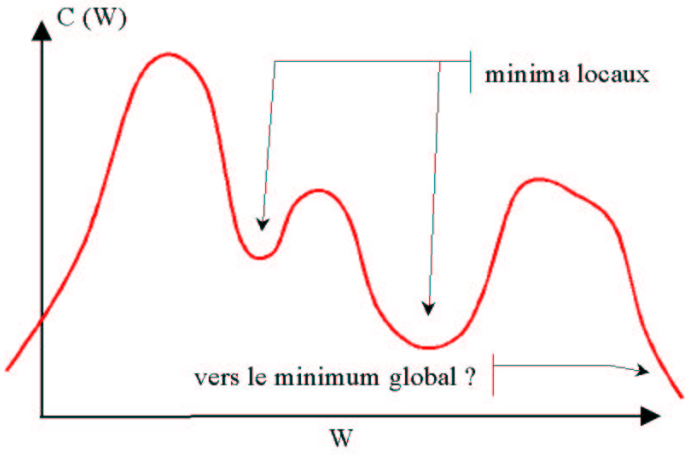


Figure 3.25: Le principe d'une fonction de coût et illustration des minima locaux.

Une fonction coût $C(W)$ peut être représentée dans l'espace des poids W du réseau (Figure 3.25). Lorsque le nombre de poids est supérieur à 2 ou 3, ce qui est souvent le cas, il devient impossible de représenter graphiquement cette fonction. Toutefois, il est possible de l'imaginer comme un paysage plus ou moins chaotique, avec des collines plus ou moins hautes et des ravins plus ou moins profonds. Le fond de ces ravins correspond à une valeur minimale de la fonction de coût.

On cherche le ravin le plus profond, c'est-à-dire celui qui aura la plus faible valeur de coût : le minimum global. Cependant, il est difficile à trouver car il y a beaucoup d'autres ravins moins profonds : les minima locaux.

La règle de la rétropropagation du gradient de l'erreur

C'est une règle très efficace, elle est donc majoritairement utilisée pour les réseaux multicouches. C'est la généralisation de la règle de Widrow-Hoff au perceptron multicouches.

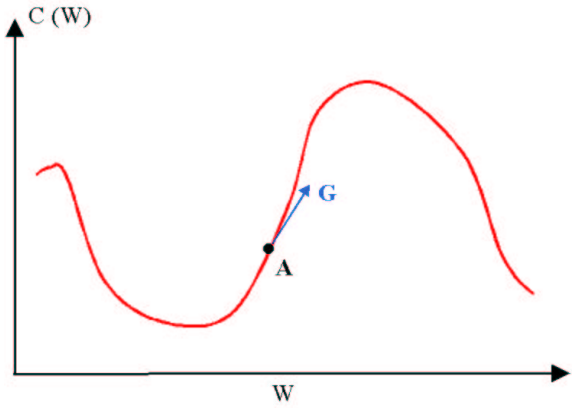


Figure 3.26: Le principe du gradient.

On considère un réseau de neurones artificiels comme un point **A** sur la fonction de coût, défini en coordonnées par la valeur de ses poids (Figure 3.26). Le principe de la procédure du gradient est de déplacer le point **A** dans le sens opposé du gradient **G** pour se rapprocher d'un minimum de la fonction de coût (Figure 3.27).

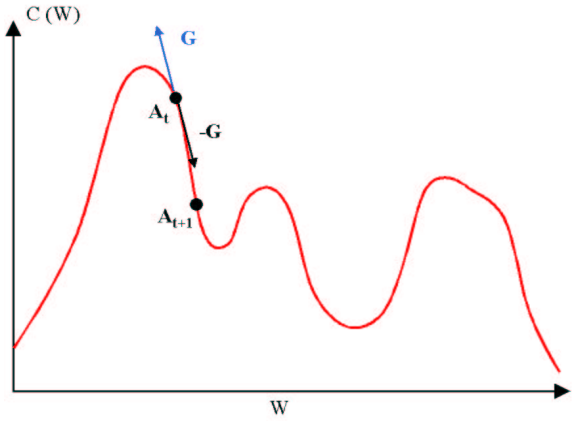


Figure 3.27: La procédure du gradient.

La procédure du gradient minimise donc une fonction de coût $C(W)$ par rapport aux poids W sur l'ensemble des exemples d'apprentissage $\mathcal{A} = (\mathcal{E}, \mathcal{D})$.

- $\mathcal{E} = (\mathbf{e}(1) \ldots \mathbf{e}(P))$ où $\mathcal{E}$ est l'ensemble des exemples d'entrées, $\mathbf{e}(p)$ est l'exemple p d'entrée avec $\mathbf{e}(p) \in \mathbb{R}^M$. M est la dimension d'entrée.
- $\mathcal{D} = (\mathbf{d}(1) \ldots \mathbf{d}(P))$ où $\mathcal{D}$ est l'ensemble des sorties désirées associées aux exemples d'entrées, $\mathbf{d}(p)$ est la sortie désirée respectivement associée à l'exemple d'entrée p avec $\mathbf{d}(p) \in \mathbb{R}^R$. R est la dimension de sortie.
- P est le nombre de motifs d'apprentissage.

Algorithme d'apprentissage "Batch"

La fonction de coût est définie sur tous les exemples d'apprentissage par une expression quadratique, la MSE (Mean Squared Error):

$$C(W) = \sum_{p=1}^{P} \sum_{r=1}^{R} (d_r(p) - s_r^L(p))^2 \tag{3.10}$$

Puis, on cherchera à minimiser cette fonction de coût, c'est-à-dire l'écart pour tous les exemples entre la sortie désirée et la sortie obtenue par le réseau. On calculera le gradient total $\frac{\partial C(W)}{\partial W}$.

Algorithme d'apprentissage Stochastique

On peut définir également une fonction de coût quadratique partielle pour un motif.

$$\begin{aligned} C(W) &= \sum_{p=1}^{P} C_p(W) \\ C_p(W) &= \sum_{r=1}^{R} (d_r(p) - s_r^L(p))^2 \end{aligned} \tag{3.11}$$

Puis, on cherchera à minimiser cette fonction de coût partielle, c'est-à-dire l'écart pour l'exemple courant p entre la sortie désirée et la sortie obtenue par le réseau. Nous allons développer le calcul du gradient partiel $\frac{\partial C_p(W)}{\partial W}$ dans le cas de l'apprentissage stochastique. Puis, nous montrerons que la procédure de mise à jour des poids est la même que pour l'apprentissage "Batch". On cherche donc à dériver la fonction de coût instantanée par rapport à un poids w_{ij}.

$$\begin{aligned} \frac{\partial C_p(W)}{\partial w_{ij}^l} &= \frac{\partial C_p(W)}{\partial a_j^l} . \frac{\partial a_j^l}{\partial w_{ij}^l} \\ &= \frac{\partial C_p(W)}{\partial a_j^l} . s_i^{l-1} \\ &= \frac{\partial C_p(W)}{\partial s_j^l} . \frac{\partial s_j^l}{\partial a_j^l} . s_i^{l-1} \\ &= \Phi_j^l . f'(a_j^l) . s_i^{l-1} \end{aligned} \tag{3.12}$$

On calcule la fonction Φ_j^l (Annexe A) définie par :

$$\Phi_j^L = \frac{\partial C_p(W)}{\partial s_j^L}$$

Le calcul de ce gradient et la modification des connexions sont réalisés par récurrence depuis les neurones de sortie vers les neurones d'entrée, dans le sens inverse de la propagation, d'où le terme de rétropropagation du gradient de l'erreur. Pour chaque connexion, l'adaptation se fait à l'aide de la règle delta généralisée (Equation 3.13) qui n'est rien d'autre qu'une descente du gradient :

$$w_{ij,t+1}^l = w_{ij,t}^l - \eta . \Delta w_{ij,t}^l \tag{3.13}$$

où η est appelé le pas d'apprentissage, et $\Delta w_{ij,t}^l$ est défini par :

$$\Delta w_{ij,t}^l = \begin{cases} l = L & : \quad f'(a_j^l)\, s_i^{l-1}\, 2\,(s_j^L(p) - d_j(p)) \\ l \neq L & : \quad f'(a_j^l)\, s_i^{l-1} \sum_{k=1}^{N_{l+1}} \Phi_k^{l+1} f'(a_k^{l+1}) w_{jk}^{l+1} \end{cases}$$

Dans l'algorithme stochastique, on présente un exemple au réseau, on calcule les gradients et on modifie les poids. Dans l'algorithme "Batch", on présente tous les exemples et on cumule chaque gradient Φ_j^l pour tous les exemples. Une fois tous les exemples présentés et tous les gradients cumulés, on modifie les poids.

Variantes: moment d'inertie, coefficients adaptatifs et fonction de coût

Il existe des variantes à la règle delta généralisée.

- Le moment d'inertie : on ajoute un terme qui dépend de la différence entre le poids actuel d'une connexion et son poids précédent. La règle est alors la suivante :

  $$w_{ij,t+1}^l = w_{ij,t}^l - \eta\, \Delta w_{ij,t}^l + \mu\, (w_{ij,t}^l - w_{ij,t-1}^l) \tag{3.14}$$

  où μ est appelé le coefficient d'inertie.

  On interprète l'influence de ce terme comme une énergie qui permettrait de sortir plus rapidement des minima locaux. Imaginons une colline (la fonction de coût) dans laquelle il y a des creux, des bosses et des ravins (les minima locaux) (Figure 3.28).

  Lâchons une bille du haut de la colline (conditions initiales de poids), si on accumule l'énergie durant la chute, alors l'inertie emmagasinée va permettre à la bille de sauter une bosse et donc de franchir le minimum local. L'inconvénient est que cela donne un nouveau paramètre à régler en plus du pas d'apprentissage.

- Les coefficients adaptatifs: le pas d'apprentissage n'est plus fixe, mais dépend de l'itération η_t. En effet, on doit se demander pourquoi le pas d'apprentissage devrait être le même pour toutes les connexions. Après tout, tous les poids n'ont pas la même influence suivant les exemples. Il paraît donc naturel d'essayer d'adapter les coefficients d'apprentissage. De la même manière, on peut imaginer d'adapter la pente de la fonction d'activation ou même le biais.

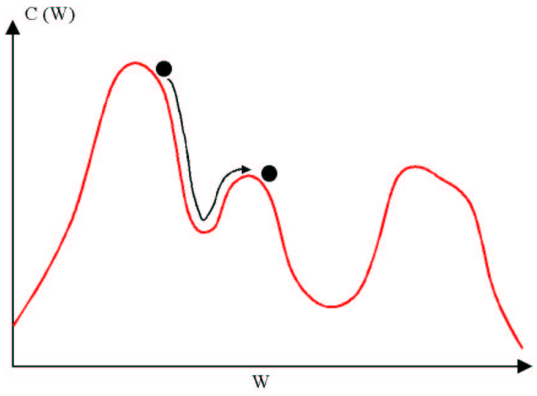


Figure 3.28: Le principe du moment d'inertie.

- La fonction de coût: une autre alternative consiste à modifier la fonction de coût classique suivant les types de problèmes à traiter. [Lemaire, 1999] propose d'utiliser, pour des problèmes de classification, un terme additif à la MSE. Cette nouvelle fonction de coût, appelée VMSE (Variance and Mean Squared Error), a pour effet d'augmenter la marge entre les classes et ainsi d'augmenter les performances en classification.

Comme nous l'avons dit, les méthodes de descente du gradient sont des mécanismes itératifs et sont potentiellement sensibles aux problèmes de minima locaux (notamment en basse dimension). Une autre approche est d'essayer de calculer directement le minimum local. La méthode de Newton est basée sur cette approche, elle n'élimine pas le problème des minima locaux, mais elle accélère la convergence.

La méthode de Newton

Le principe de la méthode de Newton se base sur l'approximation de fonction par la formule de Taylor (Equation 3.15). Cette dernière permet d'approximer une fonction $f(x)$ en un point x_0 avec une précision donnée n (ordre).

$$f(x) = f(x_0) + \sum_{i=1}^{n} \frac{1}{i!}(x - x_0)^i \frac{\partial^i f}{\partial^i x_0} \tag{3.15}$$

La méthode de Newton utilise la formule de Taylor pour approximer à l'ordre 2 une fonction de coût quadratique $C(W)$. On veut déterminer les nouveaux poids W^{t+1} :

$$\begin{aligned} C(W^{t+1}) &= C(W^t) + \nabla C(W^t)^T (W^{t+1} - W^t) \\ &+ \frac{1}{2}{}^T (W^{t+1} - W^t)^T H (W^{t+1} - W^t) \end{aligned}$$

où H est le Hessien, la matrice des dérivées secondes des poids W^t.

On dérive $C(W^{t+1})$ et on obtient:

$$\nabla C(W^{t+1}) = \nabla C(W^t) + H(W^{t+1} - W^t)$$

enfin, on annule le gradient $\nabla C(W^{t+1}) = 0$ pour déterminer W^{t+1}, d'où :

$$W^{t+1} = -H^{-1}\nabla C(W^t) + W^t \tag{3.16}$$

cela nous donne la formule de réestimation suivante pour chaque connexion :

$$\Delta w_{ij} = -\eta . \frac{\frac{\partial C}{\partial w_{ij}}}{\frac{\partial^2 C}{\partial^2 w_{ij}}} \tag{3.17}$$

Cependant, cette méthode est très coûteuse puisqu'elle nécessite l'inversion de la matrice des dérivées secondes des poids.

Il existe d'autres méthodes d'optimisation pour trouver les poids qui minimisent la fonction de coût. On peut citer les méthodes du gradient conjugué (algorithmes de Fletcher-Reeves [Fletcher et Reeves, 1964] ou de Polak-Ribiére [Hagan et al., 1996]), l'algorithme Quasi-Newton [Broyden, 1970] [Fletcher, 1970] [Golfarb, 1970] [Shanno, 1970] ou Levenberg-Marquardt [Levenberg, 1944] [Marquardt, 1963].

Autres aspects de l'apprentissage pour les réseaux de neurones

D'autres aspects sont très importants à prendre en compte pour assurer un bon déroulement du processus d'apprentissage.

- Mélanger les exemples :
  Cet aspect est très important lors de l'apprentissage stochastique. Certains auteurs ont proposés de choisir l'exemple le plus informatif. Par exemple, celui avec la plus grande erreur ou celui qui est le plus différent des précédents exemples présentés au réseau. Généralement, pour des problèmes de classification, on mélange l'ensemble d'apprentissage de tel sorte que deux exemples successifs n'appartiennent jamais ou très rarement à la même classe. Il est également possible de choisir des exemples "difficiles" plus souvent que des exemples "faciles" en prenant en compte, par exemple, leur erreur à l'itération précédente. Cependant, il est important de préciser que ces techniques, dites de "boosting" [Freund et Schapire, 1997] ou de "bagging" [Breiman, 1996] qui accélèrent l'apprentissage, sont très sensibles à la présence d'"outliers"[6] ou d'exemples mal étiquetés.

- Choisir la fonction d'activation :
  Le plus souvent, on choisit la fonction sigmoïde, mais certains lui préfèrent la fonction tangente hyperbolique, qui a la propriété d'être symétrique. Yann Le Cun, par exemple, utilise la fonction $f(x) = 1.7159 \tanh(\frac{2}{3}x)$ [LeCun, 1987].

[6]Un "outlier" est un individu atypique de la distribution de probabilité de l'ensemble des exemples dont il est tiré.

- Normaliser les entrées :
  Il est impératif de respecter la règle suivante. Chaque composante de chaque exemple d'entrée doit être décalée telle que sa moyenne sur l'ensemble d'apprentissage soit proche de 0. Typiquement, on fait la moyenne des exemples d'apprentissage, et on retranche l'exemple moyen à tous les exemples avant de les présenter au réseau. Il existe d'autres méthodes supplémentaires pour normaliser les exemples d'entrée, comme celle de normaliser l'écart type. Le lecteur intéressé peut lire à ce sujet [Bottou, 1991] ou [LeCun et al., 1998].

- Choisir les valeurs cibles à apprendre en sortie du réseau de neurones :
  Il faut choisir des cibles qui sont en accord avec le domaine de définition de la fonction d'activation, pour éviter que les valeurs des neurones de sortie saturent. Pour un problème de classification, on choisira pour la fonction sigmoïde standard 0.1 comme cible basse et 0.9 comme cible haute, et pour la fonction tangente hyperbolique -0.6 comme cible basse et 0.6 comme cible haute.

- Initialiser les poids :
  L'algorithme de la rétropropagation du gradient de l'erreur est très sensible aux minima locaux. Le choix d'une configuration initiale des poids est primordial. De plus, il faut choisir les poids de façon à utiliser la partie linéaire de la fonction de transfert. En effet, les petits poids correspondent à de forts gradients et accélère la convergence. [Bottou, 1991] propose d'initialiser les poids d'une unité i selon une loi uniforme dans l'intervalle $[-\frac{k}{\sqrt{n}} \ldots \frac{k}{\sqrt{n}}]$, où n est le "FanIn" c'est-à-dire le nombre d'entrées de l'unité i, et k dépend de la fonction d'activation. Léon Bottou propose de prendre $0 < k < 2$ pour la fonction de transfert de Yann Le Cun $f(x) = 1.7159 \tanh(\frac{2}{3}x)$.

- Choisir les pas d'apprentissage :
  Chaque poids devrait avoir son propre pas d'apprentissage. En effet, certains poids pourraient avoir besoin d'un petit pas pour éviter de diverger, alors que d'autres auraient besoin d'un grand pas pour converger. En pratique, on utilise le même pas d'apprentissage pour toutes les connexions. Dans ce cas, un pas trop fort sortira constamment des minima locaux et un pas trop faible rendra la convergence très lente. Il n'y a pas de règle universelle, il faut le déterminer expérimentalement.

- Choisir le nombre de couches et de neurones cachés :
  Les réseaux de neurones multicouches sont des approximateurs de fonctions universels. En théorie, un seule couche de neurones cachés suffit. En ce qui concerne le nombre de neurones sur cette couche, là non plus il n'existe pas de règle absolue. Il faut le déterminer expérimentalement.

3.1.7 Résultats théoriques: apprentissage par l'exemple, capacité et généralisation

Plusieurs résultats théoriques [Hornik et al., 1989] [Cybenko, 1989] montrent qu'un réseau de neurones multicouches et sans récurrence peut approximer, avec une précision arbitraire, n'importe quelle transformation continue d'un espace à dimension finie vers un autre espace à dimension finie, s'il possède suffisamment de neurones cachés. Les réseaux de neurones multicouches sont des approximateurs de fonctions universels.

Soit X une variable aléatoire de distribution de probabilité fixe P_X mais inconnue. Soit la fonction $\phi(x) : X \rightarrow Y$ telle que par exemple $x \in X$ est l'image d'une main et $y \in Y$ est la posture (forme de la main) associée. L'objectif d'une machine à apprendre est de produire une fonction paramétrique $\hat{\phi}(x, \theta) : X \rightarrow Y$ qui minimise une fonction de coût :

$$\int_{x \in X} J(\hat{\phi}(x, \theta), \phi(x)) P_X(x) dx \tag{3.18}$$

On dispose pour cela de N couples $(x_i, \phi(x_i))$ avec $i = 1 \ldots N$ et $(x_i, \phi(x_i)) \in (X, Y)$ suivant P_X. On mesure la performance d'une machine à apprendre en regardant la différence entre $\hat{\phi}$ et ϕ aux N points.

$$\sum_{i=1}^{N} J(\hat{\phi}(x_i, \theta), \phi(x_i)) \tag{3.19}$$

On ne peut pas parler d'apprentissage sans évoquer le problème de la **généralisation**. La généralisation est la faculté d'étendre une compétence à des exemples non appris. On a obtenu une machine à apprendre qui a appris $\hat{\phi}$, une approximation de ϕ. Comment se comporte $\hat{\phi}$ sur des points différents des N appris ? Avant d'approfondir cette question, il est nécessaire d'introduire la notion de capacité. En effet, des résultats théoriques permettent d'établir une relation entre la généralisation d'une machine à apprendre et sa capacité.

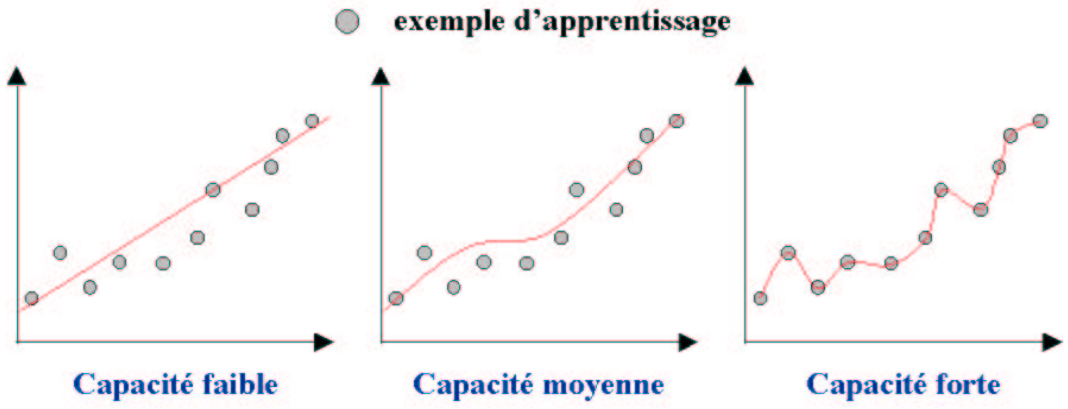


Figure 3.29: Illustration de la capacité.

Capacité

La notion de capacité a été introduite par Vapnik [Vapnik, 1995] dans la théorie de l'apprentissage. Pour illustrer la notion de capacité, on considère une suite de points que l'on cherche à interpoler avec un polynôme (Figure 3.29). D'un côté, si le polynôme n'a pas assez de coefficients (capacité faible) il apprendra mal les données. D'un autre côté, si le polynôme a trop de coefficients (capacité forte), il apprendra par cœur tous les points.

Ainsi, si l'on considère une machine à apprendre dans le cadre des problèmes de classification, la capacité représente le nombre maximum d'exemples que la machine peut classifier, quelque soit l'étiquetage. On appelle la capacité d'une machine à apprendre la dimension de Vapnik-Chervonenkis (VCdim) [Vapnik et Chervonenkis, 1971] [Vapnik, 1982]. De manière plus formelle :

Soit $z = (x, y) \in (X, Y)$ et $G(z, \theta)$ l'ensemble des fonctions paramétriques $g(z, \theta)$. Par exemple, ϕ est un réseau de neurones, $J(.)$ est une fonction de coût, et θ correspond aux poids du réseau de neurones.

$$g(z, \theta) = J(\hat{\phi}(x, \theta), \phi(x)) \tag{3.20}$$

Si $J \in \{0, 1\}$, la capacité de $G(z, \theta)$ est le nombre maximum de points $x_1 \dots x_h$ pouvant être séparés à tout coup en deux classes distinctes à l'aide d'une fonction dans $G(z, \theta)$. Vapnik donne une extension de sa définition pour des problèmes de régression où $J \in \mathbb{R}$. La capacité devient le nombre maximum d'associations $z_1 \dots z_h$ pouvant être apprises à l'aide d'une fonction dans $G(z, \theta)$ avec une précision arbitraire.

Généralisation

Pour illustrer la notion de généralisation, nous allons faire une analogie avec la régression statistique. Si un modèle statistique possède trop de paramètres libres, il se produit un phénomène d'"overfitting". Pour comprendre ce phénomène, il faut voir l'ensemble d'apprentissage comme une suite de données bruitées dont on cherche à extraire la partie déterministe.

Reprenons une suite de points épars décomposée en deux ensembles : exemples d'apprentissage et exemples de généralisation. Notre objectif est d'interpoler le mieux possible ces deux ensembles à l'aide d'un polynôme dont les coefficients sont calculés uniquement avec les points de l'ensemble d'apprentissage (Figure 3.30). Si le nombre de coefficients du polynôme est trop élevé (capacité forte), il se produit le phénomène d'"overfitting" : le système apprend par cœur les exemples d'apprentissage et les performances sur les exemples de généralisation sont mauvaises. Si le nombre de coefficients est trop faible, le polynôme réalisera une interpolation linéaire et les performances en apprentissage et en généralisation ne seront encore pas satisfaisantes. On voit alors nettement qu'un polynôme avec un nombre de coefficient moyen obtiendra un meilleur compromis entre l'erreur d'apprentissage et celle de généralisation.

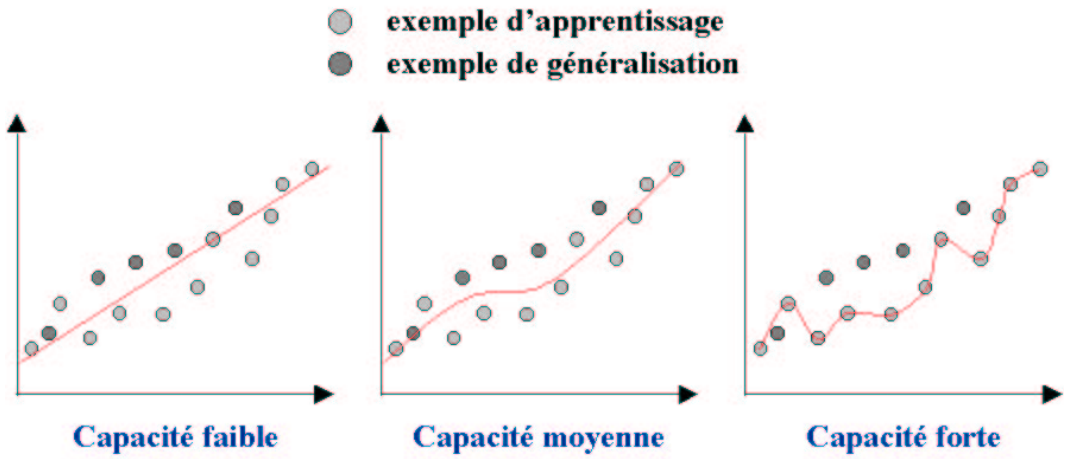


Figure 3.30: Illustration de la notion de généralisation.

Nous venons de voir que le nombre de degrés de liberté d'un système influence la généralisation. Vapnik et Chervonenkis [Vapnik et Chervonenkis, 1971; Vapnik, 1982] ont montré un certain nombre de résultats concernant l'erreur d'apprentissage et l'erreur de généralisation.

Soit $\mathcal{A}$ l'ensemble fini des exemples d'apprentissage $(x_i, \phi(x_i))$, $\mathcal{G}$ l'ensemble infini des exemples de généralisation $(x_j, \phi(x_j))$ et $J(.,.)$ une fonction de coût. Nous rappelons que $\phi(x_i)$ représente la sortie désirée associée à l'exemple i et $\hat{\phi}(x_i, \theta)$ la sortie obtenue après propagation dans le réseau de neurones θ, également associé à l'exemple i. On définit alors l'**erreur d'apprentissage** ϵ_A et l'**erreur de généralisation** ϵ_G par:

$$\begin{aligned} \epsilon_A &= \sum_{x_i \in A} J(\hat{\phi}(x_i, \theta), \phi(x_i)) \\ \epsilon_G &= \int_{x_j \in G} J(\hat{\phi}(x_j, \theta), \phi(x_j))\, dx_j \end{aligned}$$

Soit une machine à apprendre de VCdim h, et ayant appris un ensemble de N exemples d'une fonction ϕ de distribution de probabilité fixe mais inconnue, alors on peut obtenir une borne supérieure sur la différence espérée entre l'erreur d'apprentissage et l'erreur de généralisation (Equation 3.21).

$$\epsilon_A - \epsilon_G \leq \mathcal{O}(\sqrt{\frac{h}{N}}\, ln\frac{N}{h}) \tag{3.21}$$

On en déduit les résultats suivants. Dans un premier temps, on fixe le nombre d'exemples d'apprentissage N (Figure 3.31). Si on commence l'apprentissage avec une capacité faible et qu'on l'augmente progressivement, alors l'erreur de généralisation décroît jusqu'à une certaine valeur de la capacité et ensuite, cette erreur augmente avec la capacité. Dans un deuxième temps, on fixe la capacité et on augmente progressivement le nombre d'exemples d'apprentissage (Figure 3.32).

On constate alors que l'erreur d'apprentissage croît vers une valeur asymptotique et que l'erreur de généralisation décroît vers cette même valeur. Il a été démontré que la généralisation est influencée par le nombre d'exemples utilisés pour l'apprentissage, par l'architecture du réseau et par l'algorithme d'apprentissage.

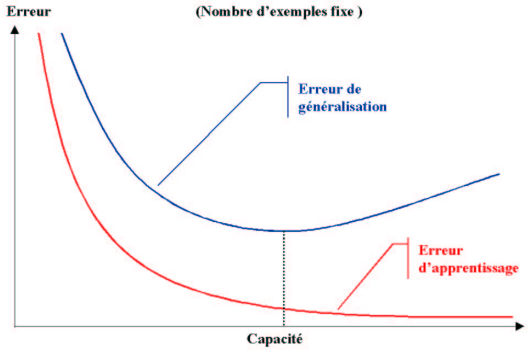


Figure 3.31: Borne de l'erreur à nombre d'exemples fixe.

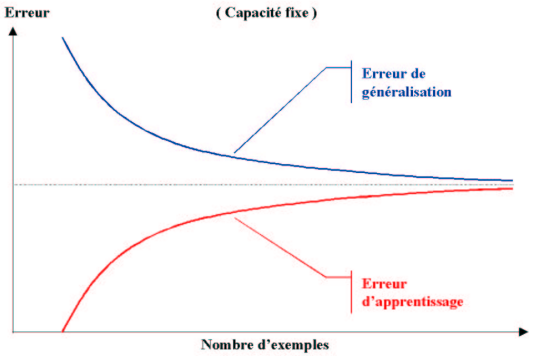


Figure 3.32: Borne de l'erreur à capacité fixe.

Ainsi, pour une capacité arbitraire h il est théoriquement possible de trouver une estimation du nombre d'exemples nécessaires pour l'apprentissage, pour obtenir une erreur de généralisation basse. Cependant, il est très compliqué de calculer la VCdim h d'un réseau de neurones multicouches à fonctions d'activations non-linéaires à valeurs dans $\mathbb{R}$. En réalité, lors d'un apprentissage, toute la capacité n'est pas exploitée. On suppose que seul un sous-ensemble de toutes les fonctions réalisables est exploré. On parle alors de capacité effective, celle-ci dépend de la procédure d'apprentissage (pas d'apprentissage, nombre d'itérations, etc).

Toutefois, il existe d'autres façons d'améliorer la généralisation d'un réseau de neurones. Toutes ces méthodes recherchent des architectures optimales en tentant, soit de réduire le nombre de poids à partir d'une capacité forte, ou soit d'augmenter le nombre de poids à partir d'une capacité faible. En effet, nous avons vu que trop de poids condamne le réseau à apprendre par cœur les exemples d'apprentissage et conduit ainsi à une faible généralisation.

Contrôler la capacité pour améliorer la généralisation

Réseaux à poids partagés

La méthode des poids partagés consiste à imposer des connexions locales et à forcer certaines connexions à partager les mêmes poids, ce qui permet de réduire le nombre de paramètres. Les réseaux à poids partagés ont été utilisés en reconnaissance de la parole [Waibel, 1989] ou en reconnaissance de l'écriture manuscrite [Le Cun et al., 1989].

"Optimal Brain Damage"

Une autre approche pour diminuer le nombre de connexions a été proposée par [LeCun et al., 1990]. Cette technique appelée "Optimal Brain Damage" consiste à

supprimer les connexions dont la disparition n'aura que peu d'influence sur l'erreur d'apprentissage.

Elimination des poids et décroissance exponentielle des poids
Dans ces deux méthodes, on ajoute à la fonction de coût un terme de complexité qui est fonction des poids, par exemple la somme des carrés des poids appelée aussi "weight decay" ou décroissance exponentielle des poids [Hinton, 1986].

Ajouter des poids : "cascade-correlation" et "structural risk minimisation"
Ces méthodes ont pour but d'augmenter la capacité à partir d'une capacité minimale. La "cascade-correlation" [Fahlman et Lebiere, 1990] [Fahlman et Lebiere, 1991] est une méthode qui crée des nouvelles unités cachées. Pour chacune de ces unités, l'algorithme tente de maximiser la magnitude de la corrélation entre les sorties de ces nouvelles unités et l'erreur résiduelle du réseau. L'approche "structural risk minimization" [Vapnik et Chervonenkis, 1974] [Guyon et al., 1992] permet d'explorer différentes architectures dont la capacité est croissante. Chaque architecture est entraînée sur un ensemble d'apprentissage, puis on mesure l'erreur commise par chaque architecture sur l'erreur de validation. L'architecture retenue est celle qui possède l'erreur de validation la plus faible.

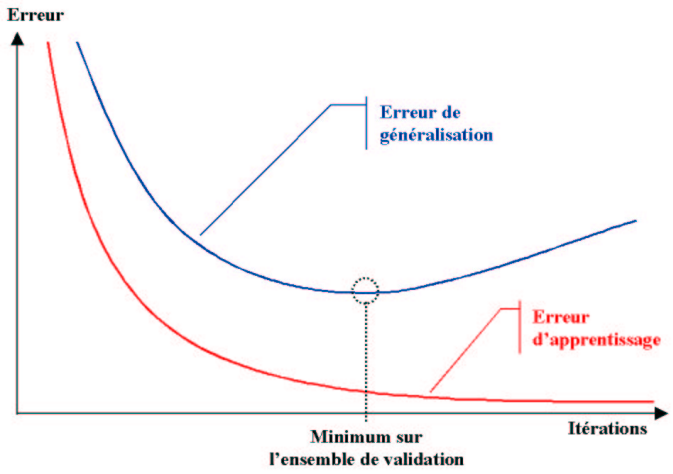


Figure 3.33: Critère d'arrêt de l'apprentissage.

Critère d'arrêt pour l'apprentissage
Nous pouvons conclure qu'il est important de définir un critère d'arrêt pour l'algorithme d'apprentissage. En effet, un réseau de neurones qui a appris par cœur l'ensemble d'apprentissage et qui est incapable de généraliser, ne sert à rien. Il existe plusieurs critères d'arrêt :

- on peut, par exemple, décider de stopper l'algorithme quand il a atteint un nombre d'itérations maximal ou quand le coût a atteint un certain seuil, mais cela nécessite de connaître la valeur à atteindre, ce qui est en général assez difficile.

- on peut utiliser la méthode de la validation. Les données sont séparées en trois ensembles : un ensemble d'apprentissage, un ensemble de validation et un ensemble de test. L'ensemble de test est utilisé pour contrôler la généralisation du réseau. Pendant l'apprentissage, le coût sur les points d'apprentissage diminue, celui sur les points de validation diminue jusqu'à un certain point, puis augmente (Figure 3.33). A partir de ce point, le réseau apprend par cœur les données de l'ensemble d'apprentissage et l'algorithme doit être stoppé. En utilisant cette méthode, on trouve l'optimum de la généralisation sur la base de validation pour l'architecture choisie et pour la fonction de coût choisie.

Estimation des performances en généralisation

Le choix du critère d'arrêt est primordial. Il va permettre d'arrêter l'apprentissage avant que la capacité effective n'augmente trop et qu'elle ne conduise à un sur-apprentissage. Le plus souvent, l'estimation de la qualité du réseau est obtenue en mesurant simplement l'erreur sur l'ensemble de test.

Plus rigoureusement, il est possible d'utiliser la méthode de la validation croisée qui permet de nous donner une meilleure estimation de l'erreur de généralisation. Cette dernière consiste à effectuer N apprentissages d'une architecture précise à l'aide d'un ensemble de données $\mathcal{D}$ réparti en trois sous-ensembles (apprentissage $\mathcal{A}$, validation $\mathcal{V}$ et test $\mathcal{T}$) et distribué différemment pour chacun des N apprentissages. Pour chacune des étapes $k = 1 \dots N$, on apprend avec $\mathcal{A}_k$ jusqu'à avoir atteint le minimum sur l'ensemble de validation $\mathcal{V}_k$, puis on évalue le modèle sur l'ensemble de test $\mathcal{T}_k$. Finalement, l'erreur de généralisation est calculée en effectuant la moyenne sur tous les apprentissages des erreurs commises sur les N ensembles de test $\mathcal{T}_k$. La méthode de la validation croisée n'est pas seulement une méthode pour calculer l'erreur de généralisation, mais elle permet également de montrer l'efficacité d'un algorithme d'apprentissage.

3.2 Les modèles de Markov cachés

En traitement de l'information, les données sont souvent de nature séquentielle. Ces données proviennent d'observations de phénomènes discrets (caractères alphabétiques ou codes génétiques) ou de phénomènes continus (températures, musique, parole, gestes).

Ces données forment un ensemble de séquences d'observations dont la longueur peut être variable. Lorsqu'une observation est de nature discrète, elle est considérée comme l'indice d'un élément d'un ensemble de symboles observables : le "codebook". Lorsque l'observation est de nature continue, il est possible de la quantifier pour se ramener au cas discret, mais on risque alors de dégrader l'observation. Toutefois, il est possible d'utiliser directement les observations continues en les modélisant par une fonction de densité de probabilité (**pdf**) ou par des mélanges de **pdf**.

Dans la suite de ce chapitre, nous abordons le modèle formel des observations et des modèles de Markov cachés. Nous présentons son utilisation et détaillons ses mécanismes d'apprentissage.

3.2.1 Introduction

Un modèle de Markov caché est un modèle statistique capable de modéliser un processus dynamique aléatoire. Ce modèle est doté d'un algorithme lui permettant d'estimer ses paramètres. Il est composé d'un ensemble d'états, dit états cachés car non observables, et il est doté de probabilités de transitions entre les états cachés. Il possède également des probabilités d'émissions pour modéliser les observations. Les modèles de Markov cachés sont utilisés dans une grande variété d'applications, comme la reconnaissance de la parole [Rabiner, 1989] ou de l'écriture manuscrite [LeCun, 1985].

3.2.2 Formalisation des observations et du modèle

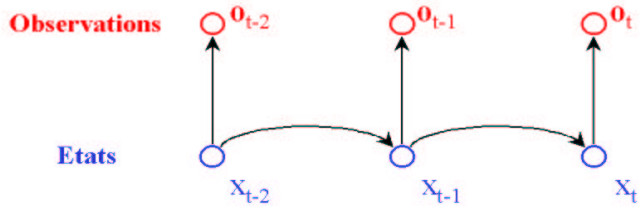
a. Les observations

Soit $\mathcal{O} = \{\mathbf{o}_1^{T_p}(p) \dots \mathbf{o}_1^{T_P}(P)\}\ \forall p = 1 \dots P$, l'ensemble des séquences d'obervations, où P est le nombre de séquences et T_p est la longueur de la séquence p couramment observée.

Par simplification, on peut omettre la variable p :

- $\mathbf{o}_1^T = \mathbf{o}_1 \dots \mathbf{o}_T$ est une séquence d'observations de nature discrète ou continue,
- T est la longueur de cette séquence.

Soit N le nombre d'états du modèle, x_t est une variable discrète représentant l'état du modèle (Figure 3.34) au temps t, avec $x_t \in \{1 \dots N\}$.



Les hypothèses Markoviennes associées aux HMM sont : $Pr(x_t \mid x_{t-1}, x_{t-2}, \ldots, \Lambda) = Pr(x_t \mid x_{t-1}, \Lambda)$ et $Pr(\mathbf{o}_t \mid x_t, x_{t-1}, \ldots, \Lambda) = Pr(\mathbf{o}_t \mid x_t, \Lambda)$.

Figure 3.34: Le modèle du HMM.

b. Le modèle

Un modèle de modèle de Markov caché (HMM) est l'ensemble des paramètres $\Lambda = \{\Pi, A, B\}$.

- le vecteur de paramètres $\Pi = (\pi_i)$ représente la distribution initiale des états. $\pi_i = \Pr(x_1 = i)$ avec $\forall i, i \in \{1 \ldots N\}$.
- la matrice $A = (a_{ij})$ détermine les probabilités de transitions d'un état à l'instant t vers un autre état à l'instant $t+1$.
  $a_{ij} = \Pr(x_{t+1} = j \mid x_t = i)$ avec $\forall i \forall j, (i,j) \in \{1 \ldots N\}^2$
- l'ensemble de paramètres $B = (b_j(\mathbf{o}))$ représente la probabilité d'observation de $\mathbf{o}$ dans l'état j. $b_j(\mathbf{o}_t) = \Pr(\mathbf{o}_t \mid x_t = j)$ avec $\forall j, j \in \{1 \ldots N\}$.
  On distingue deux cas :
  - **Les observations sont discrètes** : Dans ce cas, on dispose de D symboles distincts observables : $\forall \mathbf{o}_t, \mathbf{o}_t \in \{1 \ldots D\}$.
  - **Les observations sont continues** : Dans ce cas, les observations sont des vecteurs de dimension D : $\mathbf{o}_t \in \mathbb{R}^D$. On peut représenter $b_j(\mathbf{o}_t)$ par un mélange de M **pdf** (par exemple des gausiennes) :
    $b_j(\mathbf{o}_t) = \sum_{m=1}^{M} c_{jm} \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm})$ $\forall j, j \in \{1 \ldots N\}$.
    - c_{jm} est le coefficient de la **pdf** m dans l'état j. La somme des c_{jm}, $\sum_{m=1}^{M} c_{jm}$ vaut 1 $\forall j, j \in \{1 \ldots N\}$,
    - $\mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm})$ est la fonction de densité de probabilité, où si on suppose que $\mathcal{M}$ est une gaussienne :

$$\mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm}) = \sqrt{\frac{\mid \boldsymbol{\sigma}_{jm} \mid}{(2\pi)^D}} e^{-\frac{1}{2}^T (\mathbf{o}_t - \boldsymbol{\mu}_{jm}) \boldsymbol{\sigma}_{jm} (\mathbf{o}_t - \boldsymbol{\mu}_{jm})} \qquad (3.22)$$

    - $\boldsymbol{\mu}_{jm}$ est le vecteur moyen de la **pdf** m dans l'état j avec $\forall j, j \in \{1 \ldots N\}$ et $\forall m, m \in \{1 \ldots M\}$,
    - $\boldsymbol{\sigma}_{jm}$ est l'inverse de la matrice de covariance de la **pdf** m dans l'état j $\forall j, j \in \{1 \ldots N\}$ et $\forall m, m \in \{1 \ldots M\}$,

Pour utiliser un HMM avec les séquences d'observations, il faut pouvoir réaliser trois opérations essentielles. La première va nous permettre de calculer la probabilité qu'une séquence observée soit générée par le HMM. La seconde va nous permettre de régler les paramètres du HMM pour l'adapter à la distribution de probabilité de l'ensemble des séquences d'observations. La dernière opération trouve une représentation interne de la séquence d'observation sous la forme d'une séquence d'état.

1 Calculer $\Pr(\mathbf{o}_1^T \mid \Lambda)$: la probabilité d'observation de la séquence $\mathbf{o}_1^T = \mathbf{o}_1 \dots \mathbf{o}_T$ étant donné le modèle Λ.

2 Ajuster les paramètres de Λ pour maximiser $\Pr(\mathbf{o}_1^T \mid \Lambda)$ sur toutes les séquences p.

3 Etant donnés $\mathbf{o}_1^T$ et Λ, trouver la séquence d'états $x_1^T = x_1 \dots x_T$ qui correspond le mieux aux observations.

3.2.3 Calcul de $\Pr(\mathbf{o}_1^T \mid \Lambda)$

Dans un HMM, on peut associer à une séquence d'observations $\mathbf{o}_1^T = \mathbf{o}_1 \dots \mathbf{o}_T$, une multitude de séquences d'états $x_1^T = x_1 \dots x_T$ avec des probabilités différentes. Ainsi, la probabilité $\Pr(\mathbf{o}_1^T \mid \Lambda)$ d'observer la séquence $\mathbf{o}_1^T$ est, formellement donnée par la somme sur toutes les séquences d'états possibles de la probabilité jointe $\Pr(\mathbf{o}_1^T, x_1^T \mid \Lambda)$.

$$\Pr(\mathbf{o}_1^T \mid \Lambda) = \sum_{\{x_1 \dots x_T\}} \Pr(\mathbf{o}_1^T, x_1^T \mid \Lambda) \tag{3.23}$$

Or, d'après le théorème de Bayes et les dépendances conditionnelles illustrées sur la figure 3.34 :

$$\Pr(\mathbf{o}_1^T, x_1^T \mid \Lambda) = \Pr(\mathbf{o}_1^T \mid x_1^T, \Lambda) \Pr(x_1^T \mid \Lambda)$$

Ainsi, d'une part on a :

$$\begin{aligned} \Pr(\mathbf{o}_1^T \mid x_1^T, \Lambda) &= \prod_{t=1}^{T} \Pr(\mathbf{o}_t \mid x_t, \Lambda) \\ &= b_{x_1}(\mathbf{o}_1) b_{x_2}(\mathbf{o}_2) \cdots b_{x_T}(\mathbf{o}_T) \end{aligned} \tag{3.24}$$

Et d'autre part, on a :

$$\Pr(x_1^T \mid \Lambda) = \pi_{x_1} a_{x_1 x_2} \cdots a_{x_{T-1} x_T} \tag{3.25}$$

L'expression 3.23 est une somme de T-uplets (de l'ordre de N^T) et les expressions 3.24 et 3.25 sont des suites de T produits. Ainsi, le calcul exact de $\Pr(\mathbf{o}_1^T \mid \Lambda)$ est de l'ordre de $2TN^T$, ce qui est très coûteux.

Cependant, il existe une méthode d'estimation moins coûteuse (de l'ordre de $(2N+1)\,N\,(T-1)$) : l'algorithme **Forward-Backward**. Cet algorithme comporte deux étapes. L'étape **Forward** nous permet de calculer $Pr(\mathbf{o}_1^T \mid \Lambda)$ et l'étape **Backward** n'est utilisée que pour la réestimation des paramètres Λ du modèle. Cependant, en pratique, elles sont mises en œuvre en même temps.

On peut écrire $Pr(\mathbf{o}_1^T \mid \Lambda)$ sous la forme suivante :

$$Pr(\mathbf{o}_1^T \mid \Lambda) \;=\; \sum_{j=1}^{N} Pr(\mathbf{o}_1^T, x_T = j \mid \Lambda) \tag{3.26}$$

On pose la variable $\alpha_{j,t}$ telle que :

$$\alpha_{j,t} \;=\; Pr(\mathbf{o}_1^t, x_t = j \mid \Lambda)$$

Ainsi, on obtient (détails en annexe C):

$$\boxed{\alpha_{j,t} \;=\; b_j(\mathbf{o}_t) \sum_{i=1}^{N} a_{ij}\,\alpha_{i,t-1}} \tag{3.27}$$

Finalement, les équations (3.26) et (3.27) nous permettent d'écrire :

$$\boxed{Pr(\mathbf{o}_1^T \mid \Lambda) \;=\; \sum_{j=1}^{N} \alpha_{j,T}} \tag{3.28}$$

Concrètement, $\alpha_{j,t}$ représente la probabilité d'observation partielle de la séquence s'achevant dans l'état j au temps t (Figure 3.35). On calcule cette variable à l'aide de l'algorithme suivant :

Algorithme Forward

- Initialisation : $\alpha_{i,1} = \pi_i\, b_i(\mathbf{o}_1)\ \forall i, i \in \{1 \ldots N\}$
- Récursion : $\alpha_{j,t+1} = b_j(\mathbf{o}_{t+1})\,[\sum_{i=1}^{N} \alpha_{i,t}\, a_{ij}]$
  $\forall j, j \in \{1 \ldots N\}$ et $\forall t, t \in \{1 \ldots T-1\}$
- Terminaison : $\Pr(\mathbf{o}_1^T \mid \Lambda) = \sum_{i=1}^{N} \alpha_{i,T}\ \forall i, i \in \{1 \ldots N\}$

A présent, on cherche à ajuster Λ de sorte à maximiser $Pr(\mathbf{o}_1^T \mid \Lambda)$, de manière à modéliser des séquences d'observations.

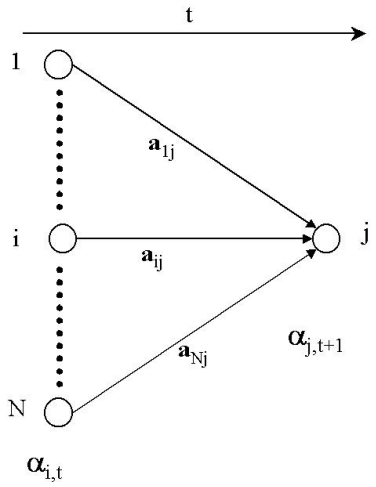


Figure 3.35: Illustration du calcul de la variable α.

3.2.4 Maximiser la vraisemblance

La procédure d'apprentissage la plus fréquemment utilisée pour les modèles de Markov cachés est basée sur le MLE (Maximum Likelihood Estimation). On cherche à maximiser la vraisemblance $L(\Lambda \mid \mathcal{O})$ (Equation 3.29) du modèle Λ sur l'ensemble des observations $\mathcal{O}$. Pour cela, nous allons chercher à ajuster les paramètres Λ du modèle pour maximiser $\Pr(\mathcal{O} \mid \Lambda)$. Nous utilisons une procédure itérative: l'algorithme **EM** [Dempster et al., 1977]. Nous trouvons des formules de réestimation des paramètres similaires à celles proposées dans la littérature [Rabiner, 1989] par l'algorithme de **Baum-Welch**.

$$L(\Lambda \mid \mathcal{O}) \quad = \quad \prod_{p=1}^{P} Pr(\mathbf{o}_1^{T_p}(p) \mid \Lambda) \tag{3.29}$$

a. Introduction à l'algorithme EM

L'algorithme **EM** (Expectation Maximisation) [Dempster et al., 1977] cherche à maximiser la fonction de log-vraisemblance (Equation 3.30) sur les paramètres Θ d'un modèle, l'ensemble des données $\mathcal{D}$ étant donné.

$$l(\Theta \mid \mathcal{D}) = log\, L(\Theta \mid \mathcal{D}) \tag{3.30}$$

Ce problème est complexe. En effet, pour certains problèmes, il n'y a pas, *a priori*, de lien connu entre les données et les paramètres. Pour simplifier ce problème, l'hypothèse EM est d'introduire un nouvel ensemble de variables $\mathcal{H}$ dites cachées. Ainsi, nous obtenons un nouvel ensemble de données $\mathcal{D}_c = (\mathcal{D}, \mathcal{H})$, appelé l'ensemble complet des données, de fonction de log-vraisemblance $l_c(\Theta \mid \mathcal{D}_c)$. Il a été démontré que maximiser la log-vraisemblance complète $l_c(\Theta \mid \mathcal{D}_c)$ maximise également $l(\Theta \mid \mathcal{D})$ et donc par conséquent maximise la vraisemblance $L(\Theta \mid \mathcal{D})$.

Cependant, cette fonction ne peut pas être maximisée directement car $\mathcal{H}$ est inconnu. Il a été déjà démontré [Dempster et al., 1977] que l'estimation itérative de la fonction auxiliaire Q (Equation 3.31) maximise $l_c(\Theta \mid \mathcal{D}_c)$ en utilisant les paramètres $\hat{\Theta}$ de l'itération précédente (cf. Annexe B).

$$Q(\Theta, \hat{\Theta}) = E_{\mathcal{H}}[l_c(\Theta \mid \mathcal{D}_c) \mid \mathcal{D}, \hat{\Theta}] \tag{3.31}$$

L'algorithme EM est le suivant :

- Pour $k = 1 \ldots K$, où K est le nombre maximum d'itérations EM
  - Etape **E** : calcul de $Q(\Theta, \Theta^{(k-1)}) = E_{\mathcal{H}}[l_c(\Theta \mid \mathcal{D}_c) \mid \mathcal{D}, \Theta^{(k-1)}]$
  - Etape **M** : $\Theta^{(k)} = \arg\max_{\Theta} Q(\Theta, \Theta^{(k-1)})$

Le calcul analytique de Q n'est pas toujours possible. Dans ce cas, on tente d'estimer les paramètres Θ indirectement, on parle alors d'algorithme GEM (Generalized Expectation Maximisation). Il a été démontré qu'il est possible de garantir la convergence vers un maximum local de la fonction de vraisemblance en réalisant une mise à jour des paramètres $\Theta^{k+1} = M(\Theta^k)$ où la fonction $M(.)$ est telle que $Q(M(\Theta^k), \Theta^k) \geq Q(\Theta^k, \Theta^k)$ [Dempster et al., 1977] [Baum, 1972].

b. Application de l'algorithme EM à un HMM

Pour compléter l'ensemble des données $\mathcal{O}$, on suppose qu'il existe un ensemble de variables $\mathcal{X}$ inconnues. Ces variables correspondent aux séquences d'états. La vraisemblance complète $L_c(\Lambda \mid \mathcal{O}, \mathcal{X})$ est donnée par :

$$\begin{aligned}
L_c(\Lambda \mid \mathcal{O}, \mathcal{X}) &= \prod_{p=1}^{P} Pr(\mathbf{o}_1^{T_p}(p), x_1^{T_p}(p) \mid \Lambda) \\
&= \prod_{p=1}^{P} Pr(\mathbf{o}_{T_p}(p), x_{T_p}(p) \mid \mathbf{o}_1^{T_p-1}(p), x_1^{T_p-1}(p), \Lambda) \\
&\quad Pr(\mathbf{o}_1^{T_p-1}(p), x_1^{T_p-1}(p) \mid \Lambda) \\
&= \prod_{p=1}^{P} \prod_{t=1}^{T_p} Pr(\mathbf{o}_t(p), x_t(p) \mid x_{t-1}(p), \Lambda)
\end{aligned}$$

Par commodité, on omet la variable p.

$$\begin{aligned}
L_c(\Lambda \mid \mathcal{O}, \mathcal{X}) &= \prod_{p=1}^{P} \prod_{t=1}^{T} Pr(\mathbf{o}_t, x_t \mid x_{t-1}, \Lambda) \\
&= \prod_{p=1}^{P} \prod_{t=1}^{T} Pr(\mathbf{o}_t \mid x_t, x_{t-1}, \Lambda)\, Pr(x_t \mid x_{t-1}, \Lambda) \\
&= \prod_{p=1}^{P} \prod_{t=1}^{T} Pr(\mathbf{o}_t \mid x_t, \Lambda)\, Pr(x_t \mid x_{t-1}, \Lambda)
\end{aligned} \tag{3.32}$$

Dans le cas, où $t = 1$ on a :

$$Pr(\mathbf{o}_1 \mid x_1, \Lambda) = \pi_{x_1} b_{x_1}(\mathbf{o}_t) \quad (3.33)$$
$$Pr(x_1 \mid x_0, \Lambda) = Pr(x_1 \mid \Lambda) = \pi_{x_1} \quad (3.34)$$

On introduit la variable $z_{i,t}$ telle que :

$$z_{i,t} = \begin{cases} 1 & : \quad x_t = i \\ 0 & : \quad x_t \neq i \end{cases}$$

Ainsi, on a :

$$E_{\mathcal{X}}[z_{i,t} \mid \mathbf{o}_1^T] = Pr(z_{i,t} \mid \mathbf{o}_1^T) = Pr(x_t = i \mid \mathbf{o}_1^T)$$

On poursuit le calcul de $L_c(\Lambda \mid \mathcal{O}, \mathcal{X})$, l'équation (3.32) devient :

$$L_c(\Lambda \mid \mathcal{O}, \mathcal{X}) = \prod_{p=1}^{P} \prod_{t=1}^{T} \prod_{j=1}^{N} \prod_{i=1}^{N} Pr(\mathbf{o}_t \mid x_t = j, \Lambda)^{z_{j,t}} \, Pr(x_t = j \mid x_{t-1} = i, \Lambda)^{z_{j,t} z_{i,t-1}}$$

A présent, on calcule $log\, L_c(\Lambda \mid \mathcal{O}, \mathcal{X})$:

$$\begin{aligned} log L_c(\Lambda \mid \mathcal{O}, \mathcal{X}) &= \sum_{p=1}^{P} \sum_{t=1}^{T} \sum_{j=1}^{N} \Big[z_{j,t} \, log\, Pr(\mathbf{o}_t \mid x_t = j, \Lambda) \\ &+ \sum_{i=1}^{N} z_{j,t} \, z_{i,t-1} \, log\, Pr(x_t = j \mid x_{t-1} = i, \Lambda) \Big] \end{aligned}$$

c. L'algorithme EM : Etape E

L'algorithme **EM** est itératif. Soit $\hat{\Lambda}$ les paramètres du modèle à l'itération précédente, on recherche les nouveaux paramètres Λ pour l'itération courante qui maximise la vraisemblance. L'étape **E** de l'algorithme **EM** nous dit de calculer la fonction auxiliaire $Q(\Lambda, \hat{\Lambda})$:

$$\begin{aligned} Q(\Lambda, \hat{\Lambda}) &= E_{\mathcal{X}}[log\, L_c(\Lambda \mid \mathcal{O}, \mathcal{X}) \mid \mathcal{O}, \hat{\Lambda}] \\ &= \sum_{p=1}^{P} \sum_{t=1}^{T} \sum_{j=1}^{N} \Big[E_{\mathcal{X}}[z_{j,t} \mid \mathbf{o}_1^T, \hat{\Lambda}] \, log\, Pr(\mathbf{o}_t \mid x_t = j, \Lambda) \\ &+ \sum_{i=1}^{N} E_{\mathcal{X}}[z_{j,t}, z_{i,t-1} \mid \mathbf{o}_1^T, \hat{\Lambda}] \, log\, Pr(x_t = j \mid x_{t-1} = i, \Lambda) \Big] \\ &= \sum_{p=1}^{P} \sum_{t=1}^{T} \sum_{j=1}^{N} \Big[\hat{g}_{j,t} \, log\, Pr(\mathbf{o}_t \mid x_t = j, \Lambda) \\ &+ \sum_{i=1}^{N} \hat{h}_{ij,t} \, log\, Pr(x_t = j \mid x_{t-1} = i, \Lambda) \Big] \end{aligned} \quad (3.35)$$

Les variables $\hat{g}_{j,t}$ et $\hat{h}_{ij,t}$ sont déterminées par les paramètres $\hat{\Lambda}$ du modèle à l'itération précédente. On a :

$$\hat{g}_{j,t} = Pr(x_t = j \mid \mathbf{o}_1^T, \hat{\Lambda}) \quad (3.36)$$

$$\hat{h}_{ij,t} = Pr(x_t = j, x_{t-1} = i \mid \mathbf{o}_1^T, \hat{\Lambda}) \quad (3.37)$$

Pour $t = 1$ on a :

$$\hat{h}_{ij,1} = Pr(x_1 = j \mid \mathbf{o}_1^T, \hat{\Lambda}) = \hat{g}_{j,1} \quad (3.38)$$

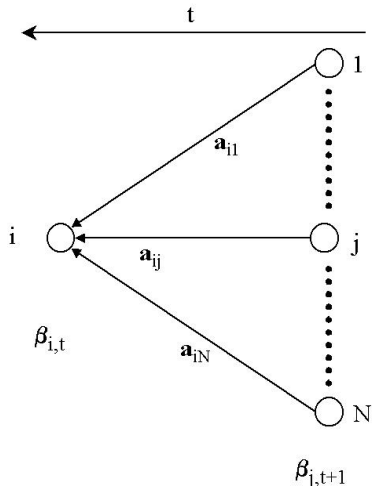


Figure 3.36: Illustration du calcul de la variable β.

On se propose de calculer ces deux variables :

- Calculons $\hat{g}_{j,t}$:

$$\begin{aligned}
\hat{g}_{j,t} &= Pr(x_t = j \mid \mathbf{o}_1^T, \hat{\Lambda}) \\
&= \frac{Pr(x_t = j, \mathbf{o}_1^T \mid \hat{\Lambda})}{Pr(\mathbf{o}_1^T \mid \hat{\Lambda})} \\
&= \frac{Pr(\mathbf{o}_{t+1}^T \mid x_t = j, \mathbf{o}_1^t, \hat{\Lambda})\, Pr(x_t = j, \mathbf{o}_1^t \mid \hat{\Lambda})}{Pr(\mathbf{o}_1^T \mid \hat{\Lambda})} \\
&= \frac{Pr(\mathbf{o}_{t+1}^T \mid x_t = j, \hat{\Lambda})\, \hat{\alpha}_{j,t}}{Pr(\mathbf{o}_1^T \mid \hat{\Lambda})}
\end{aligned}$$

On pose $\hat{\beta}_{j,t} = Pr(\mathbf{o}_{t+1}^T \mid x_t = j, \hat{\Lambda})\ \forall t \in \{1, \ldots, T-1\}$ ainsi :

$$\boxed{\hat{g}_{j,t} = \frac{\hat{\beta}_{j,t}\, \hat{\alpha}_{j,t}}{Pr(\mathbf{o}_1^T \mid \hat{\Lambda})}} \quad (3.39)$$

où $\hat{\beta}_{j,t}$ est déterminé (Annexe C) par :

$$\hat{\beta}_{j,t} = \sum_{i=1}^{N} b_i(\mathbf{o}_{t+1})\, \hat{\beta}_{i,t+1}\, a_{ji} \qquad (3.40)$$

Concrètement, $\beta_{i,t}$ est la probabilité d'observation finale de la séquence commençant au temps t dans l'état i (Figure 3.36). Le calcul des $\beta_{j,t}$ est effectué par l'algorithme **Backward**.

Algorithme Backward

- Initialisation : $\beta_{i,T} = 1,\ \forall i \in \{1 \ldots N\}$
- Récursion : $\beta_{i,t} = \sum_{j=1}^{N} a_{ij}\, b_j(\mathbf{o}_{t+1})\, \beta_{j,t+1}\ \forall i \in \{1 \ldots N\}$ et $\forall t \in \{T-1 \ldots 1\}$

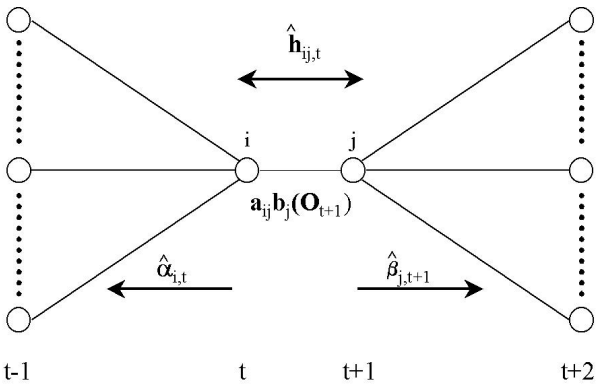


Figure 3.37: Illustration du calcul de la probabilité $\hat{h}_{ij,t}$.

- puis, le calcul de $\hat{h}_{ij,t}$ (Annexe C) nous donne :

$$\hat{h}_{ij,t} = \frac{\hat{\beta}_{j,t}\, b_j(\mathbf{o}_t)\, a_{ij}\, \hat{\alpha}_{i,t-1}}{Pr(\mathbf{o}_1^T \mid \hat{\Lambda})} \qquad (3.41)$$

La variable $\hat{h}_{ij,t}$ correspond à la probabilité d'être dans l'état i à l'instant t et dans l'état j à l'instant $t+1$ avec la connaissance de toute la séquence d'observations $\mathbf{o}_1^T$ (Figure 3.37). L'estimation de cette variable est essentielle dans la suite de l'algorithme. Elle va nous permettre de calculer les nouvelles probabilités de transitions.

d. L'algorithme EM : Etape M

On calcule les nouveaux paramètres de Λ pour maximiser $Q(\Lambda, \hat{\Lambda})$ en calculant les dérivées partielles $\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial \Lambda}$ sous les contraintes suivantes :

- La somme des probabilités initiales vaut 1 :

$$\sum_{i=1}^{N} \pi_i = 1 \tag{3.42}$$

- La somme des probabilités de transitions depuis un état vers tous les autres vaut 1 :

$$\sum_{k=1}^{N} a_{ik} = 1, \forall i \in \{1 \dots N\} \tag{3.43}$$

- Contraintes sur les probabilités d'observations :
  - Dans le cas où les observations sont discrètes : la somme des probabilités d'observations de tous les symboles depuis un état vaut 1.

$$\sum_{d=1}^{N} b_j(d) = 1, \forall j \in \{1 \dots N\} \tag{3.44}$$

  - Dans le cas où les observations sont continues et représentées par des mélanges : la somme des coefficients de toutes les **pdf** d'un état vaut 1.

$$\sum_{k=1}^{M} c_{jk} = 1, \forall j \in \{1 \dots N\} \tag{3.45}$$

Les dérivées partielles à calculer sont : $\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial \pi_i}$, $\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial a_{ij}}$ et $\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial b_j(k)}$. Les détails des calculs sont en annexe C. On obtient les formules de réestimation suivantes :

- Commençons par π_j :

$$\boxed{\pi_j = \frac{\sum_p^P \hat{g}_{j,1}}{\sum_{i=1}^N \sum_p^P \hat{g}_{i,1}}} \tag{3.46}$$

- Continuons par a_{ij} :

$$\boxed{a_{ij} = \frac{\sum_p^P \sum_t^T \hat{h}_{ij,t}}{\sum_{k=1}^N \sum_p^P \sum_t^T \hat{h}_{ik,t}}} \tag{3.47}$$

- Dans le cas où les observations sont discrètes :

$$\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial b_j(k)} = \sum_p^P \sum_t^T \frac{\hat{g}_{j,t}}{b_j(k)} \delta_{\mathbf{o}_t k}$$

où δ_{ij} est le symbole de Kronecker :

$$\delta_{ij} = \begin{cases} 1 & : \quad i = j \\ 0 & : \quad i \neq j \end{cases}$$

On obtient alors :

$$\boxed{b_j(k) = \frac{\sum_p^P \sum_t^T \hat{g}_{j,t} \delta_{\mathbf{o}_t k}}{\sum_{d=1}^D \sum_p^P \sum_t^T \hat{g}_{j,t} \delta_{\mathbf{o}_t d}}} \qquad (3.48)$$

- Poursuivons dans le cas où les observations sont continues, on suppose que les **pdf** sont des gaussiennes. On recherche les paramètres c_{jm}, $\boldsymbol{\mu}_{jm}$ et $\boldsymbol{\sigma}_{jm}$ de Λ de sorte à maximiser la fonction auxiliaire $Q(\Lambda,\hat{\Lambda})$. Pour cela, on calcule les dérivées partielles $\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial c_{jm}}$, $\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial \boldsymbol{\mu}_{jm}}$ et $\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial \boldsymbol{\sigma}_{jm}}$.

On obtient, tout d'abord, pour les coefficients des mélanges :

$$\boxed{c_{jm} = \frac{\sum_p^P \sum_t^T \frac{\hat{g}_{j,t}}{b_j(\mathbf{o}_t)} \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm})}{\sum_{k=1}^K \sum_p^P \sum_t^T \frac{\hat{g}_{j,t}}{b_j(\mathbf{o}_t)} \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jk}, \boldsymbol{\sigma}_{jk})}} \qquad (3.49)$$

puis :

$$\boxed{\boldsymbol{\mu}_{jm} = \frac{\sum_{p=1}^P \sum_{t=1}^T E_{jm}\, \mathbf{o}_t}{\sum_{p=1}^P \sum_{t=1}^T E_{jm}}} \qquad (3.50)$$

et enfin :

$$\boxed{\boldsymbol{\sigma}_{jm} = \left(\frac{\sum_{p=1}^P \sum_{t=1}^T E_{jm}\, (\mathbf{o}_t - \boldsymbol{\mu}_{jm})^T (\mathbf{o}_t - \boldsymbol{\mu}_{jm})}{\sum_{p=1}^P \sum_{t=1}^T E_{jm}} \right)^{-1}} \qquad (3.51)$$

où :

$$E_{jm} = \frac{\hat{g}_{j,t}\, c_{jm}\, \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm})}{b_j(\mathbf{o}_t)} \tag{3.52}$$

Ainsi, on peut résumer les formules de réestimation des paramètres Λ.

- $\bar{\pi}_i$ est déterminé par l'équation (3.46),
- $\bar{a}_{ij}$ est déterminé par l'équation (3.47),
- $\bar{b}_j(k)$ est déterminé par l'équation (3.48) dans le cas discret, et par les équations (3.49), (3.50) et (3.51) dans le cas continu.

Soit $\bar{\Lambda} = \{\bar{\Pi}, \bar{A}, \bar{B}\}$, on pose $\hat{\Lambda} = \bar{\Lambda}$ et on réitère le processus d'ajustement pour continuer à maximiser $\Pr(\mathcal{O} \mid \Lambda)$. On sait par l'analyse de l'algorithme **EM** que ce processus converge vers le Λ maximisant $Pr(\mathcal{O} \mid \Lambda)$ (cf. Annexe B).

3.2.5 Trouver une séquence d'états x_1^T optimale

Dans certaines applications, il peut être intéressant de trouver la séquence d'états x_1^T optimale associée à une séquence d'observations $\mathbf{o}_1^T$. Il existe plusieurs possibilités pour définir un critère permettant de trouver une séquence d'états optimale.

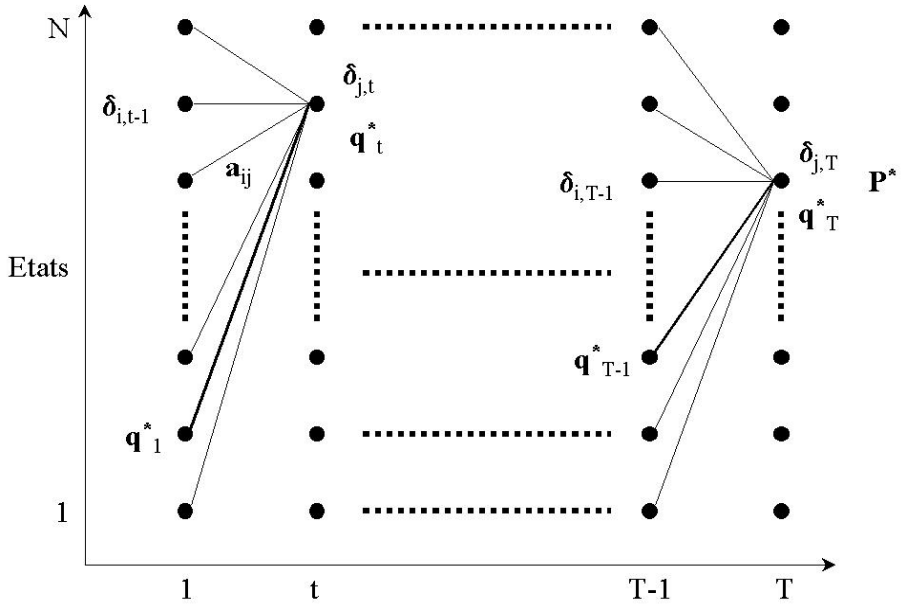


Figure 3.38: Illustration de l'algorithme de **Viterbi**.

a. Critère local

Un premier critère consiste à choisir des états x_t les meilleurs à chaque instant t. Soit $g_{i,t}$ la probabilité d'être à l'état i à l'instant t (Equation 3.36). On choisit $x_t = \arg\max_{1 \leq i \leq N} g_{i,t}$ comme étant l'état optimal à l'instant t. L'inconvénient dans cette méthode est qu'elle ne tient compte que de l'aspect local des probabilités. Une fois le choix d'un état réalisé, on ne peut pas revenir en arrière, on est obligé de continuer pour choisir un nouvel état. Ainsi, on n'a pas la garantie d'obtenir une séquence d'état qui maximise globalement la séquence d'observation.

b. Critère global : l'algorithme de Viterbi

Une autre approche plus globale consiste à trouver la meilleure séquence d'états qui prend en compte toute la séquence observée $\mathbf{o}_1^T$. On utilise alors l'algorithme de **Viterbi**. On recherche un chemin $\psi_{i,t}$ dans le graphe temporel des états du modèle (Figure 3.38) associé à la séquence d'observation partielle $\mathbf{o}_1^t$ et dont on calcule la probabilité $\delta_{i,t}$. On détermine ce chemin de la manière suivante :

- Initialisation : $\forall i \in \{1 \ldots N\}$

$$\begin{aligned} \delta_{i,1} &= \pi_i \, b_i(\mathbf{o}_1) \\ \psi_{i,1} &= 0 \end{aligned}$$

- Récursion : $\forall i \in \{1 \ldots N\}$, $\forall j \in \{1 \ldots N\}$ et $\forall t \in \{2 \ldots T\}$

$$\begin{aligned} \delta_{j,t} &= \left[\max_i \delta_{i,t-1} \, a_{ij} \right] b_j(\mathbf{o}_t) \\ \psi_{j,t} &= \arg\max_i \delta_{i,t-1} \, a_{ij} \end{aligned}$$

- Terminaison : $\forall i \in \{1 \ldots N\}$

$$\begin{aligned} \mathbf{P}^* &= \max_i \delta_{i,T} \\ x_T^* &= \arg\max_i \delta_{i,T} \end{aligned}$$

- Backtrack : $\forall t \in \{T-1 \ldots 1\}$

$$x_t^* = \psi_{x_{t+1}^*, t+1}$$

L'algorithme de Viterbi nous donne $x_1^{T*} = x_1^* \ldots x_T^*$, le chemin modélisant le mieux les observations $\mathbf{o}_1^T$ avec une probabilité $\mathbf{P}^*$.

3.2.6 Autres aspects des modèles de Markov cachés

a. Choix d'une topologie

Les nombreuses études dans le domaine des modèles de Markov cachés ont montré qu'il est nécessaire d'imposer une topologie au graphe des états. Cette topologie a pour motivation de mieux contrôler le nombre de paramètres libres (cf. la notion de généralisation) et d'injecter dans le modèle de la connaissance *a priori* sur les données. La topologie gauche-droite (Figure 3.39), par exemple, est principalement utilisée pour des tâches de reconnaissance de séquences de données comme la parole.

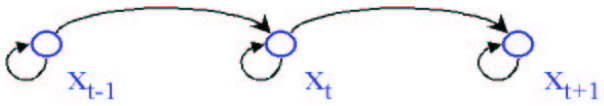


Figure 3.39: Topologie gauche-droite.

L'introduction d'une topologie dans un HMM se traduit par la présence ou l'absence de transitions entre les états. Une fois la topologie choisie, l'algorithme **Forward-Backward** et les formules de réestimation des paramètres ne changent pas.

b. Classifier des séquences d'observations

Soit $\mathcal{C}$ un ensemble de $\mathbf{C}$ classes d'observations distinctes. On dispose de $\mathbf{C}$ ensembles de séquences d'observations $\mathcal{O}_i$ où $i \in \{1, \ldots, \mathbf{C}\}$. On construit $\mathbf{C}$ modèles de Markov cachés de paramètres respectifs $\Lambda_1, \ldots, \Lambda_{\mathbf{C}}$. Puis, un modèle de Markov caché Λ_i est entraîné sur l'ensemble des séquences d'observations $\mathcal{O}_i$.

Soit, à présent, $\mathbf{u}_1^T$ une séquence d'observation appartenant à une des classes de C. L'objectif est de trouver la classe à laquelle appartient la séquence $\mathbf{u}_1^T$. Généralement, la méthode utilisée est la règle de décision de Bayes. Le principe de cette règle est de choisir le meilleur modèle c^* tel que $c^* = \arg\max_{\Lambda_i} Pr(\Lambda_i \mid \mathbf{u}_1^T)$. Cependant, on ne sait pas calculer directement $Pr(\Lambda_i \mid \mathbf{u}_1^T)$ dans un HMM. En revanche, on peut écrire :

$$Pr(\Lambda_i \mid \mathbf{u}_1^T) = Pr(\mathbf{u}_1^T \mid \Lambda_i) \frac{Pr(\Lambda_i)}{Pr(\mathbf{u}_1^T)}$$

Ainsi, $\arg\max_{\Lambda_i} Pr(\Lambda_i \mid \mathbf{u}_1^T) = \arg\max_{\Lambda_i} Pr(\mathbf{u}_1^T \mid \Lambda_i) \frac{Pr(\Lambda_i)}{Pr(\mathbf{u}_1^T)} = \arg\max_{\Lambda_i} Pr(\mathbf{u}_1^T \mid \Lambda_i) Pr(\Lambda_i)$. On peut attribuer une probabilité *a priori* $Pr(\Lambda_i)$ sur des modèles. On utilise alors des méthodes MAP (Maximum *a posteriori*) qui consiste à donner une probabilité *a priori* aux modèles pour ensuite pondérer la probabilité *a posteriori* des données $Pr(\mathbf{u}_1^T \mid \Lambda_i)$ étant donné un modèle. La définition de la probabilité *a priori* peut se baser sur la complexité des modèles. Ainsi, suivant le paradigme du rasoir d'Occam, on privilégiera un modèle simple à un modèle complexe.

Il est également possible de choisir une répartition uniforme des probabilités *a priori* des classes. Si on suppose que toutes les classes sont équiprobables, alors par conséquent $\arg\max_i Pr(\Lambda_i \mid \mathbf{u}_1^T) = \arg\max_i Pr(\mathbf{u}_1^T \mid \Lambda_i)$. Finalement, on choisit le modèle c^* tel que $c^* = \arg\max_i Pr(\mathbf{u}_1^T \mid \Lambda_i)$. Cette méthode est assez efficace et très simple à utiliser, mais elle ne garantit par un taux de reconnaissance optimal car les modèles sont entraînés séparément.

Une autre approche pour augmenter le taux de reconnaissance des HMM est d'utiliser un apprentissage non plus basé sur le MLE mais sur le MMI (Maximum Mutual Information) [Brown, 1987]. Lors de l'apprentissage basé sur le MMI, les paramètres du modèle sont ajustés en prenant en compte la vraisemblance de tous les autres modèles. Il a été montré que l'apprentissage supervisé dans les réseaux de neurones et l'apprentissage basé sur le MMI pour les HMM sont équivalents [Bridle, 1990]. Malheureusement, l'apprentissage pour les HMM basé sur le MMI ne peut se faire que par montée du gradient, ce qui ralentit la convergence.

c. Limite des modèles de Markov cachés

Les HMM sont basés sur des chaînes de Markov homogènes. En effet, la dynamique du système est déterminée uniquement, dans le cas génératif (sans Viterbi), par les probabilités de transitions. Par conséquent, les probabilités d'observations calculées dans un HMM suivent une loi exponentielle de la forme A^t où A est la matrice de transition et t est la longueur de la séquence observée. Cela pose deux problèmes. Tout d'abord, les probabilités d'observations des séquences décroissent très rapidement, ce qui gène les calculs numériques. D'autre part, un HMM fait la supposition forte que les observations sont réalisées à intervalles réguliers.

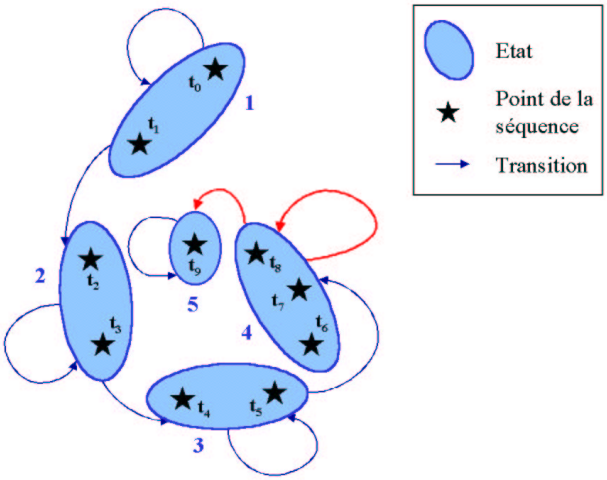


Figure 3.40: Représentation spatiale des états.

Ainsi, par exemple, considérons une séquence de 10 points $t_0, \ldots, t_9$ formant le chiffre 6 et un HMM continu doté d'une topologie gauche-droite à 5 états et d'une gaussienne par état pour représenter les probabilités d'émissions.

Il est possible de représenter les états de ce HMM dans l'espace des observations (Figure 3.40). Les premiers points peuvent se répartir sur les trois états 1, 2 et 3, le dernier point sur l'état 5 et les autres points t_6, t_7 et t_8 peuvent se retrouver dans l'état 4.

On constate que les derniers points sont de plus en proches d'un point de vue spatial. La raison de cela est que la séquence "ralentit". Or dans un HMM, les hypothèses Markoviennes ne permettent pas de prendre en compte ce phénomène. En effet, lorsqu'une nouvelle observation arrive, le système doit nécessairement changer d'état ou soit réaliser une auto-transition. Ainsi, dans notre HMM, la probabilité d'auto-transition de l'état 4 et la probabilité de transition de l'état 4 à l'état 5 seront quasiment identiques. Or, le point t_8, étant sur le "bord" de la gaussienne 4, devrait avoir une probabilité de transition vers l'état 5 plus forte que celle d'auto-transition dans l'état 4. Mais, il n'est pas possible d'attribuer à chaque point une probabilité de transition distincte, car dans un HMM, les probabilités de transitions ne dépendent pas des observations.

Pour pallier ce problème, il est possible de rajouter des états. Mais, si un HMM possède plus d'états, il y a alors plus de paramètres à estimer et par conséquent le modèle devient beaucoup plus sensible au bruit.

d. Choisir le nombre d'états optimal

Ce problème, tout comme celui de l'estimation de la capacité des distributions de transitions et d'émissions, rejoint celui de l'estimation du nombre de neurones cachés dans les réseaux de neurones artificiels. Dans les HMM, le nombre d'état est également primordial. D'un côté, si un HMM possède trop d'états, on estime trop de paramètres et donc, avec peu de données, on risque d'estimer le bruit de ces données. D'un autre côté, si un HMM possède trop peu d'états, le modèle ne capturera pas la dynamique des données. Il est possible de déterminer expérimentalement le nombre d'états optimal soit de manière exhaustive ou soit à l'aide d'un ensemble de validation.

3.3 Les modèles hybrides

3.3.1 Idée générale

Les réseaux de neurones artificiels sont très efficaces pour des problèmes de traitement statique des données, mais ils sont moins bien adaptés lorsque ces données sont de nature temporelle. Par contre, les modèles de Markov cachés ne le sont pas pour des données de nature statique. Il semble donc naturel d'essayer d'associer les réseaux de neurones artificiels et les modèles de Markov cachés pour concevoir des modèles hybrides combinant les propriétés complémentaires des deux modèles. Cette idée n'est pas récente, de nombreux auteurs ont déjà proposé des modèles qui combinent des réseaux de neurones et des modèles de Markov cachés.

3.3.2 Quelques exemples de modèles

On peut distinguer plusieurs catégories. Tout d'abord, celle où les réseaux de neurones sont utilisés comme pré-traitement. Puis, celle où les réseaux de neurones sont utilisés comme post-traitement. Et enfin, celle où les réseaux de neurones servent à calculer les paramètres internes d'un modèle de Markov caché. Dans ce dernier cas, certaines méthodes effectuent l'apprentissage des réseaux de neurones conjointement à l'estimation des paramètres et d'autres méthodes réalisent cet apprentissage séparément, c'est-à-dire à la fin de l'estimation des paramètres.

Les réseaux de neurones sont utilisés comme pré-traitement

[Bengio et al., 1995] proposent un modèle pour la reconnaissance de l'écriture manuscrite. Les auteurs utilisent un réseau de neurones pour reconnaître les informations locales des caractères et se servent d'un HMM pour capturer la dynamique de ces informations. Le HMM est utilisé comme un graphe dont les nœuds possèdent des scores. Ces scores proviennent des réseaux de neurones.

Dans un autre modèle [Bengio et al., 1992], les observations utilisées par le HMM sont issues d'un réseau de neurones récurrent.

Les réseaux de neurones sont utilisés comme post-traitement

Le modèle HMM/MLP proposé par [Cho et Kim, 1995] utilise un réseau de neurones artificiels du type MLP en sortie d'un ensemble de modèles de Markov cachés. Chaque HMM est entraîné classiquement sur un ensemble distinct de séquences d'observations. Le MLP possède autant de sorties que de classes et autant d'entrées que le nombre total d'états de tous les HMM. En chaque état i de chaque HMM j, Cho calcule la probabilité d'observation finale $\alpha_{i,T}(j)$ d'une séquence $\mathbf{o}_1^T$. Pour une séquence, la totalité des $\alpha_{i,T}(j)$ pour tous les états i et tous les HMM j forme un vecteur. Puis, le MLP apprend à classifier ce vecteur pour toutes les séquences. Le modèle HMM/MLP est aussi utilisé pour la reconnaissance de l'écriture manuscrite.

Les réseaux de neurones estiment les paramètres internes

Bourlard utilise un MLP pour estimer les postérieurs $Pr(q \mid x)$, où q est un état et x est une donnée, qu'il transforme ensuite en $Pr(x \mid q)$ par la règle de Bayes et par simplification [Bourlard et Wellekens, 1990] [Bourlard et Morgan, 1993]. Il utilise l'algorithme Viterbi au lieu de l'algorithme EM pour estimer la séquence d'états la plus probable.

[Baldi et Chauvin, 1996] utilisent une architecture hybride HMM/NN pour construire un modèle d'une famille de protéine. Les probabilités d'émission et les probabilités de transitions sont déterminées par des réseaux de neurones. Les auteurs utilisent la maximisation de la vraisemblance et entraînent leur modèle par montée de gradient.

Les "Input/Output Hidden Markov Models" constituent certainement le modèle hybride le plus abouti. Ce modèle se base sur une séquence d'entrée et une chaîne d'états conditionnée par les entrées pour générer une séquence de sortie. Le IOHMM [Bengio, 1995] est capable d'effectuer des tâches de production (conversion d'une séquence d'entrée en une séquence de sortie), de prédiction (les sorties du modèle) et de classification (les sorties sont considérées comme des classes). Il existe même une version asynchrone du modèle qui permet de générer des séquences de sortie de longueur différente de celle d'entrée. De plus, le modèle formel est clairement posé et l'apprentissage peut être réalisé par maximisation de la vraisemblance. L'algorithme EM est appliqué dans toute sa rigueur pour garantir la convergence du modèle. Les IOHMM ont montré leur efficacité pour représenter les séquences avec des dépendances à long terme. Ils ont été utilisés pour la modélisation de grammaires formelles, pour la reconnaissance de la parole et pour la prédiction de séries financières.

CHAPITRE 4

Les Input/Output Hidden Markov Models génératifs

4.1 Les Input/Output Hidden Markov Models

4.1.1 Principe

Ce chapitre décrit une nouvelle approche pour le traitement des séquences de données temporelles basé sur les Input-Output Hidden Markov Models. Les Input-Output Hidden Markov Models (IOHMM) ont été introduits par Bengio et Frasconi [Bengio, 1995] pour des problèmes d'apprentissage mettant en jeu des données de nature séquentielle. Les IOHMM ont des similarités avec les modèles de Markov cachés (Hidden Markov Models), mais permettent d'associer des séquences d'entrée à des séquences de sortie. En effet, dans de nombreux problèmes d'apprentissage, les données sont de nature séquentielle et les réseaux de neurones multi-couches (MLP) ne sont pas naturellement adaptés, à cause du manque d'un mécanisme de mémorisation, pour retenir l'information passée. Certains réseaux de neurones permettent de capturer les relations temporelles en utilisant des délais dans leurs connexions (Time Delay Neural Networks) [Waibel et al., 1989]. Néanmoins, les relations temporelles sont fixées *a priori* par l'architecture du réseau et non pas par les données elles-mêmes qui ont généralement des fenêtres temporelles de taille variable.

Les réseaux de neurones récurrents (RNN) modélisent la dynamique du système en capturant l'information contextuelle d'une observation à l'autre. L'apprentissage supervisé dans les RNN est principalement basé sur des méthodes de descente du gradient : Back-Propagation Through Time [Rumelhart et al., 1986], Real Time Recurrent Learning [Williams et Zipser, 1989] et Local Feedback Recurrent Learning [Mozer, 1989]. Néanmoins, l'apprentissage avec la descente du gradient est difficile lorsque la durée des dépendances temporelles est grande.

De précédents travaux sur des algorithmes d'apprentissage alternatifs [Bengio et Simard, 1994], comme les IOHMM, suggèrent que le problème réside dans la nature essentiellement discrète du processus de stockage de l'information contextuelle pour une durée de temps indéfinie.

Une solution potentielle est de projeter le temps dans un espace discret d'états. Ainsi, le comportement dynamique est décrit par un processus Markovien.

Dans ce cas, à chaque instant et pour chaque variable d'état, l'apprentissage peut se réduire à un problème d'apprentissage statique, c'est-à-dire apprendre les sorties et le prochain état.

Dans un IOHMM, les séquences sont traitées comme dans les réseaux de neurones récurrents. Une séquence d'entrée est transformée de manière synchrone en une séquence de sortie. Un IOHMM peut ainsi réaliser des tâches de production, de prédiction et de classification.

D'autre part dans les IOHMM, les probabilités de transitions sont conditionnées notamment par les entrées et dépendent donc du temps. Il en résulte que les IOHMM sont basés sur des chaînes de Markov non-homogènes alors que celles des HMM sont homogènes. Par conséquent, la dynamique du système n'est plus fixée *a priori*, comme dans les HMM, mais évolue au cours du temps en fonction de la séquence d'entrée.

Nous allons aborder les IOHMM par deux approches. Tout d'abord, nous introduirons les variables du modèle et leurs dépendances conditionnelles. Puis, nous aborderons de manière plus détaillée l'architecture du IOHMM.

4.1.2 Modélisation

Le principe d'un IOHMM est d'aligner une séquence d'entrée avec une séquence d'états discrets, et de générer une séquence de sortie. On dispose donc d'une séquence d'entrée $\mathbf{u}_1^T = \{\mathbf{u}_1, \ldots, \mathbf{u}_T\}$ et d'une séquence de sortie $\mathbf{y}_1^T = \{\mathbf{y}_1, \ldots, \mathbf{y}_T\}$ associée à $\mathbf{u}_1^T$. $\mathbf{u}$ est le vecteur d'entrée de dimension m ($\mathbf{u} \in \mathbb{R}^m$) et $\mathbf{y}$ est le vecteur de sortie de dimension r ($\mathbf{y} \in \mathbb{R}^r$).

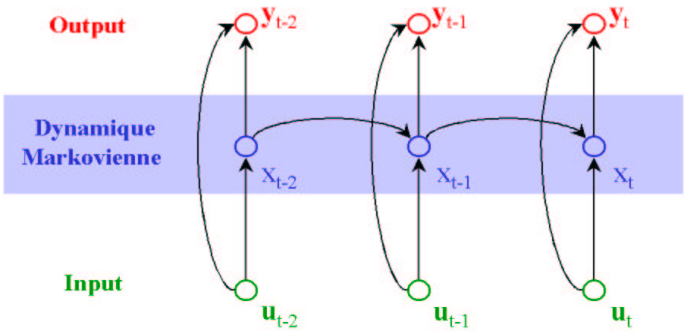


Figure 4.1: Le modèle du IOHMM.

Un IOHMM repose sur une chaîne de n états discrets, où x_t est l'état du modèle à l'instant t (Figure 4.1).

La dynamique du modèle est définie par :

$$\begin{aligned} x_t &= f(x_{t-1}, \mathbf{u}_t) \\ \mathbf{y}_t &= g(x_t, \mathbf{u}_t) \end{aligned} \tag{4.1}$$

x_t est l'état du modèle à l'instant t où $x_t \in X$ avec $X = \{1, \ldots, n\}$ et n est le nombre d'états du modèle. A chaque instant t, le système se trouve dans un état x_t, mais il peut passer dans un nouvel état x_{t+1} à l'instant $t+1$. Le passage d'un état à un autre est conditionné par l'état actuel x_t et par la nouvelle entrée $\mathbf{u}_{t+1}$. De plus, le système génère à chaque instant t une sortie $\mathbf{y}_t$ conditionnée par l'état courant x_t et par l'entrée courante $\mathbf{u}_t$.

Un IOHMM est donc constitué d'un ensemble d'états. A chacun de ces états est associé deux modèles paramétriques à entrée-sortie $\mathcal{N}$ et $\mathcal{O}$. Le modèle paramétrique $\mathcal{N}$ est appelé le réseau d'état et le modèle paramétrique $\mathcal{O}$ est appelé le réseau de sortie. Le vecteur $\mathbf{u}_t$ est l'entrée de chacun des réseaux à l'instant t. Le système met à jour un "historique" qui prend en compte les transitions successives entre les états qui ont eu lieu durant la présentation de la sous-séquence $\mathbf{u}_1^{t-1}$. Puis, cet historique vient contribuer à la génération de la sortie $\mathbf{y}_t$ par les réseaux de sortie.

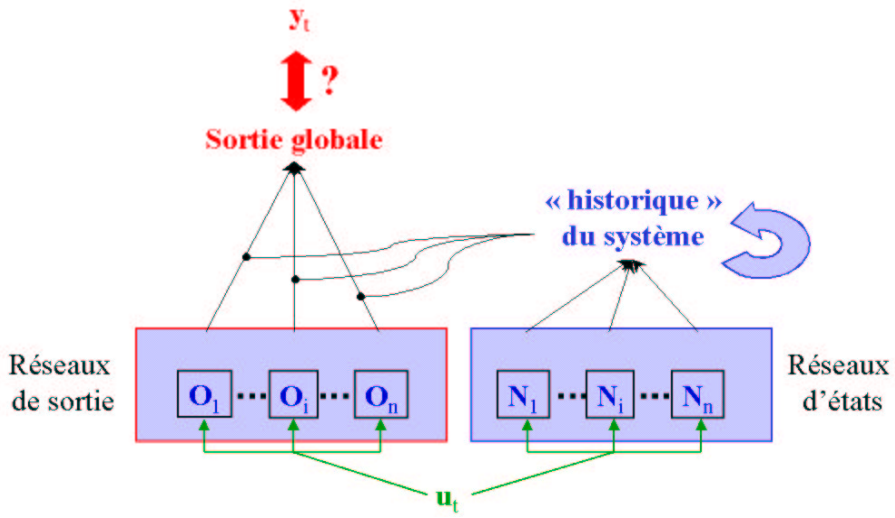


Figure 4.2: Architecture du IOHMM.

Les réseaux sont des modèles paramétriques à entrée-sortie :

- $\boldsymbol{\theta}_j$ est l'ensemble des paramètres du réseau d'état $\mathcal{N}_j$ $(\forall j \in X)$. Un réseau d'état $\mathcal{N}_j$ possède autant de sorties que d'états, et chacune de ces sorties va donner la probabilité de passer à un nouvel état sachant que l'on est dans l'état j. Ainsi, $\boldsymbol{\varphi}_{j,t} = {}^T[\varphi_{1j,t} \ldots \varphi_{nj,t}]$ est la sortie du réseau d'état $\mathcal{N}_j$ à l'intant t, avec la relation $\varphi_{ij,t} = Pr(x_t = i \mid x_{t-1} = j, \mathbf{u}_t)$, i.e. la probabilité de transition depuis l'état j vers l'état i, avec $\sum_{i=1}^{n} \varphi_{ij,t} = 1$.

- $\boldsymbol{\vartheta}_j$ est l'ensemble des paramètres du réseau de sortie $\mathcal{O}_j$ $(\forall j \in X)$, où $\boldsymbol{\eta}_{j,t}$ est la sortie du réseau de sortie $\mathcal{O}_j$ à l'instant t. Dans le modèle, on dispose de valeurs cibles $\mathbf{y}_t$ à chaque instant t et on désire faire converger les sorties des réseaux de sorties vers des valeurs cibles dont la combinaison linéaire par l'historique du système se rapprochera de $\mathbf{y}_t$.

Il est donc nécessaire de définir une mesure de l'erreur, pour l'apprentissage, entre les sorties obtenues et les valeurs cibles. Cette mesure est représentée sous la forme d'une fonction de densité de probabilité (**pdf**) des sorties $f_Y(\mathbf{y}_t;\boldsymbol{\eta}_{i,t})$, i.e. la probabilité $Pr(\mathbf{y}_t \mid x_t = i, \mathbf{u}_t)$ d'avoir la sortie désirée $\mathbf{y}_t$ connaissant le vecteur d'entrée courant $\mathbf{u}_t$ et l'état courant x_t.

Le choix de la **pdf** dépend du problème. L'exponentielle d'une erreur quadratique de la forme $f_Y(\mathbf{y}_t;\boldsymbol{\eta}_{i,t}) = e^{-\frac{1}{2}\sum_{q=1}^{r}(y_{q,t}-\eta_{qi,t})^2}$ convient à des problèmes de classification et de régression où les sorties sont multidimensionnelles ($r > 1$). Néanmoins, pour des problèmes de classification à deux classes (séquence acceptée ou séquence rejetée), il est possible de n'utiliser qu'une seule sortie ($r = 1$) et d'effectuer la supervision à la fin de la séquence. Ainsi, on peut faire l'hypothèse que la **pdf** du modèle est représentée par une loi de Bernoulli $f_Y(y_T;\eta_{i,T}) = \eta_T^{y_T}(1-\eta_T)^{1-y_T}$. Si $y_T = 1$ la séquence est acceptée, et si $y_T = 0$ la séquence est rejetée. Cependant, pour rester dans un cadre général (classification et régression), nous supposerons dans la suite de ce chapitre que la **pdf** est l'exponentielle d'une erreur quadratique.

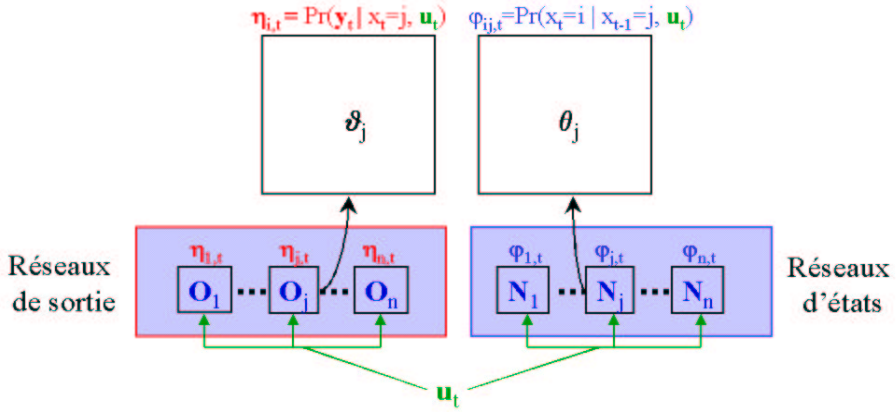


Figure 4.3: Les réseaux de sortie et les réseaux d'états du IOHMM.

Le plus souvent, il est nécessaire de donner aux IOHMM une topologie caractéristique. Cette topologie définit des relations dans le graphe des états pour autoriser ou interdire les transitions. Le choix d'utiliser une topologie va trouver son importance dans l'estimation des probabilités de transitions lors de l'apprentissage. Ainsi, chaque état i admet un ensemble S_i d'états successeurs à l'état i, où $S_i \subset X$. De plus, la topologie considère un ensemble d'états finaux $\mathcal{F}$ où $\mathcal{F} \subset X$. On fait alors l'hypothèse que toutes les séquences de $\mathcal{D}$ s'achèvent dans un état final.

Poursuivons la description du modèle :

- $\boldsymbol{\zeta}_t$ représente l'"historique" du système à l'instant t, $\boldsymbol{\zeta}_t \in \mathbb{R}^n$:

$$\begin{aligned}\zeta_{j,t} &= Pr(x_t = j \mid \mathbf{u}_1^t) = \sum_{i=1}^{n} Pr(x_t = j, x_{t-1} = i \mid \mathbf{u}_1^t) \\ &= \sum_{i=1}^{n} Pr(x_t = j \mid x_{t-1} = i, \mathbf{u}_1^t)\, Pr(x_{t-1} = i \mid \mathbf{u}_1^t)\end{aligned}$$

$$= \sum_{i=1}^{n} Pr(x_t = j \mid x_{t-1} = i, \mathbf{u}_t)\, Pr(x_{t-1} = i \mid \mathbf{u}_1^{t-1}) = \sum_{i=1}^{n} \varphi_{ji,t}\, \zeta_{i,t-1}$$

$$\boldsymbol{\zeta}_t = \sum_{j=1}^{n} \zeta_{j,t-1}\, \boldsymbol{\varphi}_{j,t} \text{ avec } t \neq 0 \tag{4.2}$$

où $\zeta_{j,t} = Pr(x_t = j \mid \mathbf{u}_1^t)$. La variable représentant l'"historique" du système $\boldsymbol{\zeta}_t$ à l'instant t est définie par combinaison linéaire de l'ensemble des sorties des réseaux d'états et de l'"historique" du système $\boldsymbol{\zeta}_{t-1}$ à l'instant $t-1$. Cette variable permet d'intégrer dans le temps toutes les probabilités de transitions. Lorsque $t = 1$, la variable $\boldsymbol{\zeta}_0$ n'est pas définie. Elle est donc aléatoirement choisie suivant une loi uniforme en respectant la contrainte $\sum_{j=1}^{n} \zeta_{j,0} = 1$.

- $\boldsymbol{\eta}_t^*$ constitue la sortie globale du système à l'instant t, $\boldsymbol{\eta}_t^* \in \mathbb{R}^r$:

$$\boldsymbol{\eta}_t^* = Pr(\mathbf{y}_t \mid \mathbf{u}_1^t) = \sum_{j=1}^{n} Pr(\mathbf{y}_t, x_t = j \mid \mathbf{u}_1^t)$$

$$= \sum_{j=1}^{n} Pr(\mathbf{y}_t \mid x_t = j, \mathbf{u}_t)\, Pr(x_t = j \mid \mathbf{u}_1^t) = \sum_{j=1}^{n} \boldsymbol{\eta}_{j,t}\, \zeta_{j,t} \tag{4.3}$$

avec la relation $\boldsymbol{\eta}_t^* = Pr(\mathbf{y}_t \mid \mathbf{u}_1^t)$, i.e. la probabilité d'avoir la sortie désirée $\mathbf{y}_t$ connaissant la séquence d'entrée $\mathbf{u}_1^t$. La variable $\boldsymbol{\eta}_t^*$ est définie par la combinaison linéaire de l'ensemble des sorties des réseaux de sorties et de l'"historique" du système $\boldsymbol{\zeta}_t$ à l'instant t.

4.1.3 Similitude des IOHMM avec des mélanges conditionnels d'experts

Les mélanges conditionnels d'experts

Les mélanges conditionnels d'experts ont été introduits par Jacobs et Jordan [Jacobs et al., 1991; Jacobs et Jordan, 1991].

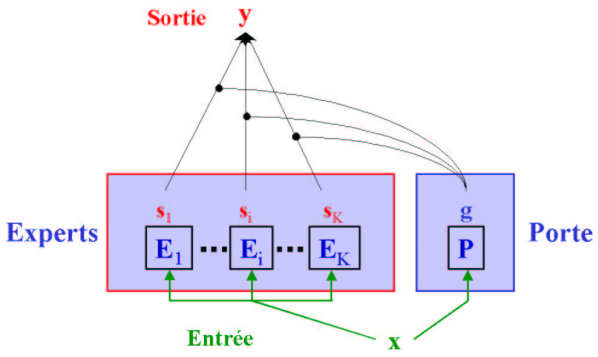


Figure 4.4: Un mélange conditionnel d'experts.

Le principe de cette approche est de décomposer une tâche globale, complexe à résoudre, en une combinaison de sous-tâches plus simples. Un mélange conditionnel de K **experts** (Figure 4.4) est composé d'une série d'estimateurs, par exemple des réseaux de neurones, et d'un réseau **porte** permettant de choisir l'expert approprié à l'entrée x. Chaque expert est une fonction, représentée par un vecteur de paramètres θ_i, qui évalue $E[y|x,\theta_i]$. Le réseau porte $\mathbf{P}$, de paramètre μ, calcule la probabilité qu'un expert $\mathbf{E}_i$ soit approprié en se basant sur la connaissance de l'entrée x. Le réseau porte peut être, par exemple, un MLP. Dans ce cas, on utilise généralement la fonction "softmax" $Pr(\theta_i|x,\mu) = e^{\mathbf{P}_i(x,\mu)} / \sum_{j=1}^{K} e^{\mathbf{P}_j(x,\mu)}$ pour contraindre les sorties $\mathbf{P}_j$ du réseau porte à sommer à 1. $\mathbf{P}(x,\mu)$ est la fonction de x dans $\mathbb{R}^K$ réalisée par le réseau porte. La sortie du mélange conditionnel d'experts est alors $E[y|x,\theta_1,...,\theta_K,\mu] = \sum_{i=1}^{K} Pr(\theta_i|x,\mu)E[y|x,\theta_i]$.

Dans un mélange conditionnel d'experts les données y sont générées par la loi de probabilité suivante : $Pr(y|x,\theta_1^*,...,\theta_K^*,\mu^*) = \sum_{i=1}^{K} Pr(\theta_i|x,\mu^*)Pr(y|x,\theta_i^*)$. L'apprentissage du mélange d'experts consiste à déterminer les paramètres $\theta_1^*,...,\theta_K^*$ et μ^* en utilisant le maximum de vraisemblance où les experts et le réseau porte doivent donc être entraînés simultanément.

IOHMM : un mélange conditionnel d'experts récurrents

Il est possible de concevoir un IOHMM comme une architecture modulaire similaire à un mélange d'experts (Figure 4.2). Les modèles $\mathcal{N}$ et $\mathcal{O}$ peuvent être considérés comme des experts (Figure 4.3).

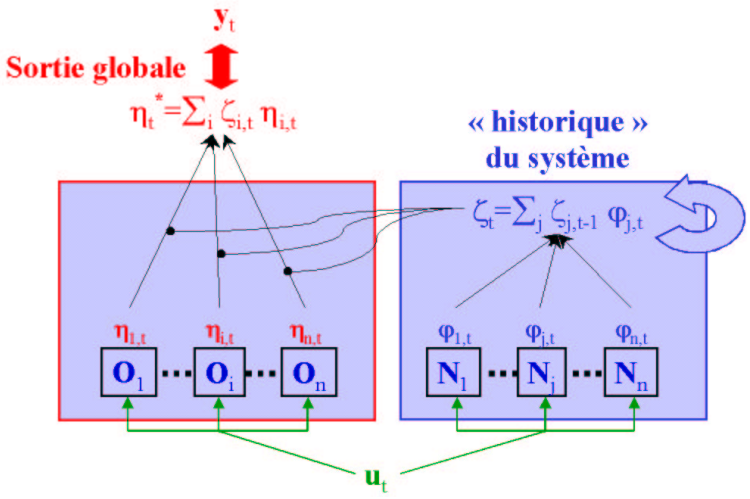


Figure 4.5: La sortie globale du IOHMM.

Les uns travaillent à produire des sorties et les autres estiment la dynamique interne du système. Finalement, cette dynamique vient pondérer chacune des sorties, à l'image de la porte dans les mélanges d'experts, pour produire une sortie unique (Figure 4.5).

4.1.4 Apprentissage

L'apprentissage a déjà été abordé au chapitre 3, où nous avons montré qu'il peut être réalisé par une technique de maximisation de la vraisemblance. Ainsi, nous formulons le problème de l'apprentissage d'un IOHMM comme un problème de maximisation de la vraisemblance de l'ensemble des séquences d'apprentissage sur l'ensemble des paramètres du modèle. La vraisemblance des séquences d'entrée-sortie $\mathcal{D}$ (Equation 4.4) est la probabilité qu'une séquence finie de sorties ait pu être générée par le IOHMM étant donné une séquence finie d'entrées.

L'ensemble $\mathcal{D}$ des séquences d'entrée-sortie $(\mathcal{U}, \mathcal{Y})$ est définie $\forall p \in \{1, \ldots, P\}$ par $\mathcal{D} = (\{\mathbf{u}_1^{T_p}(p)\}, \{\mathbf{y}_1^{T_p}(p)\})$, où P est le nombre de couples de séquences d'entrée-sortie, et où T est la longueur de la séquence observée.

$$\begin{aligned} L(\Theta \mid \mathcal{D}) &= Pr(\mathcal{Y} \mid \mathcal{U}, \Theta) \\ &= \prod_{p=1}^{P} Pr(\mathbf{y}_1^{T_p}(p) \mid \mathbf{u}_1^{T_p}(p), \Theta) \end{aligned} \tag{4.4}$$

Θ est le vecteur de paramètres donné par la concaténation des $\{\boldsymbol{\vartheta}_j\}$ et des $\{\boldsymbol{\theta}_j\}$, $\forall j \in X$. Nous utilisons l'algorithme **EM**, présenté au chapitre 3, pour maximiser la vraisemblance $L(\Theta \mid \mathcal{D})$. Pour cela, on ajoute au modèle des variables cachées données par l'ensemble $\mathcal{X}$ des séquences d'état, $\mathcal{X} = (\{\mathbf{x}_1^{T_p}(p)\})$ avec $p \in \{1, \ldots, P\}$. L'ensemble complet des données est donné par :

$$\begin{aligned} \mathcal{D}_c &= (\mathcal{U}, \mathcal{Y}, \mathcal{X}) \\ &= (\{\mathbf{u}_1^{T_p}(p)\}, \{\mathbf{y}_1^{T_p}(p)\}, \{\mathbf{x}_1^{T_p}(p)\}) \, \forall p \in \{1, \ldots, P\}. \end{aligned}$$

et la vraisemblance sur $\mathcal{D}_c$ est:

$$\begin{aligned} L(\Theta \mid \mathcal{D}_c) &= Pr(\mathcal{Y}, \mathcal{X} \mid \mathcal{U}, \Theta) \\ &= \prod_{p=1}^{P} Pr(\mathbf{y}_1^{T_p}(p), \mathbf{x}_1^{T_p}(p) \mid \mathbf{u}_1^{T_p}(p), \Theta) \end{aligned} \tag{4.5}$$

Par souci de simplifier la notation, nous avons choisi d'omettre la variable p. Ainsi, si nous prenons une séquence, nous pouvons écrire :

$$Pr(\mathbf{y}_1^T, \mathbf{x}_1^T \mid \mathbf{u}_1^T, \Theta) = Pr(\mathbf{y}_T, x_T \mid \mathbf{y}_1^{T-1}, \mathbf{x}_1^{T-1}, \mathbf{u}_1^T, \Theta) \, Pr(\mathbf{y}_1^{T-1}, \mathbf{x}_1^{T-1} \mid \mathbf{u}_1^T, \Theta)$$

De plus, les hypothèses de dépendances entre les variables, en particulier les hypothèses Markoviennes, illustrées dans la figure 4.1 nous permettent d'écrire l'équation ci-dessus comme :

$$Pr(\mathbf{y}_1^T, \mathbf{x}_1^T \mid \mathbf{u}_1^T, \Theta) = Pr(\mathbf{y}_T, x_T \mid x_{T-1}, \mathbf{u}_T, \Theta) \, Pr(\mathbf{y}_1^{T-1}, \mathbf{x}_1^{T-1} \mid \mathbf{u}_1^{T-1}, \Theta)$$

Ainsi, par récurrence on peut écrire l'équation 4.5 comme :

$$\begin{aligned} L(\Theta \mid \mathcal{D}_c) &= Pr(\mathcal{Y}, \mathcal{X} \mid \mathcal{U}, \Theta) \\ &= \prod_{p=1}^{P} \prod_{t=1}^{T} Pr(\mathbf{y}_t, x_t \mid x_{t-1}, \mathbf{u}_t, \Theta) \end{aligned}$$

Introduisons le vecteur $\mathbf{z}_t$ tel que $\mathbf{z}_t \in \{0,1\}^n$:

$$z_{i,t} = \begin{cases} 1 & : \quad x_t = i \\ 0 & : \quad x_t \neq i \end{cases} \quad \forall i \in X$$

On remarque que :

$$\begin{aligned} E[z_{i,t} \mid \mathbf{u}_1^T, \mathbf{y}_1^T] &= Pr(z_{i,t} \mid \mathbf{u}_1^T, \mathbf{y}_1^T) \\ &= Pr(x_t = i \mid \mathbf{u}_1^T, \mathbf{y}_1^T) \end{aligned} \tag{4.6}$$

Ecrivons à nouveau la vraisemblance :

$$\begin{aligned} L(\Theta \mid \mathcal{D}_c) &= \prod_{p=1}^{P} \prod_{t=1}^{T} Pr(\mathbf{y}_t, x_t \mid x_{t-1}, \mathbf{u}_t, \Theta) \\ &= \prod_{p=1}^{P} \prod_{t=1}^{T} Pr(\mathbf{y}_t \mid x_t, \mathbf{u}_t, \Theta)\, Pr(x_t \mid x_{t-1}, \mathbf{u}_t, \Theta) \\ &= \prod_{p=1}^{P} \prod_{t=1}^{T} \prod_{i=1}^{n} \prod_{j=1}^{n} Pr(\mathbf{y}_t \mid x_t = i, \mathbf{u}_t, \Theta)^{z_{i,t}}\, Pr(x_t = i \mid x_{t-1} = j, \mathbf{u}_t, \Theta)^{z_{i,t} z_{j,t-1}} \end{aligned}$$

A présent, calculons la log-vraisemblance :

$$\begin{aligned} l_c(\Theta \mid \mathcal{D}) &= log\, L_c(\Theta \mid \mathcal{D}_c) \\ &= \sum_{p=1}^{P} \sum_{t=1}^{T} \sum_{i=1}^{n} \left(z_{i,t} log\, Pr(\mathbf{y}_t \mid x_t = i, \mathbf{u}_t, \Theta) + \sum_{j=1}^{n} z_{i,t} z_{j,t-1} log\, Pr(x_t = i \mid x_{t-1} = j, \mathbf{u}_t, \Theta) \right) \end{aligned}$$

On ne peut pas maximiser directement $l_c(\Theta \mid \mathcal{D})$ car les séquences d'état $\mathcal{X}$ sont inconnues. On utilise donc l'algorithme EM.

a. Algorithme EM : Etape E

On calcule la fonction auxiliaire Q (Equation 4.7) :

$$\begin{aligned} Q(\Theta, \hat{\Theta}) &= E_{\mathcal{X}}[l_c(\Theta \mid \mathcal{D}_c) \mid \mathcal{U}, \mathcal{Y}, \hat{\Theta}] \\ &= \sum_{p=1}^{P} \sum_{t=1}^{T} \sum_{i=1}^{n} \Big[E[z_{i,t} \mid \mathbf{u}_1^T, \mathbf{y}_1^T, \hat{\Theta}]\, log\, Pr(\mathbf{y}_t \mid x_t = i, \mathbf{u}_t, \Theta) \\ &+ \sum_{j=1}^{n} E[z_{i,t}, z_{j,t-1} \mid \mathbf{u}_1^T, \mathbf{y}_1^T, \hat{\Theta}]\, log\, Pr(x_t = i \mid x_{t-1} = j, \mathbf{u}_t, \Theta) \Big] \\ &= \sum_{p=1}^{P} \sum_{t=1}^{T} \sum_{i=1}^{n} \Big[\hat{g}_{i,t} log\, Pr(\mathbf{y}_t \mid x_t = i, \mathbf{u}_t, \Theta) + \sum_{j=1}^{n} \hat{h}_{ij,t} log\, Pr(x_t = i \mid x_{t-1} = j, \mathbf{u}_t, \Theta) \Big] \\ &= \sum_{p=1}^{P} \sum_{t=1}^{T} \sum_{i=1}^{n} \left(\hat{g}_{i,t} log\, f_Y(\mathbf{y}_t; \boldsymbol{\eta}_{i,t}) + \sum_{j=1}^{n} \hat{h}_{ij,t} log\, \varphi_{ij,t} \right) \end{aligned} \tag{4.7}$$

où $\hat{g}_{i,t}$ et $\hat{h}_{ij,t}$ sont calculés à partir de $\hat{\Theta}$. L'équation 4.6 nous permet d'écrire:

$$\begin{aligned} \hat{g}_{i,t} &= Pr(x_t = i \mid \mathbf{u}_1^T, \mathbf{y}_1^T, \hat{\Theta}) \\ \hat{h}_{ij,t} &= Pr(x_t = i, x_{t-1} = j \mid \mathbf{u}_1^T, \mathbf{y}_1^T, \hat{\Theta}) \end{aligned}$$

Afin de calculer $\hat{g}_{i,t}$ et $\hat{h}_{ij,t}$, on pose:

$$\begin{aligned} \hat{\alpha}_{i,t} &= Pr(\mathbf{y}_1^t, x_t = i \mid \mathbf{u}_1^t) \\ \hat{\beta}_{i,t} &= Pr(\mathbf{y}_{t+1}^T \mid x_t = i, \mathbf{u}_1^T) \\ L &= Pr(\mathbf{y}_1^T \mid \mathbf{u}_1^T) \end{aligned}$$

Alors, on peut exprimer à présent $\hat{g}_{i,t}$ et $\hat{h}_{ij,t}$ en fonction de $\hat{\alpha}_{i,t}$, de $\hat{\beta}_{i,t}$ et de L. Les détails des calculs sont donnés en annexe D.1 et dans [Bengio, 1995].

$$\hat{g}_{i,t} = \frac{\hat{\alpha}_{i,t}\,\hat{\beta}_{i,t}}{L} \tag{4.8}$$

$$\hat{h}_{ij,t} = \frac{\hat{\alpha}_{j,t-1}\,\varphi_{ij,t}\,\hat{\beta}_{i,t}\,f_Y(\mathbf{y}_t;\boldsymbol{\eta}_{i,t})}{L} \tag{4.9}$$

Avec d'une part, $\hat{\alpha}_{i,t}$ qui est donné par récurrence $\forall t = \{1 \ldots T\}$ par :

$$\hat{\alpha}_{i,t} = f_Y(\mathbf{y}_t;\boldsymbol{\eta}_{i,t}) \sum_{j=1}^{n} \varphi_{ij,t}\,\hat{\alpha}_{j,t-1} \tag{4.10}$$

où $\hat{\alpha}_{j,0} = Pr(x_0 = j) = \zeta_{j,0}$.

Et avec d'autre part, $\hat{\beta}_{i,t}$ qui est donné par récurrence $\forall t = \{1 \ldots T-1\}$ par :

$$\hat{\beta}_{i,t} = \sum_{j=1}^{n} f_Y(\mathbf{y}_{t+1};\boldsymbol{\eta}_{j,t+1})\,\hat{\beta}_{j,t+1}\,\varphi_{ji,t+1} \tag{4.11}$$

où $\hat{\beta}_{j,T} = 1$ si $j \in \mathcal{F}$ et $\hat{\beta}_{j,T} = 0$ si $j \notin \mathcal{F}$.

Enfin, L est donné par:

$$L = \sum_{i \in \mathcal{F}} \hat{\alpha}_{i,T}$$

b. Algorithme EM : Etape M

Cette étape consiste à ajuster les paramètres des réseaux d'états et des réseaux de sorties. Ces derniers peuvent être :

- des réseaux non-linéaires comme les réseaux de neurones. La maximisation analytique de Q n'est alors pas possible directement. On utilise donc une méthode basée sur la montée du gradient. Nous n'avons plus affaire à un algorithme **EM**, mais à un algorithme **GEM**.
- des tables de probabilités (Codebooks). La maximisation analytique devient possible. On utilisera, par exemple, les multiplicateurs de Lagrange.

b.1. Les réseaux d'états et de sorties sont des réseaux de neurones

Soit ϑ_{ki} un paramètre (poids) du réseau de neurones de sortie $\mathcal{O}_i$. En dérivant l'équation 4.7, on obtient :

$$\begin{aligned}\frac{\partial Q(\Theta,\hat{\Theta})}{\partial \vartheta_{ki}} &= \sum_{p=1}^{P}\sum_{t=1}^{T}\hat{g}_{i,t}\frac{\partial log\, f_Y(\mathbf{y}_t;\boldsymbol{\eta}_{i,t})}{\partial \vartheta_{ki}} \\ &= \sum_{p=1}^{P}\sum_{t=1}^{T}\hat{g}_{i,t}\sum_{j=1}^{r}\frac{\partial log\, f_Y(\mathbf{y}_t;\boldsymbol{\eta}_{i,t})}{\partial \eta_{ji,t}}\frac{\partial \eta_{ji,t}}{\partial \vartheta_{ki}}\end{aligned}$$

La **pdf** $f_Y(\mathbf{y}_t;\boldsymbol{\eta}_{i,t})$ choisie est $f_Y(\mathbf{y}_t;\boldsymbol{\eta}_{i,t}) = e^{-\frac{1}{2}\sum_{q=1}^{r}(y_{q,t}-\eta_{qi,t})^2}$ (l'exponentielle de l'erreur quadratique). Les dérivées partielles deviennent :

$$\frac{\partial Q(\Theta,\hat{\Theta})}{\partial \vartheta_{ki}} = \sum_{p=1}^{P}\sum_{t=1}^{T}\hat{g}_{i,t}\sum_{j=1}^{r}(y_{j,t}-\eta_{ji,t})\frac{\partial \eta_{ji,t}}{\partial \vartheta_{ki}} \tag{4.12}$$

Ainsi, les dérivées partielles $\frac{\partial \eta_{ji,t}}{\partial \vartheta_{ki}}$ peuvent être calculées par la rétropropagation de $(y_{j,t}-\eta_{ji,t})$ sur chaque neurone de sortie j des réseaux de neurones de sorties $\mathcal{O}_i$. $(y_{j,t}-\eta_{ji,t})$ est la différence entre la sortie désirée $y_{j,t}$ et la sortie obtenue $\eta_{ji,t}$, pour le neurone j du réseau de sortie i et pour l'exemple d'entrée $\mathbf{u}_t$.

Soit θ_{kj} un paramètre (poids) du réseau de neurones d'état $\mathcal{N}_j$. En dérivant l'équation 4.7, on obtient :

$$\frac{\partial Q(\Theta,\hat{\Theta})}{\partial \theta_{kj}} = \sum_{p=1}^{P}\sum_{t=1}^{T}\sum_{i=1}^{n}\hat{h}_{ij,t}\frac{1}{\varphi_{ij,t}}\frac{\partial \varphi_{ij,t}}{\partial \theta_{kj}} \tag{4.13}$$

Les dérivées partielles $\frac{\partial \varphi_{ij,t}}{\partial \theta_{kj}}$ ne peuvent pas être calculées directement par rétropropagation dans les réseaux de neurones d'états $\mathcal{N}_j$. En effet, les $\varphi_{ij,t}$ sont des probabilités de transitions avec la contrainte $\sum_{i=1}^{n}\varphi_{ij,t} = 1\,\forall j \in X$. Or, les sorties

s_{ij} des réseaux de neurones d'états $\mathcal{N}_j$ ne vérifient pas cette contrainte. On utilise donc la fonction "Softmax" (Equation 4.14) pour calculer les $\varphi_{ij,t}$ à partir des s_{ij}.

$$\varphi_{ij,t} = \frac{e^{s_{ij}}}{\sum_{l=1}^{n} e^{s_{lj}}} \tag{4.14}$$

Ainsi, l'équation 4.13 devient :

$$\frac{\partial Q(\Theta,\hat{\Theta})}{\partial \theta_{kj}} = \sum_{p=1}^{P}\sum_{t=1}^{T}\sum_{i=1}^{n}\frac{\hat{h}_{ij,t}}{\varphi_{ij,t}}\sum_{q=1}^{n}\frac{\partial \varphi_{ij,t}}{\partial s_{qj}}\frac{\partial s_{qj}}{\partial \theta_{kj}}$$

A présent, si on calcule la dérivée partielle $\frac{\partial \varphi_{ij,t}}{\partial s_{qj}}$ (détails en annexe D.2.1), deux cas se présentent :

- $q = i$: $\frac{\partial \varphi_{ij,t}}{\partial s_{ij}} = \varphi_{ij,t}\,(1 - \varphi_{ij,t})$
- $q \neq i$: $\frac{\partial \varphi_{ij,t}}{\partial s_{qj}} = -\varphi_{ij,t}\,\varphi_{qj,t}$

Finalement, l'équation 4.13 devient (détails en annexe D.2.2) :

$$\begin{aligned}\frac{\partial Q(\Theta,\hat{\Theta})}{\partial \theta_{kj}} &= \sum_{p=1}^{P}\sum_{t=1}^{T}\sum_{i=1}^{n}\frac{\hat{h}_{ij,t}}{\varphi_{ij,t}}\sum_{q=1}^{n}\frac{\partial \varphi_{ij,t}}{\partial s_{qj}}\frac{\partial s_{qj}}{\partial \theta_{kj}} \\ &= \sum_{p=1}^{P}\sum_{t=1}^{T}\sum_{q=1}^{n}\frac{\partial s_{qj}}{\partial \theta_{kj}}\,(\hat{h}_{qj,t} - \varphi_{qj,t}\sum_{i=1}^{n}\hat{h}_{ij,t})\end{aligned} \tag{4.15}$$

Ainsi, les dérivées partielles $\frac{\partial s_{qj}}{\partial \theta_{kj}}$ peuvent être calculées par rétropropagation de $(\hat{h}_{qj,t} - \varphi_{qj,t}\sum_{i=1}^{n}\hat{h}_{ij,t})$ sur chaque neurone de sortie q des réseaux de neurones d'états $\mathcal{N}_j$. Il faut noter une particularité dans le cas de l'utilisation d'une topologie. En effet, une topologie implique l'absence de certaines transitions, ce qui doit ce traduire par des probabilités de transitions nulles en sortie des réseaux de neurones. Pour éviter aux réseaux de représenter des probabilités nulles, connues *a priori*, on considérera qu'un réseau de neurones d'état i possède autant de sorties que de successeurs S_i.

b.2. Les réseaux d'états et de sorties sont des tables de probabilités

Dans le cas où les séquences sont de nature continue, on réalise une approximation pour se ramener à des données de nature discrète. Ainsi, $\forall u_t$, $u_t \in \{1,\ldots,W\}$ et $\forall y_t$, $y_t \in \{1,\ldots,Z\}$, où W est la dimension du "Codebook" d'entrée et Z est la dimension du "Codebook" de sortie. Ainsi, on pose :

$$\begin{aligned} w_{ijk} &= Pr(x_t = i \mid x_{t-1} = j, u_t = k) \\ &\quad \forall i \in X,\ \forall j \in X,\ \forall k \in \{1,\ldots,W\} \\ z_{qik} &= Pr(y_t = q \mid x_t = i, u_t = k) \\ &\quad \forall i \in X,\ \forall k \in \{1,\ldots,W\},\ \forall q \in \{1,\ldots,Z\} \end{aligned}$$

Les variables w_{ijk} et z_{qik} sont soumises à des contraintes :

$$\sum_{l=1}^{n} w_{ljk} = 1 \; \forall j \in X, \; \forall k \in \{1, \ldots, W\} \tag{4.16}$$

$$\sum_{l=1}^{Z} z_{lik} = 1 \; \forall i \in X, \; \forall k \in \{1, \ldots, Z\} \tag{4.17}$$

Soit w_{ijk} un paramètre de la table de probabilité d'état $\mathcal{N}_j$ pour l'entrée $u_t = k$, dérivons l'équation 4.7. On obtient :

$$\begin{aligned} \frac{\partial Q(\Theta, \hat{\Theta})}{\partial w_{ijk}} &= \sum_{p=1}^{P} \sum_{t=1}^{T} \hat{h}_{ij,t} \frac{\partial \log \, w_{ijk,t}}{\partial w_{ijk}} \\ &= \sum_{p=1}^{P} \sum_{t=1}^{T} \frac{\hat{h}_{ij,t}}{w_{ijk}} \end{aligned}$$

Pour maximiser Q, on utilise les multiplicateurs de Lagrange (Annexe D.2.3). On obtient :

$$\boxed{w_{ijk} = \frac{\sum_p^P \sum_t^T \hat{h}_{ij,t}}{\sum_{l=1}^n \sum_p^P \sum_t^T \hat{h}_{lj,t}}} \tag{4.18}$$

Soit z_{qik} un paramètre de la table de probabilité de sortie $\mathcal{O}_j$ pour l'entrée $u_t = k$, dérivons l'équation 4.7. On obtient :

$$\begin{aligned} \frac{\partial Q(\Theta, \hat{\Theta})}{\partial z_{qik}} &= \sum_{p=1}^{P} \sum_{t=1}^{T} \hat{g}_{i,t} \frac{\partial \log \, z_{qik,t}}{\partial z_{qik}} \\ &= \sum_{p=1}^{P} \sum_{t=1}^{T} \frac{\hat{g}_{i,t}}{z_{qik}} \end{aligned}$$

Pour maximiser Q, on utilise les multiplicateurs de Lagrange (Annexe D.2.4). On trouve :

$$\boxed{z_{qik} = \frac{\sum_p^P \sum_t^T \hat{g}_{i,t} \, \delta_{y_t q}}{\sum_{l=1}^Z \sum_p^P \sum_t^T \hat{g}_{i,t} \, \delta_{y_t l}}} \tag{4.19}$$

où δ_{ij} est le symbole de Kronecker :

$$\delta_{ij} = \begin{cases} 1 & : \quad i = j \\ 0 & : \quad i \neq j \end{cases}$$

c. Algorithme d'apprentissage d'un IOHMM

L'algorithme d'apprentissage du IOHMM est présenté ci-dessous. Cet algorithme itératif résume les équations précédentes et utilise chaque couple de séquence d'entrée-sortie pour calculer des variables intermédiaires (Etape **E**), puis pour maximiser les paramètres des réseaux d'états et des réseaux de sorties (Etape **M**).

Pour chaque couple $(\mathbf{u}_1^{T_p}(p), \mathbf{y}_1^{T_p}(p))$ **de séquences** $\forall p \in \{1, \ldots, P\}$

- Etape **E**
  - Pour chaque couple $(\mathbf{u}_t, \mathbf{y}_t)$ de vecteur de la séquence p $\forall t \in \{1, \ldots, T_P\}$
    - Pour chaque état $j \in X$
      - Calculer $\varphi_{ij,t}$ $\forall i \in S_j$
      - Calculer $\boldsymbol{\eta}_{j,t}$
  - Pour chaque couple $(\mathbf{u}_t, \mathbf{y}_t)$ de vecteur de la séquence p $\forall t \in \{1, \ldots, T_P\}$
    - Pour chaque état $i \in X$
      - Calculer $\hat{\alpha}_{i,t}$ suivant l'équation 4.10
      - Calculer $\hat{\beta}_{i,t}$ suivant l'équation 4.11
  - Pour chaque couple $(\mathbf{u}_t, \mathbf{y}_t)$ de vecteur de la séquence p $\forall t \in \{1, \ldots, T_P\}$
    - Pour chaque état $j \in X$
      - Calculer $\hat{g}_{j,t}$ suivant l'équation 4.8
      - Calculer $\hat{h}_{ij,t}$ $\forall i \in S_j$ suivant l'équation 4.9
- Etape **M**
  - Pour chaque état $j \in X$
    - Ajuster les paramètres $\boldsymbol{\theta}_j$ du réseau d'état $\mathcal{N}_j$
      - Equation 4.15 si $\mathcal{N}_j$ est un réseau de neurones
      - Equation 4.18 si $\mathcal{N}_j$ est une table de probabilités
    - Ajuster les paramètres $\boldsymbol{\vartheta}_j$ du réseau de sortie $\mathcal{O}_j$
      - Equation 4.12 si $\mathcal{O}_j$ est un réseau de neurones
      - Equation 4.19 si $\mathcal{O}_j$ est une table de probabilités

4.1.5 Discussion

Pour des problèmes de classification, on fait la supposition qu'un état final est associé à chaque classe de séquence. Ainsi, on peut considérer que l'on ne s'intéresse à la classe de la séquence qu'une fois qu'un état final a été atteint. Dans ce cas, la distribution de probabilité sur les états finaux $Pr(x_T \mid \mathbf{u}_1^T)$ nous donnera directement la distribution de probabilité sur les classes de sorties $Pr(\mathbf{y} \mid \mathbf{u}_1^T)$.

Par conséquent, on n'a pas forcément besoin d'utiliser des réseaux de sortie pour des problèmes de classification où on associe un état final à une classe. En revanche, pour les problèmes de production ou de prédiction, les réseaux de sortie s'imposent.

Les IOHMM sont plus adaptés que les HMM pour des tâches de reconnaissance. En effet, un apprentissage basé simplement sur la maximisation de la vraisemblance est suffisant pour entraîner efficacement le IOHMM à la discrimination. Il n'est pas nécessaire d'employer les techniques de MMI (cf. Chapitre 3) comme pour les HMM.

Néanmoins, les IOHMM possèdent une faiblesse pour des tâches de reconnaissance. Un IOHMM est capable d'apprendre efficacement à discriminer un certain nombre de classes de séquences. Mais, que se passe-t-il lorsqu'on présente au modèle une séquence qu'il n'a jamais observée ? Formellement, il n'y a rien dans le IOHMM pour nous permettre de rejeter cette séquence. De même, on ne dispose pas d'informations nous indiquant le début ou la fin d'une séquence. Pourtant, ceci est primordial dans le cadre de la reconnaissance en ligne de séquences. Ainsi, considérons une séquence $\mathbf{u}_1^T$ dont la sous-séquence $\mathbf{u}_{t_{debut}}^{t_{fin}}$ a été apprise par un IOHMM. Le IOHMM reçoit à chaque instant un échantillon $\mathbf{u}_t$ de la séquence $\mathbf{u}_1^T$. Ce qui nous préoccupe n'est pas tant de savoir quand la sous-séquence commence, mais plutôt quand la sous-séquence se termine. Dans un IOHMM, lorsque la fin d'une séquence se présente, on sait que la sortie du modèle va nous indiquer la classe de la séquence. En reconnaissance en ligne, il faut donc être capable de détecter l'instant t_{fin}.

4.2 Une approche générative des IOHMM

4.2.1 Motivation

Je propose une extension au modèle des IOHMM par une approche générative dans le cadre du problème de la reconnaissance de séquences de données. Dans ce cadre, ma motivation pour l'élaboration de ce modèle se base sur une limite des IOHMM. Mon objectif est de rejeter les séquences non apprises et d'être capable de détecter l'instant t_{fin}. L'idée principale de l'approche générative est d'apprendre à reconstruire la séquence d'entrée en estimant sa distribution de probabilité.

4.2.2 Modélisation

Nous introduisons, dans le modèle classique du IOHMM, une nouvelle variable $\mathbf{v}_t$ qui va nous permettre de représenter la distribution de probabilité de l'entrée $\mathbf{u}_t$ depuis un état donné. On considère cette variable comme une nouvelle sortie du modèle. Le principe de ce nouveau modèle, appelé GIOHMM (Generative Input-Output Hidden Markov Model), est d'aligner une séquence d'entrée avec une séquence d'états discrets, de générer une séquence de sortie, et de produire une séquence de distribution de probabilité $\mathbf{v}_1^T = \{\mathbf{v}_1, \ldots, \mathbf{v}_T\}$. On dispose donc toujours d'une séquence d'entrée $\mathbf{u}_1^T = \{\mathbf{u}_1, \ldots, \mathbf{u}_T\}$ et d'une séquence de sortie $\mathbf{y}_1^T = \{\mathbf{y}_1, \ldots, \mathbf{y}_T\}$ associée à $\mathbf{u}_1^T$.

On considère que la séquence $\mathbf{v}_1^T$ permet de modéliser la séquence d'entrée $\mathbf{u}_1^T$. $\mathbf{u}$ est le vecteur d'entrée de dimension m ($\mathbf{u} \in \mathbb{R}^m$), $\mathbf{y}$ est le vecteur de sortie de dimension r ($\mathbf{y} \in \mathbb{R}^r$) et $\mathbf{v}$ est une variable discrète ($\mathbf{v} \in \{1 \dots V\}$). Un GIOHMM repose également sur une chaîne de n états discrets, où x_t est l'état du modèle à l'instant t (Figure 4.6).

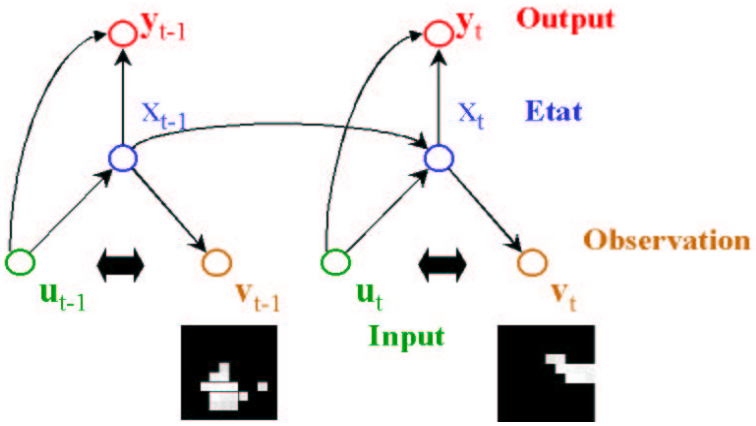


Figure 4.6: Représentation du IOHMM génératif.

La dynamique du modèle est définie à présent par :

$$\begin{aligned} x_t &= f(x_{t-1}, \mathbf{u}_t) \\ \mathbf{y}_t &= g(x_t, \mathbf{u}_t) \\ \mathbf{v}_t &= h(x_t) \end{aligned} \tag{4.20}$$

x_t est l'état du modèle à l'instant t où $x_t \in X$ avec $X = \{1, \dots, n\}$ et n est le nombre d'états du modèle. A présent, le système génère à chaque instant t une sortie $\mathbf{v}_t$ conditionnée par l'état courant x_t avec une probabilité d'émission $d_{x_t}(\mathbf{v}_t)$ (Equation 4.21) et où toutes les émissions l depuis un état i somment à 1 (4.22).

$$d_i(\mathbf{v}_t) = \Pr(\mathbf{v}_t \mid x_t = i, \Theta)\,, \forall \mathbf{v}_t, \mathbf{v}_t \in \{1, \dots, V\} \tag{4.21}$$

$$\sum_{l=1}^{V} d_i(l) = 1\,, \forall i \in X \tag{4.22}$$

4.2.3 Apprentissage

De même que pour les IOHMM, nous formulons le problème de l'apprentissage comme un problème de maximisation de la vraisemblance de l'ensemble des paramètres du modèle sur l'ensemble des séquences d'apprentissage. La vraisemblance des nouvelles séquences d'entrées-sorties $\mathcal{D}$ (Equation 4.23) est la probabilité qu'une séquence finie de sorties ait pu être générée par le GIOHMM étant donné une séquence finie d'entrées.

La particularité du GIOHMM est d'utiliser les entrées $\mathcal{U}$ comme valeurs cibles pour apprendre les sorties $\mathcal{V}$ à générer $\mathcal{U}$. L'ensemble $\mathcal{D}$ des séquences d'entrées-sorties $(\mathcal{U}, \mathcal{Y}, \mathcal{V})$ est défini par $(\{\mathbf{u}_1^{T_p}(p)\}, \{\mathbf{y}_1^{T_p}(p)\}, \{\mathbf{v}_1^{T_p}(p)\})$, $\forall p \in \{1, \ldots, P\}$. P est le nombre de couples de séquences et T est la longueur de la séquence observée.

$$\begin{aligned} L(\Theta \mid \mathcal{D}) &= Pr(\mathcal{Y}, \mathcal{V} \mid \mathcal{U}, \Theta) \\ &= \prod_{p=1}^{P} Pr(\mathbf{y}_1^{T_p}(p), \mathbf{v}_1^{T_p}(p) \mid \mathbf{u}_1^{T_p}(p), \Theta) \end{aligned} \tag{4.23}$$

Θ est le vecteur de paramètres donné par la concaténation des $\{\boldsymbol{\vartheta}_j\}$, des $\{\boldsymbol{\theta}_j\}$, et des $\{\boldsymbol{d}_j\}$ $\forall j \in X$. Nous utilisons l'algorithme **EM** pour estimer le maximum de la vraisemblance $L(\Theta \mid \mathcal{D})$, en utilisant comme variables cachées l'ensemble $\mathcal{X}$ des séquences d'états, $\mathcal{X} = (\{\mathbf{x}_1^{T_p}(p)\})$ avec $p \in \{1, \ldots, P\}$. L'ensemble complet des données est donné par :

$$\begin{aligned} \mathcal{D}_c &= (\mathcal{U}, \mathcal{Y}, \mathcal{V}, \mathcal{X}) \\ &= (\{\mathbf{u}_1^{T_p}(p)\}, \{\mathbf{y}_1^{T_p}(p)\}, \{\mathbf{v}_1^{T_p}(p)\}, \{\mathbf{x}_1^{T_p}(p)\}) \, \forall p \in \{1, \ldots, P\}. \end{aligned}$$

et la vraisemblance sur $\mathcal{D}_c$ est:

$$\begin{aligned} L(\Theta \mid \mathcal{D}_c) &= Pr(\mathcal{Y}, \mathcal{V}, \mathcal{X} \mid \mathcal{U}, \Theta) \\ &= \prod_{p=1}^{P} Pr(\mathbf{y}_1^{T_p}(p), \mathbf{v}_1^{T_p}(p), \mathbf{x}_1^{T_p}(p) \mid \mathbf{u}_1^{T_p}(p), \Theta) \end{aligned} \tag{4.24}$$

Par souci de simplifier la notation, nous avons choisi d'omettre la variable p. Ainsi, si nous prenons une séquence, nous pouvons écrire :

$$\begin{aligned} Pr(\mathbf{y}_1^T, \mathbf{v}_1^T, \mathbf{x}_1^T \mid \mathbf{u}_1^T, \Theta) &= Pr(\mathbf{y}_T, \mathbf{v}_T, x_T \mid \mathbf{y}_1^{T-1}, \mathbf{v}_1^{T-1}, \mathbf{x}_1^{T-1}, \mathbf{u}_1^T, \Theta) \\ &\quad Pr(\mathbf{y}_1^{T-1}, \mathbf{v}_1^{T-1}, \mathbf{x}_1^{T-1} \mid \mathbf{u}_1^T, \Theta) \end{aligned}$$

De plus, les hypothèses de dépendances entre les variables, en particulier les hypothèses Markoviennes, illustrées dans la figure 4.6, nous permettent d'écrire l'équation ci-dessus comme :

$$Pr(\mathbf{y}_1^T, \mathbf{v}_1^T, \mathbf{x}_1^T \mid \mathbf{u}_1^T, \Theta) = Pr(\mathbf{y}_T, \mathbf{v}_T, x_T \mid x_{T-1}, \mathbf{u}_T, \Theta) \, Pr(\mathbf{y}_1^{T-1}, \mathbf{v}_1^{T-1}, \mathbf{x}_1^{T-1} \mid \mathbf{u}_1^{T-1}, \Theta)$$

Ainsi, par récurrence on peut écrire l'équation 4.24 comme :

$$\begin{aligned} L(\Theta \mid \mathcal{D}_c) &= Pr(\mathcal{Y}, \mathcal{V}, \mathcal{X} \mid \mathcal{U}, \Theta) \\ &= \prod_{p=1}^{P} \prod_{t=1}^{T} Pr(\mathbf{y}_t, \mathbf{v}_t, x_t \mid x_{t-1}, \mathbf{u}_t, \Theta) \end{aligned}$$

Introduisons le vecteur $\mathbf{z}_t$ tel que $\mathbf{z}_t \in \{0,1\}^n$:

$$z_{i,t} = \begin{cases} 1 & : \quad x_t = i \\ 0 & : \quad x_t \neq i \end{cases} \quad \forall i \in X$$

On remarque que :

$$E[z_{i,t} \mid \mathbf{u}_1^T, \mathbf{y}_1^T, \mathbf{v}_1^T] \;=\; Pr(z_{i,t} \mid \mathbf{u}_1^T, \mathbf{y}_1^T, \mathbf{v}_1^T) = Pr(x_t = i \mid \mathbf{u}_1^T, \mathbf{y}_1^T, \mathbf{v}_1^T) \quad (4.25)$$

Ecrivons à nouveau la vraisemblance :

$$\begin{aligned}
L(\Theta \mid \mathcal{D}_c) &= \prod_{p=1}^{P} \prod_{t=1}^{T} Pr(\mathbf{y}_t, \mathbf{v}_t, x_t \mid x_{t-1}, \mathbf{u}_t, \Theta) \\
&= \prod_{p=1}^{P} \prod_{t=1}^{T} Pr(\mathbf{y}_t, \mathbf{v}_t \mid x_t, x_{t-1}, \mathbf{u}_t, \Theta)\, Pr(x_t \mid x_{t-1}, \mathbf{u}_t, \Theta) \\
&= \prod_{p=1}^{P} \prod_{t=1}^{T} Pr(\mathbf{y}_t \mid \mathbf{v}_t, x_t, \mathbf{u}_t, \Theta)\, Pr(\mathbf{v}_t \mid x_t, \mathbf{u}_t, \Theta)\, Pr(x_t \mid x_{t-1}, \mathbf{u}_t, \Theta) \\
&= \prod_{p=1}^{P} \prod_{t=1}^{T} Pr(\mathbf{y}_t \mid x_t, \mathbf{u}_t, \Theta)\, Pr(\mathbf{v}_t \mid x_t, \Theta)\, Pr(x_t \mid x_{t-1}, \mathbf{u}_t, \Theta) \\
&= \prod_{p=1}^{P} \prod_{t=1}^{T} \prod_{i=1}^{n} \prod_{j=1}^{n} Pr(\mathbf{y}_t \mid x_t = i, \mathbf{u}_t, \Theta)^{z_{i,t}}\, Pr(\mathbf{v}_t \mid x_t = i, \Theta)^{z_{i,t}} \\
&\quad Pr(x_t = i \mid x_{t-1} = j, \mathbf{u}_t, \Theta)^{z_{i,t} z_{j,t-1}}
\end{aligned}$$

A présent, calculons la log-vraisemblance :

$$\begin{aligned}
l_c(\Theta \mid \mathcal{D}) &= log\, L_c(\Theta \mid \mathcal{D}_c) \\
&= \sum_{p=1}^{P} \sum_{t=1}^{T} \sum_{i=1}^{n} z_{i,t} log\, Pr(\mathbf{y}_t \mid x_t = i, \mathbf{u}_t, \Theta) + z_{i,t} log\, Pr(\mathbf{v}_t \mid x_t = i, \Theta) \\
&+ \sum_{j=1}^{n} z_{i,t} z_{j,t-1} log\, Pr(x_t = i \mid x_{t-1} = j, \mathbf{u}_t, \Theta)
\end{aligned}$$

a. Algorithme EM : Etape E

On ne peut pas maximiser directement $l_c(\Theta \mid \mathcal{D})$ car les séquences d'états $\mathcal{X}$ sont inconnues. On calcule la fonction auxiliaire Q (Equation 4.26) :

$$\begin{aligned}
Q(\Theta, \hat{\Theta}) &= E_{\mathcal{X}}[l_c(\Theta \mid \mathcal{D}_c) \mid \mathcal{U}, \mathcal{Y}, \mathcal{V}, \hat{\Theta}] \\
&= \sum_{p=1}^{P} \sum_{t=1}^{T} \sum_{i=1}^{n} E[z_{i,t} \mid \mathbf{u}_1^T, \mathbf{y}_1^T, \mathbf{v}_1^T, \hat{\Theta}] \Big(log\, Pr(\mathbf{y}_t \mid x_t = i, \mathbf{u}_t, \Theta) + log\, Pr(\mathbf{v}_t \mid x_t = i, \Theta) \Big) \\
&+ \sum_{j=1}^{n} E[z_{i,t}, z_{j,t-1} \mid \mathbf{u}_1^T, \mathbf{y}_1^T, \mathbf{v}_1^T, \hat{\Theta}]\, log\, Pr(x_t = i \mid x_{t-1} = j, \mathbf{u}_t, \Theta)
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{p=1}^{P}\sum_{t=1}^{T}\sum_{i=1}^{n}\hat{g}_{i,t}\Big(log\, Pr(\mathbf{y}_t \mid x_t = i, \mathbf{u}_t, \Theta) + log\, Pr(\mathbf{y}_t \mid x_t = i, \Theta)\Big) \\
&+ \sum_{j=1}^{n}\hat{h}_{ij,t} log\, Pr(x_t = i \mid x_{t-1} = j, \mathbf{u}_t, \Theta) \\
&= \sum_{p=1}^{P}\sum_{t=1}^{T}\sum_{i=1}^{n}\hat{g}_{i,t}\Big(log\, f_Y(\mathbf{y}_t; \boldsymbol{\eta}_{i,t}) + log\, d_i(\mathbf{v}_t)\Big) + \sum_{j=1}^{n}\hat{h}_{ij,t} log\, \varphi_{ij,t} \qquad (4.26)
\end{aligned}
$$

où $\hat{g}_{i,t}$ et $\hat{h}_{ij,t}$ sont calculés à partir de $\hat{\Theta}$. L'équation 4.25 nous permet d'écrire :

$$
\begin{aligned}
\hat{g}_{i,t} &= Pr(x_t = i \mid \mathbf{u}_1^T, \mathbf{y}_1^T, \mathbf{v}_1^T, \Theta) \\
\hat{h}_{ij,t} &= Pr(x_t = i, x_{t-1} = j \mid \mathbf{u}_1^T, \mathbf{y}_1^T, \mathbf{v}_1^T, \Theta)
\end{aligned}
$$

Afin de calculer $\hat{g}_{i,t}$ et $\hat{h}_{ij,t}$, on pose :

$$
\begin{aligned}
\hat{\alpha}_{i,t} &= Pr(\mathbf{y}_1^t, \mathbf{v}_1^t, x_t = i \mid \mathbf{u}_1^t) \\
\hat{\beta}_{i,t} &= Pr(\mathbf{y}_{t+1}^T, \mathbf{v}_{t+1}^T \mid x_t = i, \mathbf{u}_1^T) \\
L &= Pr(\mathbf{y}_1^T, \mathbf{v}_1^T \mid \mathbf{u}_1^T)
\end{aligned}
$$

Alors, on peut exprimer à présent $\hat{g}_{i,t}$ et $\hat{h}_{ij,t}$ en fonction de $\hat{\alpha}_{i,t}$, de $\hat{\beta}_{i,t}$ et de L. Les détails des calculs sont donnés en annexe E.1.

$$
\boxed{\hat{g}_{i,t} = \frac{\hat{\alpha}_{i,t}\,\hat{\beta}_{i,t}}{L}} \qquad (4.27)
$$

$$
\boxed{\hat{h}_{ij,t} = \frac{\hat{\alpha}_{j,t-1}\,\varphi_{ij,t}\,\hat{\beta}_{i,t}\,f_Y(\mathbf{y}_t; \boldsymbol{\eta}_{i,t})\,b_i(\mathbf{v}_t)}{L}} \qquad (4.28)
$$

Avec d'une part, $\hat{\alpha}_{i,t}$ qui est donné par récurrence $\forall t = \{1 \ldots T\}$ par :

$$
\boxed{\hat{\alpha}_{i,t} = f_Y(\mathbf{y}_t; \boldsymbol{\eta}_{i,t})\, b_i(\mathbf{v}_t) \sum_{j=1}^{n}\varphi_{ij,t}\,\hat{\alpha}_{j,t-1}} \qquad (4.29)
$$

où $\hat{\alpha}_{j,0} = Pr(x_0 = j) = \zeta_{j,0}$.

Et avec d'autre part, $\hat{\beta}_{i,t}$ qui est donné par récurrence $\forall t = \{1 \dots T-1\}$ par :

$$\hat{\beta}_{i,t} = \sum_{j=1}^{n} f_Y(\mathbf{y}_{t+1}; \boldsymbol{\eta}_{j,t+1})\, b_i(\mathbf{v}_t)\, \hat{\beta}_{j,t+1}\, \varphi_{ji,t+1} \quad (4.30)$$

où $\hat{\beta}_{j,T} = 1$ si $j \in \mathcal{F}$ et $\hat{\beta}_{j,T} = 0$ si $j \notin \mathcal{F}$.

Enfin, L est donné par:

$$L = \sum_{i \in \mathcal{F}} \hat{\alpha}_{i,T}$$

b. Algorithme EM : Etape M

L'estimation des paramètres des réseaux d'états et des réseaux de sorties est identique à celle des IOHMM. C'est pourquoi, je ne présente ici que l'estimation des nouveaux paramètres génératifs.

Déterminons $\frac{\partial Q(\Theta,\hat{\Theta})}{\partial d_j(k)}$ dans le cas où les sorties $\mathbf{v}_t$ sont discrètes :

$$\frac{\partial Q(\Theta,\hat{\Theta})}{\partial d_j(k)} = \sum_{p}^{P} \sum_{t}^{T} \frac{\hat{g}_{j,t}}{d_j(k)}\, \delta_{\mathbf{v}_t k}$$

Une nouvelle fois, on fait appel aux multiplicateurs de Lagrange. On pose $f(.)$ la fonction $Q(\Theta,\hat{\Theta})$ et $g(.)$ la fonction de contrainte (4.22), alors au maximum de Q :

$$\begin{aligned} & \nabla f = \lambda_j \nabla g \\ \Leftrightarrow \quad & \sum_{p}^{P} \sum_{t}^{T} \frac{\hat{g}_{j,t}}{d_j(k)}\, \delta_{\mathbf{v}_t k} = \lambda_j \frac{\partial(\sum_{l=1}^{V} d_j(l) - 1)}{\partial d_j(k)} \\ \Leftrightarrow \quad & d_j(k) = \frac{\sum_p^P \sum_t^T \hat{g}_{j,t}\, \delta_{\mathbf{v}_t k}}{\lambda_j} \end{aligned} \quad (4.31)$$

Or, on a $\sum_{l=1}^{V} d_j(l) = 1$ (Equation 4.22) d'où:

$$\begin{aligned} & \sum_{l=1}^{V} \frac{\sum_p^P \sum_t^T \hat{g}_{j,t}\, \delta_{\mathbf{v}_t = l}}{\lambda_j} = 1 \\ \Leftrightarrow \quad & \lambda_j = \sum_{l=1}^{V} \sum_{p}^{P} \sum_{t}^{T} \hat{g}_{j,t}\, \delta_{\mathbf{v}_t = l} \end{aligned} \quad (4.32)$$

Ainsi les équations (4.31) et (4.32) nous donnent :

$$d_j(k) = \frac{\sum_p^P \sum_t^T \hat{g}_{j,t}\, \delta_{\mathbf{v}_t=k}}{\sum_{l=1}^V \sum_p^P \sum_t^T \hat{g}_{j,t}\, \delta_{\mathbf{v}_t=l}} \qquad (4.33)$$

Dans cette approche générative, la variable $\mathbf{v}_t$ modélise l'entrée $\mathbf{u}_t$. Cette modélisation est présentée dans le cas discret, mais peut être étendue au cas continu (par exemple par des mélanges de gaussiennes). Ainsi, l'ensemble des paramètres $\mathbf{d}_j$ forme un "Codebook" qui permet de modéliser la distribution de probabilité des entrées. Il est donc possible de calculer la probabilité d'observation des entrées.

4.2.4 Propriétés discriminantes de l'aspect génératif

L'aspect génératif du GIOHMM possède une propriété discriminante envers les observations qui n'ont jamais été rencontrées. Le modèle calcule la probabilité L_t^V d'observation des entrées $\mathbf{u}_1^t$ à l'instant t.

L_t^V est appelée la Vlikelihood et est définie par :

$$\begin{aligned} L_t^V &= Pr(\mathbf{v}_1^t \mid \mathbf{u}_1^t, \Theta) \\ &= \sum_{i \in \mathcal{F}} Pr(\mathbf{v}_1^t, x_t = i \mid \mathbf{u}_1^t, \Theta) \\ &= \sum_{i \in \mathcal{F}} \alpha_{i,t}^V \end{aligned} \qquad (4.34)$$

où :

$$\begin{aligned} \alpha_{i,t}^V &= Pr(\mathbf{v}_1^t, x_t = i \mid \mathbf{u}_1^t, \Theta) \\ &= d_i(\mathbf{v}_t) \sum_{j=1}^n \varphi_{ij,t}\, \alpha_{j,t-1}^V \end{aligned}$$

avec $\alpha_{j,0}^V = \zeta_{j,0}$.

La Vlikelihood L_t^V a la propriété d'être nulle si le modèle n'est pas dans un état final. De plus, une fois dans l'état final, le log de L_t^V décroît tant que le modèle reste dans l'état final.

Il est possible de calculer la valeur minimum de $\log L_T^V$ de toutes les trajectoires de gestes de l'ensemble de test. Cette valeur sert alors de seuil pour rejeter les séquences inconnues. La Vlikelihood est doublement robuste. Tout d'abord, les trajectoires très différentes de celles observées n'activent jamais un état final, ainsi la Vlikelihood est nulle. D'autre part, dans le cas où un élément de la trajectoire (fin identique à une séquence apprise) active l'état final, le seuil permet de rejeter la séquence.

CHAPITRE 5

Reconnaissance des gestes de la main en vision par ordinateur

La détection et l'analyse des personnes est un problème fondamental en vision par ordinateur et plus spécifiquement dans la conception des interfaces gestuelles. Les interfaces gestuelles contribuent à la construction des interfaces homme-machine évoluées. En effet, nous avons suggéré au chapitre 2 que les machines seraient plus faciles à utiliser en associant commande vocale et commande gestuelle, et que l'analyse des gestes basée sur l'image est la voie la plus naturelle pour la construction de ces interfaces homme-machine.

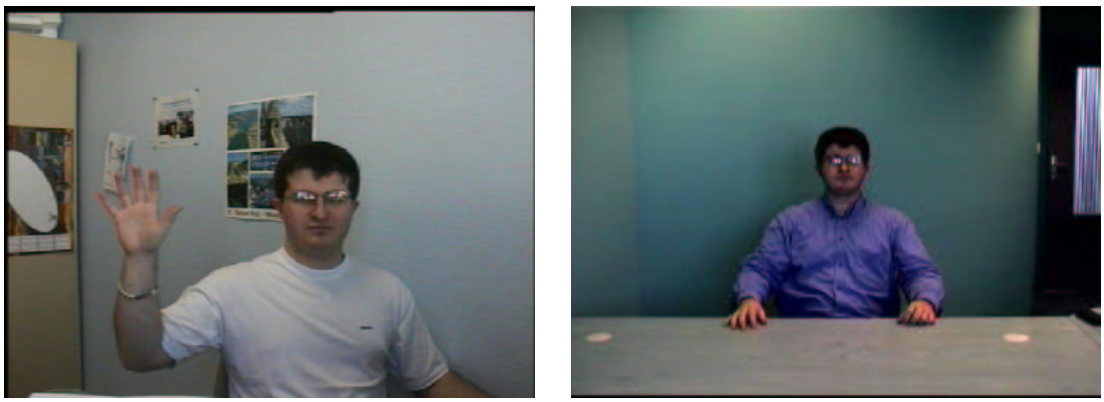


Figure 5.1: Images de type visiophone individuel à analyser.

Afin de libérer au maximum l'utilisateur des contraintes matérielles, nous utilisons les images issues d'une caméra vidéo. La caméra est placée devant l'utilisateur, ce qui nous donne des images de face (Figure 5.1). Les séquences d'images obtenues sont au format CIF (384x288 pixels) et sont capturées dans de bonnes conditions d'éclairage et avec des contraintes sur le fond et sur les personnes. Il est nécessaire, tout d'abord, d'imposer un fond qui ne soit pas d'une couleur similaire à celle de la peau. Toutefois, ce fond peut être complexe ou uniforme. Puis, on suppose la présence d'une seule personne dans le champ de la caméra, et que ce champ est assez large pour voir les mains.

5.1 Problématique

LISTEN (Locating Individual Speakers and Tracking ENvironment) est un système de suivi de personne pour des applications de terminaux individuels et de visioconférence "personne libre" [Collobert et al., 1996], mis au point par l'équipe TNT de France Télécom R&D. Dans le système LISTEN, rappelons-le (cf. Chapitre 1), il n'y a pas de commande explicite. La personne est détectée et suivie en permanence, mais elle ne peut pas commander le système.

L'extension aux interfaces homme-machine est toute naturelle et m'amène donc à m'intéresser à l'intégration de la modalité gestuelle à LISTEN. Je me place dans le cadre d'un terminal individuel. Ainsi, je m'intéresse uniquement à la reconnaissance des gestes d'une personne. Je me base sur la détection du visage de la personne pour repérer le corps. Puis, je recherche les mains, pour reconnaître et interpréter les gestes. Pour réaliser ce travail, je propose l'approche suivante:

1 **Segmentation et suivi de la main**:
Avant de pouvoir reconnaître un geste de la main, il faut pouvoir l'extraire dans l'image, puis la suivre d'une image à l'autre. Cependant, il est difficile de suivre, dans plusieurs images, une main en mouvement. Je décide donc de ramener le problème du suivi à un problème d'appariement de zones. Ainsi, la qualité du suivi est déterminée par la qualité et la stabilité (taille, forme) de ces zones d'une image à l'autre. Tout d'abord, j'utilise une méthode d'extraction de zone rudimentaire. Par la suite, j'étudie des techniques statistiques basées sur des informations spatiales et chromatiques.

2 **Reconnaissance de gestes statiques (postures de main)**:
Comme nous l'avons vu au premier chapitre, les techniques de reconnaissance des gestes statiques (postures) sont diverses. L'équipe TNT possède une grande expérience des réseaux de neurones artificiels et les résultats obtenus sur la détection de visage en utilisant cette technique sont parmi les meilleurs au monde [Féraud et al., 2000]. De plus, les modèles utilisés sont capables de travailler sur de petites images. J'envisage donc d'utiliser des techniques à base de réseaux de neurones artificiels pour effectuer la reconnaissance de la posture de la main (main ouverte, main fermée, index tendu). J'utiliserai les images brutes comme vecteurs d'entrée des réseaux dont l'apprentissage se fera sur une large base d'images d'exemples et de contre-exemples.

3 **Détection des gestes dynamiques locaux (changements de postures)**:
Un autre aspect de mon travail est de pouvoir reconnaître un geste dynamique local. Je considère qu'un tel geste est composé par l'enchaînement de postures de la main. Je pars du principe que je sais détecter une posture de la main dans une image à un instant donné. Cependant, cette méthode est coûteuse en temps de calcul et ne peut pas être utilisée pour chaque image. Je choisis donc de m'orienter vers une technique de détection de changement de posture qui est moins coûteuse. Ainsi, il deviendra possible d'appliquer spécifiquement

la détection de posture. Le problème se ramène donc à celui de la reconnaissance de séquences de caractères, où les caractères sont issus d'un alphabet de postures de la main.

4 **Reconnaissance des gestes dynamiques non-locaux (trajectoires)**: Lors de la reconnaissance des gestes dynamiques non-locaux, je ne prête pas attention à la configuration de la main durant l'exécution du geste. Je cherche plutôt à reconnaître la forme de la trajectoire gestuelle que je considére comme un paramètre du geste.

J'ai récolté de nombreuses séquences d'images représentant différents sujets (hommes et femmes) exécutant un nombre fini de postures et de gestes. Toutes ces images constituent un corpus qui va me permettre de mettre en œuvre les techniques présentées ci-dessus.

5.2 Rappels de traitement d'image

Cette section introduit la couleur[1], puis présente différentes techniques de traitement d'image. Le but de ces techniques est de pré-traiter les images, de réduire le bruit et d'extraire des informations pertinentes. Nous allons utiliser certaines de ces techniques pour reconnaître les gestes de la main.

5.2.1 La couleur

L'œil (Figure 5.2) est l'organe de base du système visuel. Il est en mesure de percevoir seulement une dizaine de niveaux de gris mais par contre 350 000 nuances de couleurs en moyenne. On voit donc que la couleur est une information précieuse. Ainsi, il semble naturel de la prendre en compte dans des systèmes de vision par ordinateur.

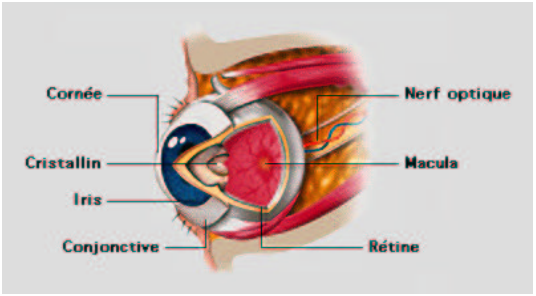


Figure 5.2: L'oeil.

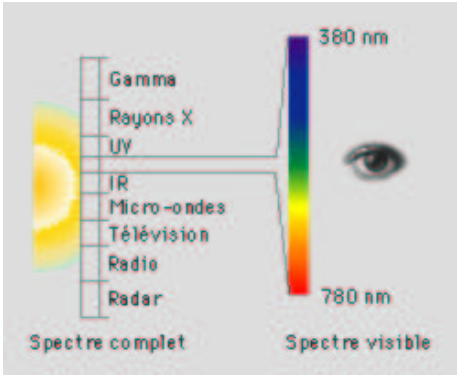


Figure 5.3: Spectre de la lumière.

[1]Toutes les figures de la section suivante viennent du site http://www.erag.ch/MAC/Base.htm

Les couleurs font parties du spectre visible (Figure 5.3) et sont définies par trois paramètres physiques représentés dans le modèle de Munsell (Figure 5.4) :

- la longueur d'onde dominante : elle correspond à la longueur d'onde, exprimée en nanomètres, de la couleur pure qui se rapproche le plus de la couleur considérée. Elle correspond, par abus de langage, au terme de **teinte**.
- le facteur de luminance : il s'agit de l'énergie globale réfléchie par une couleur et exprimée en pourcentage par rapport à un blanc de référence. Il correspond au terme de **luminosité**.
- la pureté d'excitation : elle donne la vivacité de la teinte. Elle s'exprime en pourcentage et correspond au terme de **saturation**.

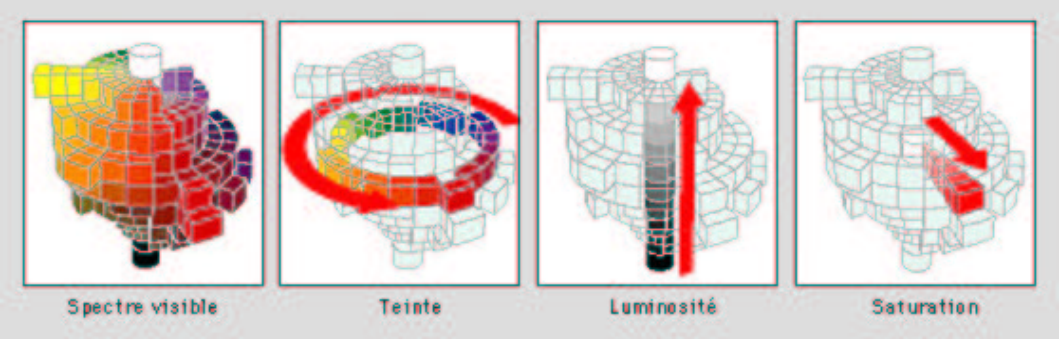


Figure 5.4: Modèle de Munsell des trois paramètres physiques de la couleur.

Généralement, l'aspect coloré d'un objet est déterminé par la teinte et la saturation. On regroupe ces deux paramètres sous le terme de chromaticité.

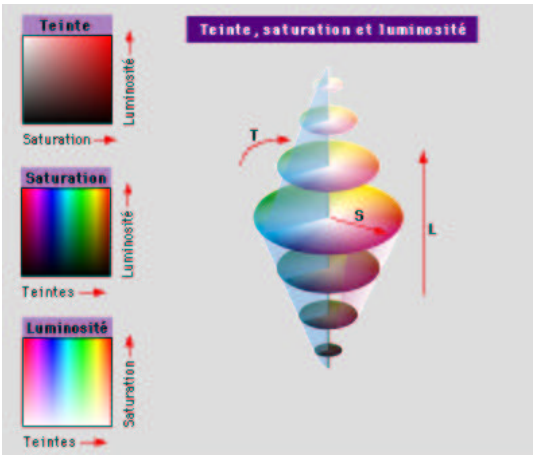


Figure 5.5: Espace TSL.

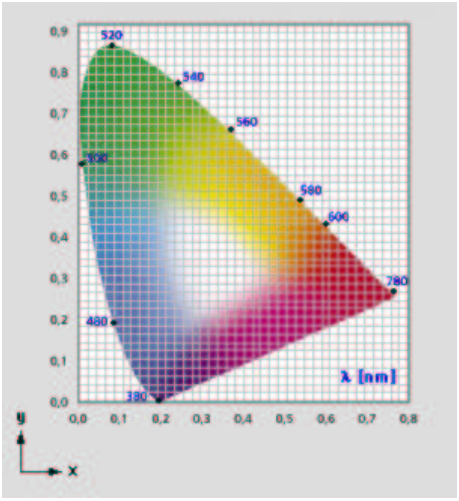


Figure 5.6: Espace de couleur CIE xyY.

En traitement numérique des images, il est nécessaire d'introduire des systèmes de codage de la couleur. Le système TSL (Teinte, Saturation et Luminosité) (Figure 5.5) est une simplification du modèle de la couleur de Munsell. Plus pratique, le diagramme CIE xyY (Figure 5.6) permet de représenter la chromaticité sur un plan (axes x et y). La troisième dimension correspond à la luminosité.

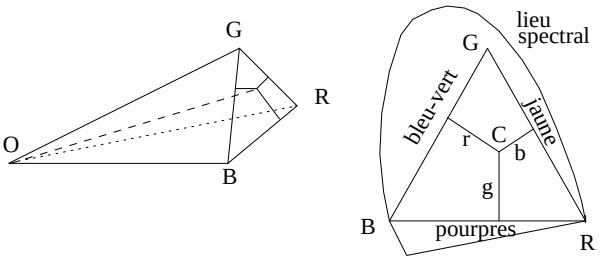


Figure 5.7: Le triangle de Maxwell.

Il a été montré que trois paramètres sont suffisants pour représenter la plupart des couleurs. La convention la plus utilisée est le codage RGB (Red Green Blue). Ainsi, une couleur du lieu spectral xy peut s'exprimer par une relation additive des trois composantes R, G et B. En réalité, à cause des propriétés de cette relation additive, seules les couleurs du triangle de Maxwell (Figure 5.7) peuvent être synthétisées.

5.2.2 Pré-traitements et segmentation

Filtrage des pixels

Le filtrage des pixels constitue une étape importante pour restreindre la quantité d'informations à traiter dans les images. Les pixels sont sélectionnés suivant un critère particulier au problème à traiter, par exemple la teinte chair.

Figure 5.8: Image originale à filtrer.

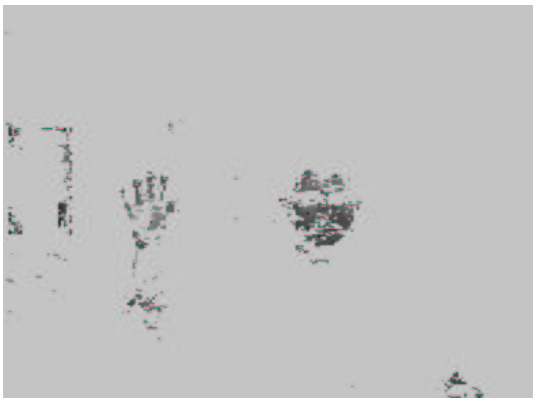


Figure 5.9: Pixels de teinte chair.

Il existe de nombreuses méthodes pour modéliser la teinte chair, principalement à l'aide de gaussiennes [Yang et al., 1998] [Yang et Ahuja, 1999]. De plus, divers espaces colorimétriques peuvent être utilisés. J'ai choisi l'espace colorimétrique YUV,

car c'est le format issu de la caméra et il est relativement robuste au changement d'éclairage grâce à la séparation de la luminosité (Y) et de la chromaticité (UV). L'image est filtrée en utilisant une table de correspondance des pixels de teinte chair (Figure 5.9). Cette table est constituée à l'aide d'une grande collection d'images de teinte chair. Cette technique bien que rudimentaire est efficace et très rapide.

Rehaussage ou égalisation d'histogramme

L'objectif du rehaussage est de normaliser des images en niveau de gris obtenues sous des conditions d'éclairage différentes. Pour cela, on modifie l'histogramme des niveaux de gris, d'où le terme d'égalisation d'histogramme.

Soit $I(x, y)$ une image en niveau de gris de NxM pixels et $h(I, i)$ le nombre d'échantillons de niveau de gris i de l'image $I(x, y)$ où $i \in \{1, \ldots, Q\}$ et Q est le nombre maximum de niveaux de gris.

Figure 5.10: Image avant rehaussage.

Figure 5.11: Image après rehaussage.

Le rehaussage d'une image originale (Figure 5.10) produit une image résultante I' (Figure 5.11) dont la répartition des niveaux de gris est uniforme (Equation 5.1).

$$h(I', i) = \frac{N\,M}{Q} \tag{5.1}$$

L'image résultante est obtenue en inversant l'histogramme cumulé des niveaux de gris de l'image d'origine.

Lissage ou débruitage

Les images peuvent contenir du bruit, soit en provenance du système d'acquisition ou soit causé par des pré-traitements. Le principe du débruitage est d'utiliser un filtre passe-bas pour réduire les hautes fréquences. On utilise, par exemple, une gaussienne centrée en zéro.

$$f(x, y) = e^{-\frac{x^2+y^2}{\sigma^2}}$$

Ce filtre est appliqué à l'image d'origine (Figure 5.12), on parle alors de convolution d'un filtre par une image, pour obtenir une image filtrée (Figure 5.13) :

$$I'(x',y') = I(x,y) \star f(x,y) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} I(x,y) f(x'-x, y'-y)\, dxdy$$

Comme une image est un signal discret, la convolution par un filtre nxn, où n est supposé impair, devient :

$$I'(k,l) = \sum_{j=0}^{n} \sum_{i=0}^{n} I(k - \frac{n-1}{2} + i, l - \frac{n-1}{2} + j) f(i,j)$$

Figure 5.12: Image avant lissage.

Figure 5.13: Image après lissage.

Contours

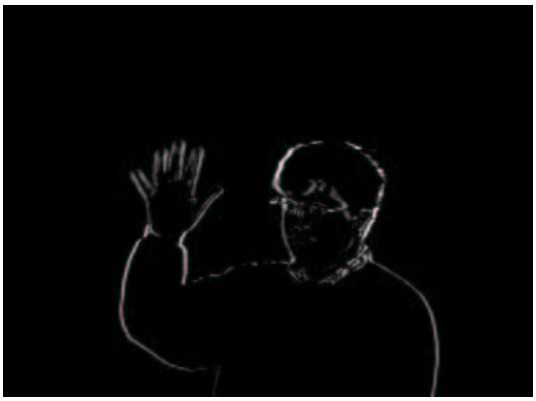
Les contours correspondent à des discontinuités de l'intensité lumineuse dans une image $I(x,y)$. Il existe de nombreuses méthodes pour détecter les contours. La plupart d'entre elles se basent sur le calcul du gradient $\nabla I(x,y)$ de l'intensité lumineuse (Figure 5.14).

Figure 5.14: Les contours.

Cette section n'ayant pas pour vocation de décrire les techniques de détection de contours, j'invite le lecteur à consulter des ouvrages de référence [Binford et Nalwa, 1986] [Canny, 1986] [Shen et Castan, 1986]. Néanmoins, la figure 5.14 permet d'illustrer la difficulté d'obtenir des contours exploitables sur des objets comme la main.

Le mouvement

Dans les séquences d'images, on s'intéresse naturellement au mouvement. En théorie, le mouvement est très complexe. Il est possible, par exemple, de calculer le vecteur directeur de mouvement en chaque point de l'image. En pratique, on réalise le plus souvent une simple différence entre deux images successives I_t et I_{t+1} pour obtenir une image I' du mouvement (Figure 5.15).



$$I' = \begin{cases} 1 & : \mid I_t - I_{t+1} \mid \geq \mu \\ 0 & : \mid I_t - I_{t+1} \mid < \mu \end{cases}$$

où μ est le seuil de détection de mouvement.

Figure 5.15: Le mouvement.

J'ai choisi d'utiliser un certain nombre de ces techniques pour des tâches bien spécifiques :

- le filtrage des pixels de teinte chair permet de délimiter rapidement les parties de l'image à analyser,
- la détection du mouvement permet de concentrer l'analyse sur les parties de l'image qui bougent (visage, mains),
- le rehaussage permet d'augmenter le contraste des imagettes[2] avant de les analyser,
- le lissage est utilisé, à la suite du rehaussage, pour réduire le bruit dans les imagettes.

[2]Les imagettes sont des petites images extraites des images issues de la caméra.

5.3 Segmentation

Le rôle de la segmentation est de construire des zones d'intérêt dans l'image. La construction de ces zones est déterminée par le type des objets que l'on cherche à segmenter. Généralement, les objets importants tels que le visage ou les mains sont détectés à l'aide de la couleur de la peau. Il existe de nombreux modèles pour représenter et segmenter la couleur de la peau, mais encore faut-il regrouper les pixels pour former des régions pertinentes autour des objets et éviter que ces dernières ne fusionnent entre elles ou avec le fond. Les visages et les mains possèdent une teinte caractéristique qu'il est possible de séparer du reste de l'image. J'ai choisi de distinguer deux approches pour la construction de zones : une approche géométrique et une approche statistique.

5.3.1 Approche géométrique

Une première approche pour regrouper les pixels est de tenir compte de leur connexité. Les pixels de teinte chair sont filtrés puis, un algorithme d'expansion permet de déterminer des zones de pixels de teinte chair connexes (Figure 5.16).

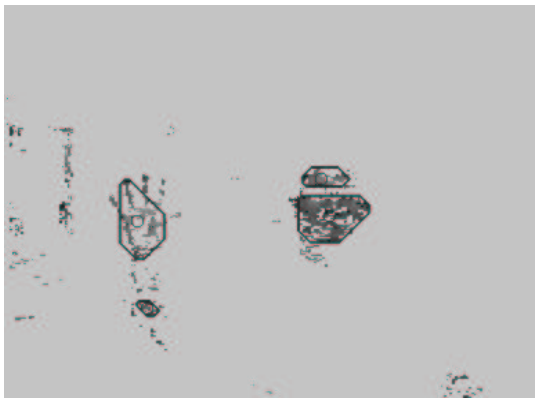


Figure 5.16: Zones connexes de teinte chair à l'image t.

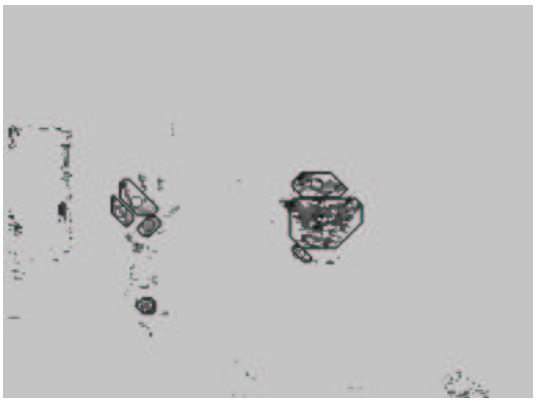


Figure 5.17: Zones connexes de teinte chair à l'image $t+1$.

De plus, on élimine les zones de hauteur ou de largeur inférieures à 10 pixels. Ceci explique pourquoi certaines zones n'apparaissent pas dans le fond. Cependant, le bruit de la caméra et la variation de l'illumination dans la scène rendent ces zones géométriquement instables. Certaines zones apparaissent ou disparaissent, d'autres peuvent changer de taille ou fusionner ensemble (Figures 5.16 et 5.17).

5.3.2 Approche statistique

Une autre approche est de construire des régions homogènes de pixels où les pixels ne sont plus groupés selon un critère de connexité, mais sur un critère de distance. Dans l'espace des paramètres des pixels (position et couleur), ces derniers forment

des amas. Je présente une technique statistique pour regrouper ces amas : les blobs. Les blobs sont définis par un modèle probabiliste gaussien. Plusieurs modèles de blobs ont déjà été utilisés pour la détection des personnes [Wren et al., 1997] ou pour la reconnaissance des gestes de la main [Pavlovic et al., 1997].

La section qui suit pose formellement le modèle des blobs et introduit l'algorithme EM pour adapter les paramètres des blobs aux données de l'image.

5.3.3 Le modèle des blobs

Un blob est défini comme un modèle probabiliste gaussien que l'on cherche à ajuster aux pixels de teinte chair de composantes spatiales (x,y) et colorimétriques (Y,U,V).

Introduisons les notations suivantes :

- $\mathcal{I} = \{\mathbf{x}_i\}$: l'ensemble des pixels de teinte chair de l'image,
  avec $i = 1 \dots P$ et où P est le nombre de pixels de teinte chair dans l'image
- $\mathbf{x}_i = {}^T[x_i, y_i, u_i, v_i, l_i]$
  - x_i, y_i : position du pixel dans l'image,
  - u_i, v_i : chromaticité du pixel dans l'image,
  - l_i : luminosité du pixel dans l'image.
- $\mathcal{B} = (\{\boldsymbol{\mu}_j\}, \{\boldsymbol{\sigma}_j\})$: ensemble des blobs, avec $j = 1 \dots B$ et où B est le nombre de blobs fixé au départ. $\boldsymbol{\mu}_j$ est le vecteur centre des pixels du blob j et $\boldsymbol{\sigma}_j$ est l'inverse de la matrice de covariance des pixels du blob j.

Dans ce modèle, on suppose connu *a priori* le nombre de blobs B. Je recherche les paramètres $\{\boldsymbol{\mu}_j\}$ et $\{\boldsymbol{\sigma}_j\}$ qui permettent d'obtenir les meilleurs blobs étant donné les pixels de teinte chair obtenus par le filtrage de l'image, c'est-à-dire ceux qui maximisent la vraisemblance des pixels observés. Nous recherchons des blobs sur le visage et les mains. Ainsi, le nombre de ces blobs est impossible à estimer. Nous supposerons donc que le nombre de zones sur-estime le nombre de blobs B.

5.3.4 La vraisemblance des blobs

Soit Θ la collection des paramètres $\{\boldsymbol{\mu}_j\}$ et $\{\boldsymbol{\sigma}_j\}$. On peut alors introduire la fonction de vraisemblance L (Equation 5.2) des paramètres des blobs Θ sur l'ensemble des données $\mathcal{I}$ (les pixels de teinte chair). Ces pixels sont supposés indépendants bien qu'en réalité ce ne soit peut-être pas le cas.

$$L(\mathcal{I} \mid \Theta) \quad = \quad \Pr(\mathcal{I} \mid \Theta) = \prod_{i=1}^{P} \Pr(\mathbf{x}_i \mid \Theta) \tag{5.2}$$

L'équation 5.2 peut être maximisée par l'algorithme EM (Chapitre 3). Pour le modèle des blobs, je développe la maximisation analytique (Etape M) pour annuler les dérivées partielles $\frac{\partial Q(\Theta,\hat{\Theta})}{\partial \boldsymbol{\mu}_j} = 0$ et $\frac{\partial Q(\Theta,\hat{\Theta})}{\partial \boldsymbol{\sigma}_j} = 0$.

5.3.5 Ajustement des blobs

Les variables cachées

L'ajout de variables cachées doit faciliter la résolution du problème. Dans mon cas, ces variables cachées vont me permettre d'associer un pixel à un et un seul blob. Soit les paramètres cachés $\mathcal{H} = h_1 \dots h_P$, où h_i est le blob associé au pixel i. L'ensemble complet des données $\mathcal{D}$ est défini par :

$$\mathcal{D} = (\mathcal{I}, \mathcal{H}) = (\mathbf{x}_1 \dots \mathbf{x}_P, h_1 \dots h_P)$$

La fonction de vraisemblance sur $\mathcal{D}$ est :

$$\begin{aligned} L(\mathcal{D} \mid \Theta) &= \Pr(\mathcal{I}, \mathcal{H} \mid \Theta) = \prod_{i=1}^{P} \Pr(\mathbf{x}_i, h_i \mid \Theta) \\ &= \prod_{i=1}^{P} \Pr(\mathbf{x}_i \mid h_i, \Theta) \Pr(h_i \mid \Theta) \end{aligned} \tag{5.3}$$

Posons z_{ij} la variable aléatoire définie par :

$$z_{ij} = \begin{cases} 1 & : \quad h_i = j \\ 0 & : \quad h_i \neq j \end{cases}$$

Alors on peut réécrire l'équation 5.3 comme :

$$L(\mathcal{D} \mid \Theta) = \prod_{i=1}^{P} \prod_{j=1}^{B} \Pr(\mathbf{x}_i \mid h_i = j, \Theta)^{z_{ij}} \Pr(h_i = j \mid \Theta)^{z_{ij}}$$

Etape d'Estimation

L'équation 3.31 me permet d'écrire la fonction auxiliaire :

$$\begin{aligned} Q(\Theta, \hat{\Theta}) &= E_{\mathcal{H}}[log L(\Theta, \mathcal{D}) \mid \mathcal{I}, \hat{\Theta}] \\ &= \sum_{i=1}^{P} \sum_{j=1}^{B} \Pr(h_i = j \mid \mathcal{I}, \hat{\Theta}) \\ &\quad [\log \Pr(\mathbf{x}_i \mid h_i = j, \Theta) + \log \Pr(h_i = j \mid \Theta)] \end{aligned}$$

On suppose que tous les blobs sont équiprobables car il est très difficile de donner une probabilité *a priori* à un blob. D'où :

$$\Pr(h_i = j \mid \Theta) = \frac{1}{B}$$

De plus, je fais l'hypothèse que la fonction de densité de probabilité d'un pixel de teinte chair $\mathbf{x}_i$ connaissant le blob j est une gaussienne, ainsi :

$$\begin{aligned} \Pr(\mathbf{x}_i \mid h_i = j, \Theta) &= \mathcal{G}(\mathbf{x}_i, \boldsymbol{\mu}_j, \boldsymbol{\sigma}_j) \\ &= \sqrt{\frac{\mid \boldsymbol{\sigma}_j \mid}{(2\pi)^5}}\, e^{-\frac{1}{2}^T (\mathbf{x}_i - \boldsymbol{\mu}_j) \boldsymbol{\sigma}_j (\mathbf{x}_i - \boldsymbol{\mu}_j)} \end{aligned}$$

D'autre part, étant donné qu'un pixel appartient à un et un seul blob :

$$Pr(h_i = j \mid \mathcal{I}, \hat{\Theta}) \;=\; \frac{1}{B} \frac{\Pr(\mathbf{x}_i \mid h_i = j, \hat{\Theta})}{\sum_{l=1}^{B} \Pr(\mathbf{x}_i \mid h_i = l, \hat{\Theta})}$$

Finalement, Q s'écrit :

$$\begin{aligned} Q(\Theta, \hat{\Theta}) \;&=\; \sum_{i=1}^{P} \sum_{j=1}^{B} E_{ij} \, \log \mathcal{G}(\mathbf{x}_i, \boldsymbol{\mu}_j, \boldsymbol{\sigma}_j) \\ &+\; \sum_{i=1}^{P} \sum_{j=1}^{B} E_{ij} \, \log \frac{1}{B} \end{aligned}$$

avec :

$$E_{ij} = \frac{\mathcal{G}(\mathbf{x}_i, \hat{\boldsymbol{\mu}}_j, \hat{\boldsymbol{\sigma}}_j)}{\sum_{l=1}^{B} \mathcal{G}(\mathbf{x}_i, \hat{\boldsymbol{\mu}}_l, \hat{\boldsymbol{\sigma}}_l)} \tag{5.4}$$

L'étape d'estimation de l'algorithme EM s'achève sur le calcul des E_{ij} en utilisant les données $\mathcal{I}$ et les paramètres précédents $\hat{\Theta}$.

Etape de Maximisation

Je recherche les paramètres Θ de sorte à maximiser la fonction auxiliaire. Pour cela, je calcule les paramètres $\boldsymbol{\mu}_j$ et $\boldsymbol{\sigma}_j$ tels que $\frac{\partial Q(\Theta, \hat{\Theta})}{\partial \boldsymbol{\mu}_j} = 0$ et $\frac{\partial Q(\Theta, \hat{\Theta})}{\partial \boldsymbol{\sigma}_j} = 0$. Les dérivées des paramètres d'une gaussienne ont déjà été développées pour les modèles de Markov cachés (Annexe C).

Ainsi, la dérivation de $Q(\Theta, \hat{\Theta})$ par rapport à $\boldsymbol{\mu}_j$ donne une somme des pixels pondérés par l'espérance E_{ij} :

$$\frac{\partial Q(\Theta, \hat{\Theta})}{\partial \boldsymbol{\mu}_j} = 0 \;\Leftrightarrow\; \boldsymbol{\mu}_j = \frac{\sum_{i=1}^{P} E_{ij} \, \mathbf{x}_i}{\sum_{i=1}^{P} E_{ij}} \tag{5.5}$$

et la dérivation de $Q(\Theta, \hat{\Theta})$ par rapport à $\boldsymbol{\sigma}_j$ donne l'inverse de la matrice de corrélation des pixels $\mathbf{x}_i$ pondérée par les espérances E_{ij} :

$$\begin{aligned} &\frac{\partial Q(\Theta, \hat{\Theta})}{\partial \boldsymbol{\sigma}_j} = 0 \\ \Leftrightarrow\quad &\boldsymbol{\sigma}_j = \left(\frac{\sum_{i=1}^{P} E_{ij} \, (\mathbf{x}_i - \boldsymbol{\mu}_j)^T (\mathbf{x}_i - \boldsymbol{\mu}_j)}{\sum_{i=1}^{P} E_{ij}} \right)^{-1} \end{aligned} \tag{5.6}$$

L'algorithme EM appliqué aux blobs devient à présent :

- Pour k = 1 ... K, avec K étant le nombre d'itérations EM
  - Calculer les E_{ij} d'après l'équation 5.4,
  - Calculer les $\{\boldsymbol{\mu}_j\}$ d'après l'équation 5.5,
  - Calculer les $\{\boldsymbol{\sigma}_j\}$ d'après l'équation 5.6.

5.3.6 Résultats

Les blobs obtenus (Figure 5.18 et 5.19) sont plus robustes que les zones de pixels de teinte chair connexes. En effet, on constate que les blobs sont géométriquement plus stables que les zones.

Figure 5.18: Exemple de blobs du visage et de la main sur fond uniforme.

Figure 5.19: Exemple de blobs du visage et de la main sur fond complexe.

Par exemple, dans une séquence ne contenant que le visage, la variation de la surface d'un blob-visage est moindre que celle d'une zone-visage (Figure 5.20).

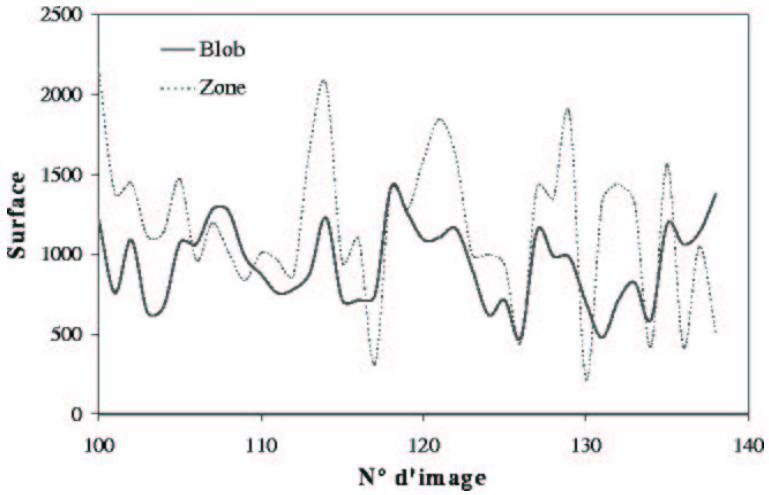


Figure 5.20: Variation de la surface d'une zone et d'un blob sur une séquence visage seul.

Une fois les blobs obtenus, je cherche à reconnaître une main aux coordonnées données par le blob. Le suivi des blobs dans une séquence d'images est réalisé par un appariement. Les blobs de deux images successives sont appariés s'ils sont proches, s'ils ont la même forme (surface identique) ou s'ils ont le même vecteur directeur. Ainsi, la reconnaissance d'un geste de la main devient possible.

5.4 Reconnaissance statique

Cette section présente une approche générative de réseau de neurones artificiels pour la reconnaissance statique de posture de la main dans les images.

5.4.1 Reconnaissance des postures de la main

Les réseaux de neurones (Chapitre 3), comme les modèles discriminants [Murakami et Taguchi, 1991] ou les modèles à base de cartes topologiques de Kohonen [Boehm et al., 1994], ont été précédemment appliqués à la reconnaissance des postures de la main. Dans cette thèse, nous proposons d'utiliser un modèle de réseau de neurones déjà appliqué à la détection de visage : le modèle génératif contraint (CGM) [Feraud, 1997] (Figure 5.21). Le modèle CGM a déjà obtenu de très bons résultats en détection de visage.

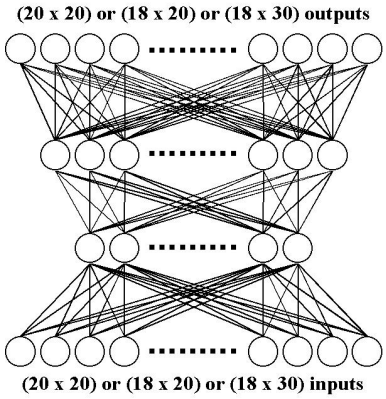


Figure 5.21: Le modèle génératif contraint.

Le but de l'apprentissage génératif contraint est de se rapprocher le plus possible de la distribution de probabilité de l'ensemble des mains $\mathcal{M}$ en utilisant un réseau de neurones non-linéaire à compression et un ensemble de contre-exemples $\bar{\mathcal{M}}$.

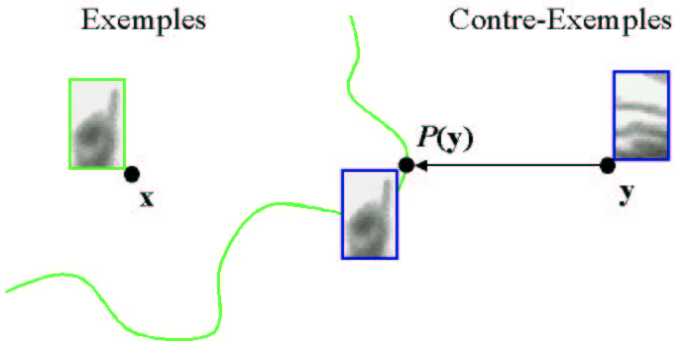


Figure 5.22: Projection d'un contre-exemple sur l'ensemble des exemples.

Le CGM possède autant d'entrées que de sorties. On présente une image en entrée et on reconstruit cette image en sortie. D'un côté, chaque exemple de main $\mathbf{x} \in \mathcal{M}$ est reconstruit à l'identique. De l'autre côté, chaque contre-exemple de main $\mathbf{y} \in \bar{\mathcal{M}}$ est contraint à être reconstruit comme $\mathcal{P}(y)$. $\mathcal{P}(y)$ est la projection de y sur l'ensemble des exemples de main $\mathcal{M}$. Cette projection est calculée en effectuant la moyenne des exemples appartenant au voisinage de l'exemple de main qui est le plus proche du contre-exemple (Figure 5.22).

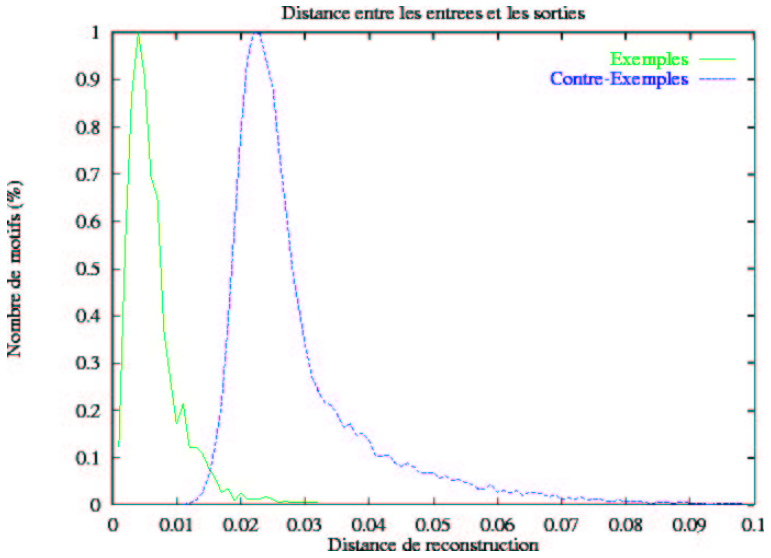


Figure 5.23: Distribution de la distance entre l'entrée et la sortie.

Après l'apprentissage, la distance euclidienne entre les entrées et les sorties du CGM est calculée. Cette distance (Figure 5.23) est plus faible pour les exemples de main que pour les contre-exemples de main. Ainsi, en utilisant cette distance, il est possible de séparer les mains des contre-exemples, grâce à un seuil. Ce seuil est estimé sur l'histogramme de la distance entrée-sortie des exemples et des contre-exemples d'un ensemble de validation.

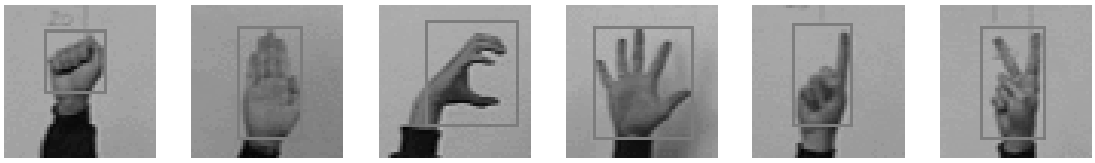


Figure 5.24: Exemples d'images de postures de la main extraites de la base HandTNT-B.

Mon objectif étant de reconnaître des postures de la main utilisées dans les gestes de commande, j'ai sélectionné un petit ensemble de catégories de postures de la main: A, B, C, Compte5, Pointe et V. Le choix de ce vocabulaire de postures est inspiré de la langue des signes. De plus, ce sont les postures de la main parmi les plus simples à exécuter.

Une base de données de plusieurs milliers d'images de postures de la main sur fond uniforme et sur fond complexe a été construite. Ces images proviennent d'un corpus vidéo et d'images fixes. Le corpus vidéo a été construit en filmant une dizaine de sujets (hommes et femmes) exécutant différents gestes de la main avec des fonds uniformes et complexes. Les images fixes proviennent d'Internet, elles représentent, notamment, des sportifs (basket ball, volley ball) dont on voit la main. Puis, les images sont rehaussées pour augmenter les contrastes, puis elles sont lissées pour réduire le bruit.

On apprend un CGM pour chaque posture. Les tailles des fenêtres pour chaque posture (correspondant au nombre d'entrée de chaque CGM) sont : 20x20 pixels pour A, 18x20 pixels pour C et Compte5, et 18x30 pixels pour B, Pointe et V (Figure 5.24). L'architecture des réseaux CGM (Table 5.1) a été choisie expérimentalement sur l'ensemble des exemples et sur un premier ensemble de contre-exemples. Pour chaque posture, l'architecture retenue est celle qui reconstruit le mieux en validation les exemples et les contre-exemples. Puis, une fois l'architecture choisie, des contre-exemples spécifiques à chaque modèle ont été récoltés. La plus grande partie de la base de données a été utilisée pour l'apprentissage (80 %, base HandTNT-A) et le reste pour le test (20 %, base HandTNT-B).

Tableau 5.1: Taille des réseaux de neurones CGM utilisés.

Réseau CGM	# Neurones d'entrée	# Neurones cachés (couche 1)	# Neurones cachés (couche 2)	# Neurones de sortie
A	400	50	90	400
C, Compte5	360	40	70	360
B, Pointe, V	540	60	90	540

5.4.2 Résultats sur notre base d'images de test.

Dans les images de test de la base HandTNT-B, les postures de la main possèdent des tailles variables. Afin de trouver le cadre englobant une posture de main, les images de cette base sont testées à différentes positions et à différentes échelles par un CGM spécifique à chaque posture. A l'issue de cette phase, on obtient des images contenant des cadres de différentes tailles. Si un cadre contient une posture correspondant au CGM appliqué alors on considère qu'il s'agit là d'une détection. Si le cadre contient une posture ne correspondant pas au CGM appliqué ou si le cadre ne contient pas de posture alors on considère qu'il s'agit là d'une fausse alarme.

Calculer le taux de fausses alarmes est indispensable pour évaluer la performance du système. Ce taux doit être faible au regard du nombre de tests dans une image (10^6 tests en moyenne par image). En l'état actuel, le taux de fausses alarmes sur 200 images variées ne contenant pas de main est d'environ 1 fausse alarme pour 27 800 tests. Dans la suite de ce chapitre, nous préciserons les intervalles de confiance associés aux résultats. On suppose que le résultat R est une variable aléatoire qui suit une loi normale. Ainsi, étant donné α un réel ($0 \leq \alpha \leq 1$), la table de la loi normale centrée réduite permet de déterminer le réel t_α tel que $Pr[|R| < t_\alpha] = 1 - \alpha$.

L'intervalle de confiance d'une réalisation X de R est alors donné par $\pm\sqrt{\frac{X(1-X)}{n}}t_\alpha$ en fonction du seuil α et de la taille de l'échantillon n.

Tableau 5.2: Résultats obtenus sur la base de test HandTNT-B sur fonds uniformes et sur fonds complexes donnés avec un intervalle de confiance à 95% ($\alpha = 0,05$ et $t_\alpha = 1,96$)

Fonds uniformes			
Postures	Nombre d'images	Taux de détection moyen	Taux de fausse alarme moyen
A,B,C et V	241	$94\% \pm 3\%$	1/11 100
de A à V	382	$93,5\% \pm 2,5\%$	1/12 000
Fonds complexes			
Postures	Nombre d'images	Taux de détection moyen	Taux de fausse alarme moyen
A,B,C etV	165	$75\% \pm 6,6\%$	1/18 900
de A à V	277	$76\% \pm 5\%$	1/14 000

Même si les images avec fond complexe sont plus difficiles, le CGM obtient un bon taux de reconnaissance et un faible taux de fausses alarmes (Table 5.2). Néanmoins, on remarque que le taux de fausse alarme est plus élevé pour les fonds uniformes que pour les fonds complexes, ce qui semblerait venir du fait que le nombre d'images uniformes est plus important que le nombre d'images complexes. Ainsi, il y a plus de tests qui sont réalisés dans le premier cas, ce qui augmente les chances de se tromper. Je n'ai pas jugé nécessaire de déterminer les matrices de confusion. En effet, les postures n'ont pas toutes la même taille de fenêtre d'entrée. Il devient ainsi difficile d'estimer les détections et les fausses alarmes sur des images réelles. De plus, les détecteurs de postures n'ont pas été entraînés pour s'exclure. A l'heure actuelle, il n'y a pas de travaux similaires dont on peut comparer les résultats. Néanmoins, une base d'images existe et peut être utilisée comme benchmark.

5.4.3 Résultats sur une base de Benchmark

Le CGM a aussi été testé sur la base d'images de Jochen Triesch [Triesch et Malsburg, 1996], téléchargée en 1998 sur internet depuis l'archive PEIPA[3].

Figure 5.25: Exemples d'images de postures extraites de la base de Jochen Triesch.

[3]PEIPA: the Pilot European Image Processing Archive, http://peipa.essex.ac.uk

Tableau 5.3: Résultats obtenus sur la base de Jochen Triesch sur fonds uniformes et fonds complexes donnés avec un intervalle de confiance de 95%.

Fonds uniformes			
Postures	Nombre d'images	Taux de détection moyen	Taux de fausse alarme moyen
A,B,C,V	191	$94\% \pm 3,4\%$	1/25 600
Fonds complexes			
Postures	Nombre d'images	Taux de détection moyen	Taux de fausse alarme moyen
A,B,C,V	96	$85\% \pm 7,2\%$	1/15 900

Cette base contient des images en niveau de gris de 128x128 pixels de 10 signes de la main effectués par 24 personnes sur des fonds uniformément éclairés, sur des fonds uniformément sombres et sur des fonds complexes. J'ai seulement testé les postures A, B, C et V (Figure 5.25) car les postures Compte5 et Pointe n'existent pas dans la base de Jochen. Le CGM appliqué sur ces images donne des résultats satisfaisants (Table 5.3).

Néanmoins, je pense que ces résultats peuvent être encore améliorés en ajoutant plus d'exemples et surtout de contre-exemples pour réduire le taux de fausses alarmes.

J'ai intégré la détection des postures de la main dans un nouveau système basé sur LISTEN. Ce système, appelé GHOST et détaillé à la fin de ce chapitre, fonctionne en temps réel à 12Hz. Cependant, lorsqu'une posture de la main est présentée, il n'y a aucune stratégie pour choisir le CGM à appliquer. Actuellement, tous les CGM sont utilisés et celui avec la plus forte réponse permet d'identifier la posture. Une prochaine perspective serait de trouver une stratégie pour choisir le CGM à utiliser.

5.5 Détection des transitions

5.5.1 Détection des changements de postures

Dans cette section, je me concentre sur les gestes dynamiques locaux que je considère comme une séquence de postures de la main. Partant du principe que le réseau de neurones CGM détecte et identifie les différentes postures, je cherche à présent une technique pour discriminer les changements de postures. Je propose d'utiliser une technique de décomposition en moments géométriques et plus particulièrement celle des moments de Zernike. Cette étude sur les moments de Zernike a fait l'objet d'un stage d'ingénieur dont j'ai assuré l'encadrement.

Le principe de la discrimination est le suivant. A partir de la séquence vidéo, on extrait les images successives et on calcule les moments de Zernike correspondant à l'image précédente de la main et à l'image courante de la main. On détermine

l'erreur entre les moments de ces deux images et suivant un seuil fixé au préalable, on détermine s'il y a eu un changement de posture. On suppose que l'on sait qu'il y a présence d'une main dans l'image.

Les fonctions mathématiques étudiées portent le nom du physicien qui en est à l'origine : **Fritz Zernike**, Prix Nobel de physique (1953). Il est à l'origine de la technique du contraste de phase [Zernike, 1934] qui consiste à mettre en phase lumière réfléchie et transmise pour observer les corps d'une extrême transparence. Les propriétés d'invariance de formes de ces fonctions sont aujourd'hui utilisées pour analyser et quantifier les aberrations géométriques des instruments d'optique (astigmatisme, focus, coma, trefoil) mais aussi en traitement d'image. Le choix des moments de Zernike par rapport aux autres moments (Hu, Fourier-Mellin), ainsi que la description des propriétés générales des moments sont présentés en annexe F.

5.5.2 Description des enchaînements de postures

Un vocabulaire de gestes de la main formé d'enchaînement de postures [Quek, 1994] (Figure 5.26) a déjà été proposé. Nous nous inspirons de certains d'entre eux (Stop, Rotate) pour établir notre propre vocabulaire. Notre vocabulaire comporte six gestes de la main.

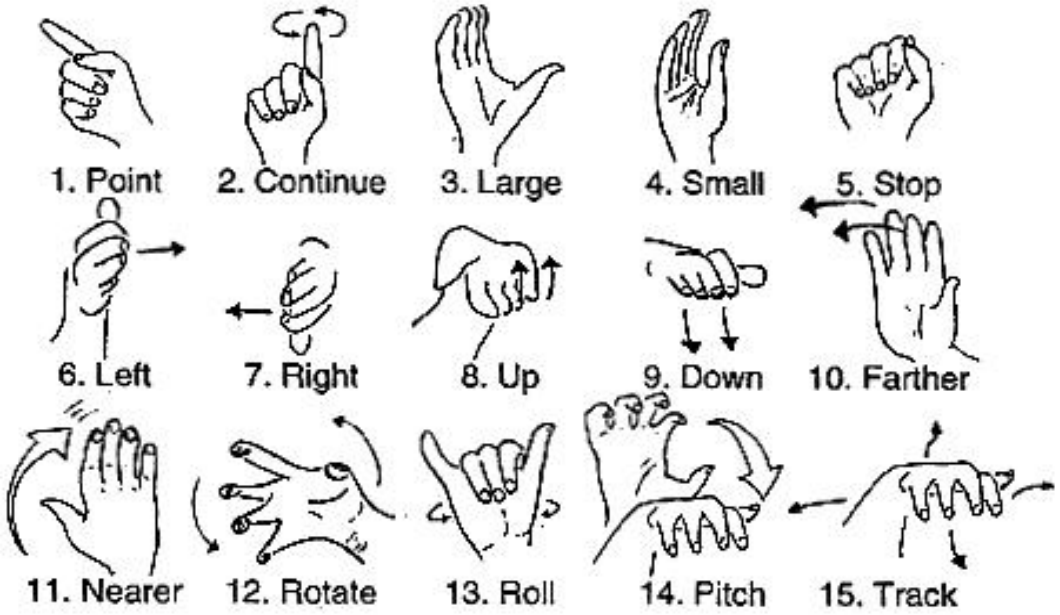


Figure 5.26: Vocabulaire de gestes de la main (Extrait de [Quek, 1994]).

Ces gestes sont composés par l'enchaînement de quatre des six postures de la main décrites précédemment (A, B, Pointe et Compte5). Les deux autres postures (C et V) n'ont pas été choisies car elles n'avaient pas de significations évidentes pour être utilisées comme gestes de commande. De plus, il s'est avéré que la posture C est la plus difficile à réaliser.

Le tableau suivant indique pour tous les enchaînements de postures le nombre de séquences disponibles. Ces séquences contiennent en moyenne une centaine d'images couleurs d'environ 60x60 pixels. Pour faciliter le traitement, les séquences Stop, Grasp et OK ont été regroupées. On obtient l'enchaînement suivant : A-C5-A-B.

Tableau 5.4: Les enchaînements de postures utilisés.

Geste	Postures enchaînées	Nombre de séquences	Annexe
Clic	A-P-A	15	G.1
Stop	A-C5	15	G.2
Grasp	C5-A	15	G.2
OK	A-B	15	G.2
No	P + *mvt oscillatoire*	14	G.3
Rotate	C5 + *rotation*	13	G.4

Dans le but d'améliorer la précision du traitement, on propose d'effectuer un pré-traitement localisant grossièrement la main et réduisant le champ d'étude dans l'image. Cela permet comme sur les bases d'images de se focaliser sur l'objet à décomposer. Pour cela, on utilise un traitement déjà utilisé dans l'application LISTEN : la détection de la teinte chair. A partir du masque de teinte chair, on calcule les caractéristiques géométriques relatives aux moments de Zernike, c'est-à-dire le centroïde et le rayon d'étendue. On obtient donc un cercle centré sur la main (Figure 5.27).

Figure 5.27: Réduction de la zone à partir du masque binaire de teinte chair, du centroïde et du rayon d'étendue.

5.5.3 Discrimination

On peut maintenant calculer les moments sur la zone concernée. La partie en dehors du cercle unité n'est donc plus utilisée, on se focalise uniquement sur l'objet. Pour détecter le changement de posture, on calcule l'erreur sur la valeur du module des moments entre deux images successives. On prend le module des moments car il est invariant en rotation (voir Annexe F.10) et préserve l'information de forme.

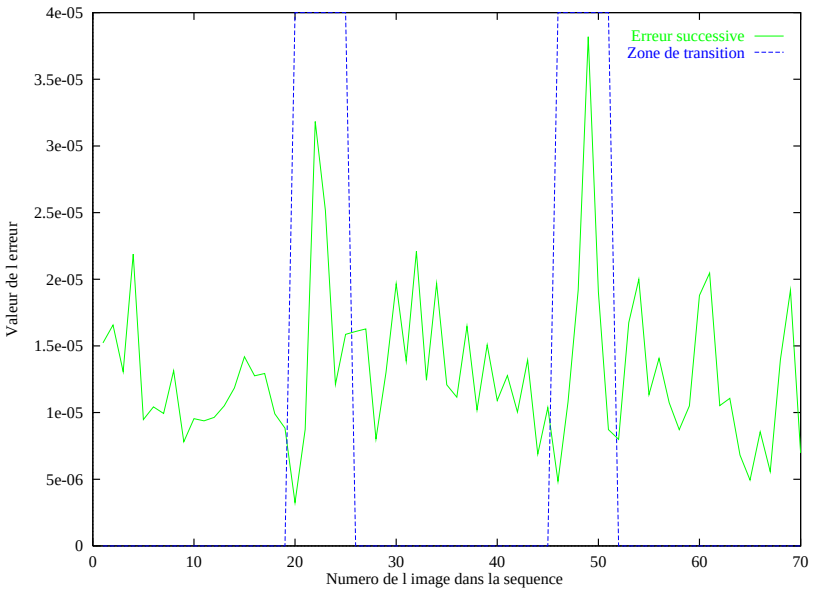


Figure 5.28: L'erreur successive sur la valeur des moments pour une séquence Clic met en évidence par des pics d'erreurs les zones de transitions.

Ainsi, on voit apparaître des pics correspondant aux zones de transitions (Figure 5.28). Pour visualiser correctement cette erreur, on a indexé les différentes images correspondant aux zones de transition. L'erreur successive semble indiquer fidèlement les changements. Lors de la transition, il y a de fortes fluctuations des moments jusqu'à la posture suivante où l'erreur se stabilise. La difficulté est de déterminer un seuil de détection qui fonctionne avec toutes les séquences de la base.

On utilise une fenêtre glissante qui englobe plusieurs valeurs précédentes de l'erreur et on calcule une moyenne locale sur cette fenêtre. En appliquant un coefficient multiplicatif à cette moyenne locale, on est en mesure d'obtenir un seuil correct. On dispose donc de deux paramètres de réglage de ce seuil : la taille de la fenêtre glissante T_{FG} et le coefficient appliqué à la moyenne α_M. Ces deux paramètres ont été obtenus de manière expérimentale sur la totalité de nos séquences. Ces expériences constituent une approche biaisée, car tout l'ensemble de données est utilisé au lieu de le faire sur un ensemble de validation, mais le peu de séquences à notre disposition nous l'a imposé. Néanmoins, l'approche générale reste valide.

5.5.4 Résultat de détection sur les bases de séquences

Tout d'abord, il faut préciser la méthode employée pour calculer les performances en détection. Les séquences ont été divisées en deux groupes, celles qui présentent des changements (Clic, Stop-Grasp-Ok) et celles qui n'en présentent pas (No, Rotate). Puis, on détermine le taux de détection TD et le taux de fausses alarmes TFA.

$$\text{TD} = \frac{\#\text{DC}}{\#\text{C}} \qquad (5.7) \qquad\qquad \text{TFA} = \frac{\#\text{FDC}}{\#\text{TI}} \qquad (5.8)$$

TD est le taux de détection, #DC est le nombre de détection d'un changement estimé, #C est le nombre de changements effectifs, #FDC est le nombre de fausses détections d'un changement estimé, TFA est le taux de fausses alarmes et #TI est nombre total d'images de toutes les séquences.

Tableau 5.5: Résultats obtenus pour les séquences Clic et Stop-Grasp-OK donnés avec un intervalle de confiance à 95%.

TD moyen	TFA moyen
$93\% \pm 5,6\%$ (71 / 75)	$2,2\% \pm 0,5\%$ (73/3317)

Les taux de fausses alarmes des séquences No et Rotate (Table 5.6) ont été séparés, car il s'est avéré que les différentes séquences Rotate présentent une rotation n'étant pas dans le plan de la caméra (voir Annexe G - Séquence G.4 - Image 9) et sont interprétées comme un changement de posture.

Tableau 5.6: Résultats obtenus pour les séquences No et Rotate donnés avec un intervalle de confiance à 95%.

Séquences	TFA
No	$0,6\% \pm 1,5\%$(6 / 996)
Rotate	$2,2\% \pm 0,7\%$ (43 / 1903)

En combinant la discrimination de postures par les moments de Zernike et la reconnaissance des postures par réseau de neurones, il semble possible de mettre en place un système capable d'identifier des gestes simples assimilés à des séquences de postures de la main. Cependant, il semble pour l'heure exclu d'obtenir un fonctionnement en temps réel, le calcul des moments, même à l'ordre 20, reste coûteux en temps de calcul, même s'il est moindre que celui des réseaux de neurones. De plus, la reconnaissance des gestes incluant une rotation (No, Rotate) nécessite de posséder une information d'angle. Cette information peut être extraite à partir de la valeur des moments en utilisant la relation F.10.

5.6 Reconnaissance dynamique

La reconnaissance des gestes dynamiques est un problème difficile. Le problème est, bien évidemment, de reconnaître la trajectoire gestuelle dans l'espace des paramètres, mais il est aussi de pouvoir détecter la fin d'un geste. Si l'on considère le mouvement de la main comme un signal, alors les gestes que l'on cherche à reconnaître peuvent se trouver dans ce signal. Ainsi, comment reconnaître un échantillon de ce signal comme un geste lorsque l'on ignore le début, la fin et la durée de cet échantillon ?

Un grand nombre de méthodes a été proposé pour la reconnaissance des gestes dynamiques : des modèles récurrents de réseaux de neurones [Murakami et Taguchi, 1991], des modèles de Markov cachés [Starner et Pentland, 1995] ou des décompositions des gestes en espaces propres [Watanabe et Yachida, 1998]. D'un côté, les réseaux de neurones obtiennent de bonnes performances en classification, mais ne sont pas adaptés naturellement à des observations de longueurs variables. Même si des modèles de réseaux de neurones récurrents existent, et sont en théorie capables de traiter les séquences de données, ils ne sont pas naturellement adaptés à ce genre de problèmes. D'un autre côté, les HMM permettent de calculer plus fidèlement la probabilité que les observations aient été générées par le modèle, mais ils n'ont pas des propriétés de discrimination efficaces.

Dans cette partie, nous étudierons les HMM, les MLP, les IOHMM et les GIOHMM appliqués à la reconnaissance de trajectoires gestuelles.

5.6.1 Les trajectoires gestuelles

Les trajectoires sont des séquences de points $[x_t, y_t]$ (coordonnées de la main dans une image à l'instant t) segmentées manuellement (le début et la fin sont indexés), puis extraites de notre corpus vidéo par un suivi automatique des blobs (cf la première partie de ce chapitre sur la segmentation). Le corpus vidéo a été construit à l'aide de plusieurs personnes exécutant plusieurs fois les différentes classes de gestes.

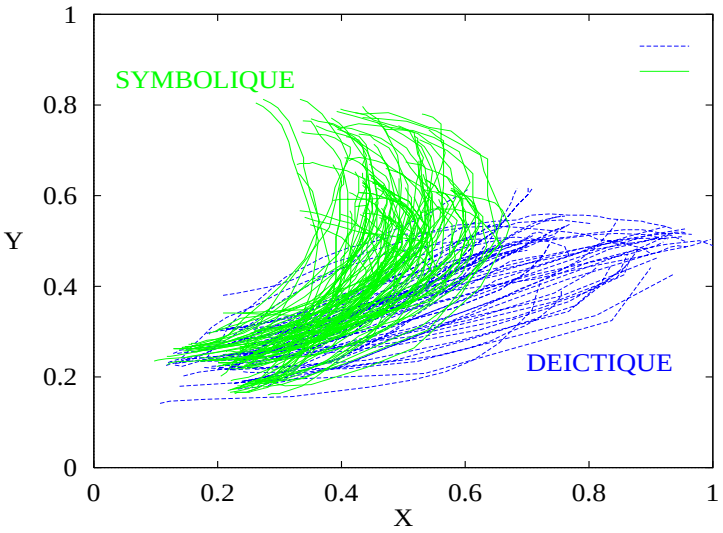


Figure 5.29: Trajectoires des gestes déictiques et symboliques dans l'espace corporel centré sur le visage.

Nous disposons de deux trajectoires de gestes élémentaires : les gestes déictiques et les gestes symboliques (Figure 5.29). Les gestes déictiques sont des mouvements de pointage de la main vers la droite ou la gauche. Les gestes symboliques correspondent à des gestes de commandes qui amènent la main à la hauteur des épaules.

Tableau 5.7: Description de la base de gestes.

Gestes déictiques

	P	T_{min}	T_{moyen}	T_{max}
Ensemble d'apprentissage	152	5	13	29
Ensemble de validation	76	5	14	29
Ensemble de test	67	5	15	28

Gestes symboliques

	P	T_{min}	T_{moyen}	T_{max}
Ensemble d'apprentissage	196	5	17	36
Ensemble de validation	98	5	17	36
Ensemble de test	99	8	18	37

Gestes de rétractation déictiques

	P	T_{min}	T_{moyen}	T_{max}
Ensemble d'apprentissage	140	5	14	27
Ensemble de validation	60	5	13	27
Ensemble de test	49	5	14	25

Gestes de rétractation symboliques

	P	T_{min}	T_{moyen}	T_{max}
Ensemble d'apprentissage	96	7	16	27
Ensemble de validation	62	8	17	28
Ensemble de test	48	9	18	33

De plus, ces deux gestes s'achèvent par une phase de rétractation qui ramène la main dans sa position initiale. Ces deux phases de rétractation distinctes sont également décrites par des trajectoires. On obtient finalement quatre gestes distincts : les gestes déictiques, les gestes symboliques, les gestes de rétractation déictiques et les gestes de rétractation symboliques. Toutes les trajectoires sont normalisées dans un espace corporel dont la définition se trouve à la fin de ce chapitre.

La base de données de trajectoires ainsi obtenue est présentée (Table 5.7). L'ensemble des trajectoires d'une catégorie de geste est divisé en trois sous-ensemble : l'ensemble d'apprentissage est utilisé pour entraîner un modèle statistique, l'ensemble de validation est utilisé pour contrôler l'apprentissage et l'ensemble de test est utilisé pour évaluer les performances du modèle en généralisation. La table 5.7 présente, pour chaque geste et chaque ensemble, le nombre total de trajectoires P, ainsi que le nombre minimum T_{min}, moyen T_{moyen} et maximum T_{max} de points.

5.6.2 Résultats comparatifs de classification

Nous comparons les performances en classification de deux gestes (déictique et symbolique) pour les MLP (cf. chapitre 3), les HMM (cf. chapitre 3) et les IOHMM (cf. chapitre 4). Tous les résultats présentés sont obtenus sur les ensembles de test et sont donnés avec un intervalle de confiance à 95%.

Classification par un MLP

Tout d'abord, nous comparons les IOHMM à une autre méthode basée sur un réseau de neurones multicouches (MLP) avec une fenêtre d'entrée fixe. En effet, comme les trajectoires gestuelles sont de longueurs variables, elles sont interpolées avant leurs présentations au MLP. J'ai choisi d'interpoler toutes les séquences au nombre moyen d'observations $T_{moyen} = 16$. Dans cette expérience, il s'agit simplement de montrer la facilité avec laquelle un MLP statique peut résoudre le problème. C'est pourquoi, on ne cherche pas une architecture optimale au sens d'un ensemble de validation. Par conséquent, la taille du vecteur d'entrée est $m = 32$ pour le MLP basé sur des trajectoires gestuelles interpolées. Le MLP ne possède qu'un seul neurone de sortie s. Ainsi, j'ai choisi d'apprendre $s = 1$ comme sortie désirée pour les gestes déictiques et $s = 0$ comme sortie désirée pour les gestes symboliques.

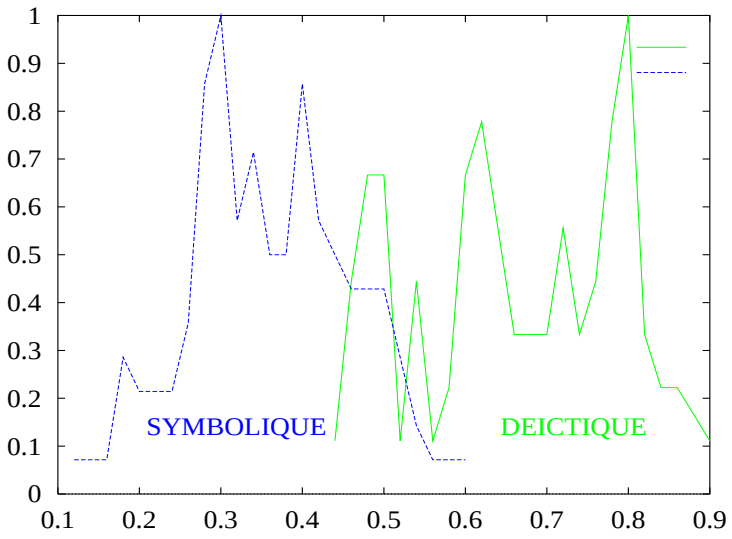


Figure 5.30: Histogramme de la sortie du MLP pour les gestes de l'ensemble de test.

La figure 5.30 représente la distribution de la sortie du MLP pour tous les gestes déictiques et tous les gestes symboliques de l'ensemble de test. A l'issue de l'apprentissage, le MLP réalise donc la séparation des gestes déictiques et des gestes symboliques. Ainsi, il est possible de choisir une valeur de seuil (aux alentours de 0.5) et d'estimer le taux de classification (Table 5.8). Le MLP sépare les gestes très efficacement. En effet, le MLP prend toute la trajectoire comme vecteur d'entrée, ce qui impose d'en connaître le début et la fin.

En réalité, on ne connaît pas le début d'un geste et encore moins la fin. Il est possible de détecter les phases de début et de fin en analysant l'accélération ou l'arrêt d'un mouvement. Cependant, le problème n'est toujours pas résolu si le mouvement contient plusieurs gestes et que la main ne s'est pas arrêtée. Il est également possible d'utiliser une fenêtre glissante que l'on complète avec le dernier événement. Néanmoins, cette méthode requiert un a priori sur la durée des trajectoires, or les gestes peuvent être très variables.

Classification par un HMM

On associe un HMM à une classe de séquences d'observations. Un HMM utilise l'observation courante (un point) et calcule la probabilité que la séquence observée au cours du temps ait été générée par le modèle. Cette probabilité, appelée vraisemblance, est utilisée pour choisir le modèle le mieux adapté.

J'ai construit, dans le cas discret et dans le cas continu, deux HMM avec une topologie gauche-droite : un pour les gestes déictiques et un pour les gestes symboliques. Dans le cas discret, chaque axe de l'espace des observations ($\mathbb{R}^2$) est divisé en d intervalles pour obtenir un "codebook" de dimension $D = d^2$. En pratique, j'ai testé différentes valeurs du nombre d'états $N = 3, 5, 7, 9, 10, 20$, du nombre d'intervalles $d = 4, 6, 10$ et du nombre de mélanges de gaussiennes $M = 1, 2, 3$. J'observe également que les HMM discrets obtiennent toujours, même à nombre de paramètres comparables, de meilleurs taux de classification que les HMM continus. Aussi, seuls les résultats sur les HMM discrets sont présentés (Table 5.8). Le taux de classification est déterminé sur l'ensemble de test en choisissant le modèle dont la probabilité d'observation de la séquence est la plus forte (Maximum A Posteriori).

Classification par un IOHMM

Les trajectoires sont des séquences de points $[x_t, y_t]$ (coordonnées de la main dans une image à l'instant t) et nous travaillons sur deux classes de gestes. Par conséquent, la dimension d'entrée est $m = 2$ et la dimension de sortie est $r = 1$. J'ai choisi d'apprendre $y_1 = 1$ comme sortie pour les gestes déictiques et $y_1 = 0$ comme sortie pour les gestes symboliques. Le IOHMM est composé de trois états (un état initial, un état final et un état intermédiaire) et possède une topologie gauche-droite comme pour les HMM. Dans ce IOHMM, les réseaux d'états et les réseaux de sorties sont des MLP dotés de 2 entrées et respectivement de 6 (cf. le nombre d'états) et 1 (une valeur haute et une valeur basse) sorties. La classe de la séquence d'entrée à chaque instant t est donnée par la sortie globale $\boldsymbol{\eta}_t^*$ du IOHMM (Equation **??**).

La figure 5.31 présente, pour toutes les séquences de test des deux classes d'apprentissage, la moyenne et la variance de la sortie globale. On constate que le IOHMM peut séparer un geste déictique d'un geste symbolique à la fin de la séquence. Ainsi, en utilisant un seuil sur la sortie globale, il est possible de mesurer le taux de reconnaissance (Table 5.8).

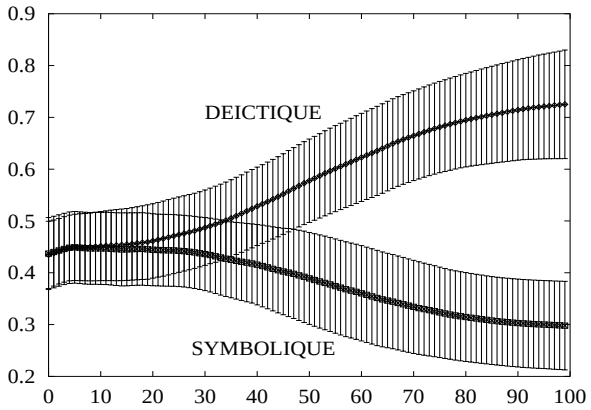


Figure 5.31: Distribution de la sortie globale ($\boldsymbol{\eta}_t^*$) du IOHMM en fonction de la durée t de la séquence.

Résultats comparatifs

Les taux de classification sur l'ensemble de test pour les MLP, les HMM discrets et les IOHMM sont présentés (Table 5.8). On constate, tout d'abord, que le MLP obtient de très bon résultats en classification. Puis, on remarque que le taux de reconnaissance des gestes symboliques par le HMM est faible. Ce mauvais résultat provient du fait de l'utilisation du maximum a posteriori et du fait que les HMM sont entraînés séparément.

Tableau 5.8: Taux de classification pour les HMM discrets, les IOHMM entre **90**% et **100**% de la séquence, et le MLP utilisant les trajectoires interpolées.

	Déictique	Symbolique
MLP	$98,2\% \pm 3,18\%$	$98,9\% \pm 2,05\%$
HMM discret	$96,8\% \pm 4,2\%$	$65,6\% \pm 9,35\%$
IOHMM (90%-100%)	$97,6\% \pm 3,66\%$	$98,9\% \pm 2,05\%$

Le IOHMM discrimine un geste déictique d'un geste symbolique en utilisant l'observation courante après que 60% de la séquence ait été présentée. Il obtient son meilleur taux de reconnaissance entre 90% et 100% de la séquence. Dans ce cas, le IOHMM donne des résultats équivalents aux MLP et meilleurs que les HMM. Le IOHMM est plus avantageux que le MLP utilisé. L'entrée est le vecteur d'observation courant $[x_t, y_t]$, et il n'est pas nécessaire de connaître, comme pour le MLP, la fin de la séquence, et ainsi de fixer une fenêtre temporelle a priori. Toutefois, les gestes non-appris (par exemple les gestes de rétractation déictiques et symboliques) ne peuvent pas être correctement classifiés ni par la sortie globale du IOHMM ni par la sortie du MLP basé sur les trajectoires interpolées.

5.6.3 Résultats de classification et de rejet

Nous venons de mettre en évidence les propriétés des MLP, des HMM et des IOHMM pour la reconnaissance de trajectoires gestuelles. A présent, nous présentons des résultats concernant le GIOHMM. Les trajectoires sont toujours des séquences de points $[x_t, y_t]$. De plus, nous voulons apprendre les quatre gestes. Par conséquent, la dimension d'entrée est $m = 2$ et la dimension de sortie est $r = 4$. D'une part, nous présentons les résultats de classification des trajectoires apprises par un GIOHMM. D'autre part, nous montrons les résultats de rejet de trajectoires non-apprises par le même GIOHMM.

Classification par un GIOHMM

Le GIOHMM possède quatre sorties et chacune de ces sorties est associée à une classe. De plus, si on observe le sens des trajectoires, on constate qu'elles ne commencent pas toutes au même endroit et qu'elles ne s'achèvent pas toutes au même endroit.

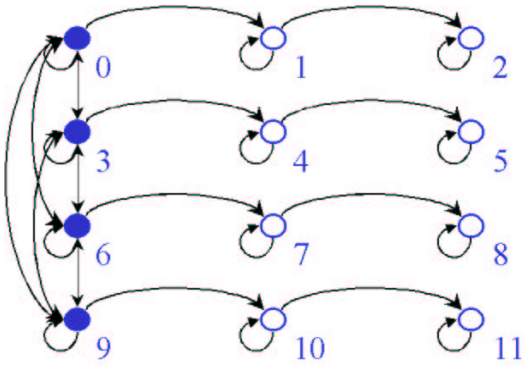


Figure 5.32: Topologie du GIOHMM utilisé pour reconnaître les gestes.

La topologie choisie pour le GIOHMM est inspirée de celle utilisée pour les HMM. Elle est formée par quatre chaînes d'états de topologie gauche-droite (Figure 5.32). Ainsi, le GIOHMM possède au total douze états : quatre états initiaux, quatre états finaux et quatre états intermédiaires. Tous les états initiaux sont reliés entre eux et tous les états initiaux sont reliés aux états intermédiaires (ni un état initial, ni un état final). Les probabilités de transitions, estimées par l'algorithme d'apprentissage (cf. Chapitre 4), dépendent des entrées (les points des trajectoires) et sont représentées par des tables de probabilités ($W = 400$ car chaque dimension est discrétisée à 20). Cependant, les réseaux de sortie sont des MLP (2 entrées, 9 neurones cachés et 4 sorties).

A chaque instant t d'une trajectoire, chaque état du modèle à une probabilité. Il est possible de représenter cette probabilité (Figures 5.33 et 5.34) pour chaque état (en ordonnée) et à chaque instant t (en abscisse). On remarque que les deux séquences commencent à l'instant ($t = 1$) dans un état initial distinct, et s'achèvent ($t = T$) dans un état final différent.

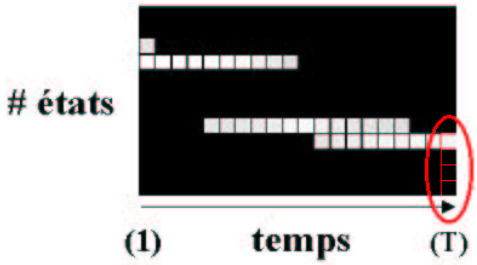


Figure 5.33: Probabilité des états associés à un geste déictique.

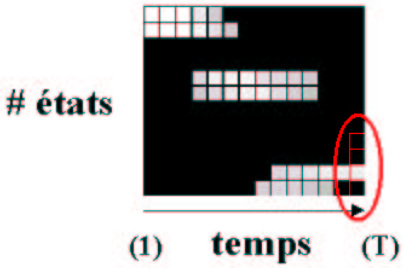


Figure 5.34: Probabilité des états associés à un geste de rétractation déictique.

Les résultats de classification des quatre gestes par le GIOHMM sont présentés (Table 5.9). Ils montrent que la classification de plusieurs trajectoires différentes est efficace et que les confusions sont faibles.

Tableau 5.9: Taux de classification pour le GIOHMM entre **90**% et **100**% de la séquence.

	Déictique	Symbolique	Rétractation Déictique	Rétractation Symbolique
Déictique	100%	0%	0%	0%
Symbolique	16%	$84\% \pm 7,22\%$	0%	0%
Rétractation Déictique	2%	0%	$92\% \pm 7,84\%$	6%
Rétractation Symbolique	0%	0%	10%	$90\% \pm 8,48\%$

Malheureusement, les gestes non-appris ne peuvent toujours pas être classifiés par la sortie globale du GIOHMM. En revanche, l'extension générative du modèle nous permet de calculer une vraisemblance pour différencier les gestes appris des autres.

Rejet d'une classe inconnue par le GIOHMM

En l'absence de gestes supplémentaires pour évaluer les performances de rejet, j'utilise des trajectoires de chiffres manuscrits[4] (Figure 5.35). Ces dernières possèdent une grande variété de formes et de mouvements. Pour chaque classe de chiffres, toutes les séquences sont présentées au GIOHMM ayant appris les quatre gestes. Le GIOHMM calcule, à chaque instant t pour une séquence d'entrée $\mathbf{u}_1^t$, la Vlikelihood (Equation 4.34).

[4]PenDigits, Machine Learning Databases ftp://ftp.ics.uci.edu/pub/machine-learning-databases

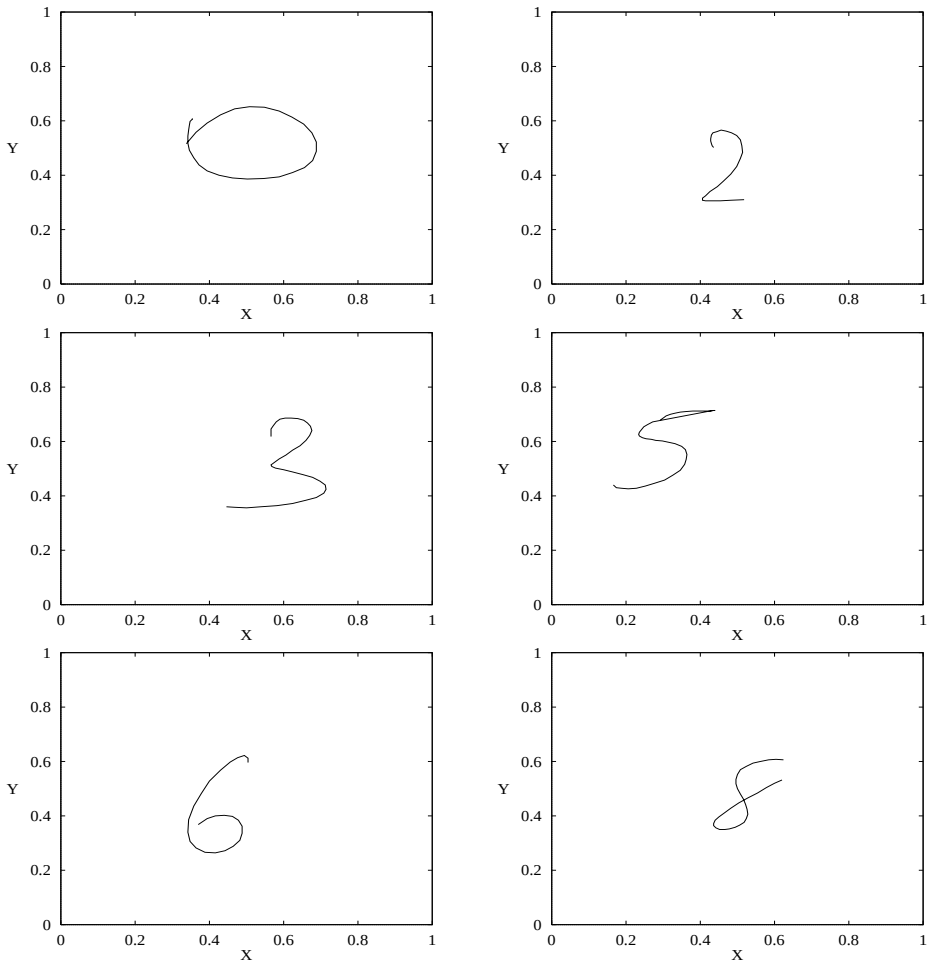


Figure 5.35: Exemples de chiffres manuscrits.

Les séquences, dont la Vlikelihood est nulle ou dont le log de la Vlikelihood est inférieur à un seuil, sont rejetées. Le seuil est déterminé sur les trajectoires de gestes de l'ensemble de test.

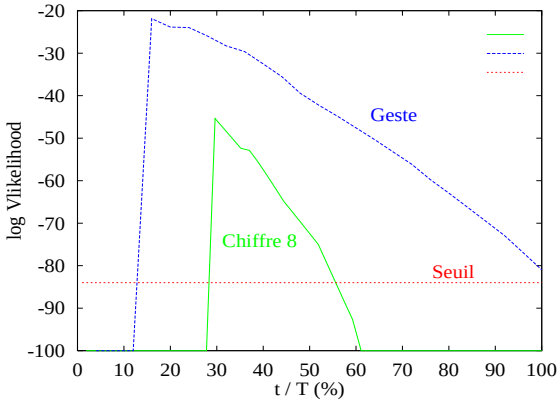


Figure 5.36: Le log de la Vlikelihood d'un geste et d'un chiffre manuscrit.

On remarque que le log de la Vlikelihood décroît plus lentement pour un geste que pour un chiffre manuscrit (Figure 5.36). Il est donc possible de discriminer une séquence observée d'une séquence inconnue. Le résultat du rejet des chiffres manuscrits est présenté (Figure 5.35).

Tableau 5.10: Taux de rejet des chiffres manuscrits par le GIOHMM à la fin de la séquence.

0	97,3%	±2,3%
1	79,78%	±5,7%
2	87,7%	±4,7%
3	95,69%	±2,9%
4	96,56%	±2,5%
5	95,6%	±2,97%
6	87,64%	±4,8%
7	98,55%	±1,62%
8	100%	
9	94,9%	±3,25%

Les résultats de rejet des chiffres manuscrits sont, dans l'ensemble, très satisfaisants. On constate que les plus mauvais résultats sont obtenus pour les chiffres 1, 2 et 6. La principale cause à cela est certainement leur ressemblance avec l'une des trajectoires de geste appris. Cependant, nous ne possédons pas de résultats comparables pour pouvoir apprécier réellement ceux obtenus.

5.7 Application à un système de reconnaissance des gestes de la main

5.7.1 GHOST: une interface gestuelle mono-utilisateur

GHOST[5] est un système de reconnaissance des gestes de la main basé sur la technologie LISTEN. Ce système est composé de deux modules indépendants. Le premier utilise la reconnaissance de postures de la main pour exécuter une commande. Afin de détecter l'intention de l'utilisateur d'exécuter une commande, des fenêtres actives sont définies dans l'espace corporel. Quand un blob entre dans une fenêtre active, la reconnaissance des postures de la main, utilisant un CGM spécifique pour chaque posture, est déclenchée. Le second de ces modules réalise la reconnaissance des gestes de la main grâce au GIOHMM.

A l'heure actuelle, les deux modules n'ont pas été intégrés ensemble, mais ils peuvent être assemblés très rapidement. La reconnaissance des gestes par le GIOHMM

[5]GHOST: "Gesturing Hand recOgnition based on uSer Tracking".

peut se faire en temps réel (seul la trajectoire 2D est utilisée), mais la construction et le suivi des blobs doivent être correctement réalisés.

Je présente, tout d'abord, la construction de l'espace corporel de l'utilisateur. Cet espace est nécessaire pour pouvoir repérer les mains et normaliser les gestes. Enfin, le principe du fonctionnement de GHOST est illustré dans son intégralité à travers une session d'utilisation.

5.7.2 Un espace corporel centré sur le visage

L'objet de cette partie est d'introduire une technique de segmentation spatiale de l'espace corporel : l'espace corporel centré sur le visage.

a. Principe

Dans le but d'aider à segmenter les gestes, à reconnaître les postures de main et à normaliser les trajectoires, nous définissons des régions dans l'espace corporel de l'utilisateur. Ce modèle s'inspire de l'espace discret pour la localisation des mains [McNeill, 1992]. Le plus souvent, un geste commence par une position de repos (Figure 5.37 a), continue par le corps du geste (Figure 5.37 b) et se termine par un retour à une position de repos (Figure 5.37 c) [Kendon et al., 1986]. Ainsi, dans notre espace, lorsque la personne est droite et face à la caméra, on peut repérer les mains au repos généralement dans la périphérie et l'extrême périphérie inférieure (cf. Chapitre 2, Figure 2.17).

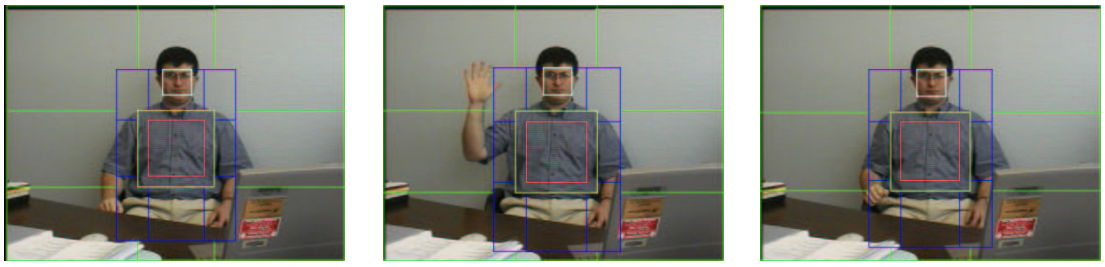


Figure 5.37: De gauche à droite, (a) Position de repos initiale, (b) Geste de commande, (c) Position de repos finale.

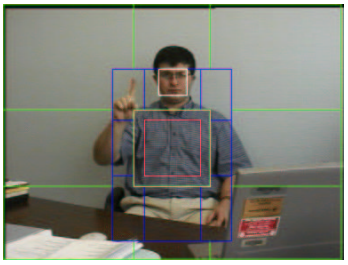


Figure 5.38: Commande de sélection.

Les régions forment des zones d'activités gestuelles dans lesquelles on peut mesurer l'activité des gestes et des postures. En effet, on peut établir les fréquences d'apparition des postures dans chaque zone et même calculer les probabilités de transitions entre les zones. De plus, afin de pouvoir détecter efficacement l'intention de l'utilisateur de s'adresser au système, je réserve une partie de l'espace corporel (fenêtre active) comme espace de travail où les postures seront seulement reconnues.

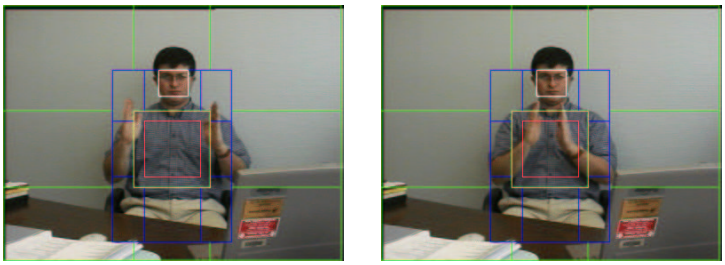


Figure 5.39: Expression d'une envergure.

Par exemple, on peut situer la fenêtre active d'un geste de commande à la gauche (ou la droite) des épaules dans la périphérie et l'extrême périphérie supérieure (Figure 5.37 (b) et 5.38), et on peut situer la fenêtre active d'un geste déictique à la gauche (ou la droite) de l'espace corporel. Les gestes bi-manuels (expression d'une envergure par exemple) se situent généralement dans la périphérie gauche ou droite, dans le centre et l'hyper-centre (Figure 5.39).

b. Construction

Nous projetons donc sur l'utilisateur un espace corporel basé sur une "discrétisation de l'espace corporel pour l'emplacement des mains" [McNeill, 1992] et centré sur le visage de l'utilisateur détecté par LISTEN.

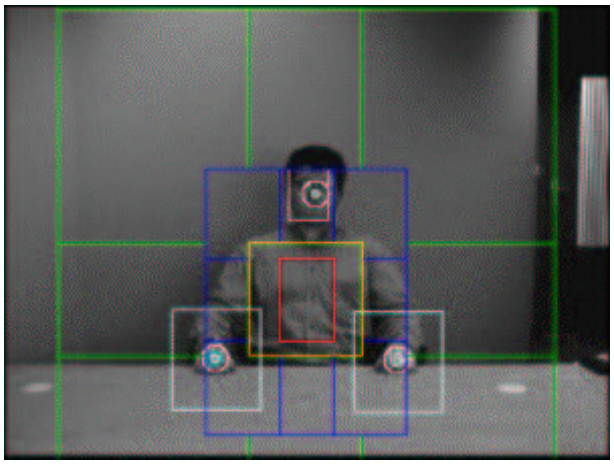


Figure 5.40: L'espace corporel centré sur le visage dans une image.

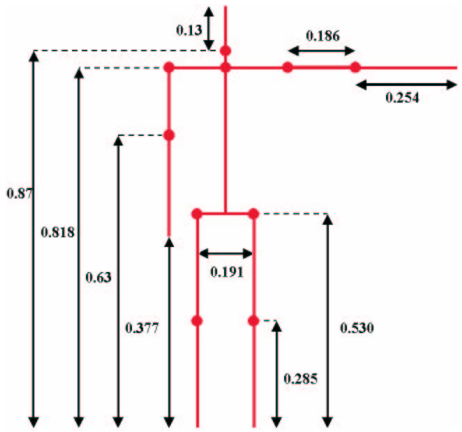


Figure 5.41: Modèle anthropométrique du corps.

L'espace corporel centré sur le visage (Figure 5.40) est construit en utilisant un modèle anthropométrique du corps exprimé en fonction de la hauteur totale de l'utilisateur (Figure 5.41). La hauteur totale est elle-même calculée à partir de la hauteur du visage.

5.7.3 Le principe de fonctionnement de GHOST

Le fonctionnement du système GHOST se déroule en plusieurs étapes. GHOST commence par acquérir des images. Il segmente les régions d'intérêt et il détecte le visage de l'utilisateur. Puis, il construit un espace corporel, centré sur le visage détecté, dans lequel il va suivre, d'une image à l'autre, les blobs qui ne sont pas des visages. Enfin, le système GHOST est capable de reconnaître si la trajectoire d'un blob est un geste de la main. Finalement, une application de reconnaissance et d'interprétation des postures de la main est proposée. GHOST reconnaît quatre des six postures de la main précédemment présentées (A, B, Pointe, Compte5) et exécute une commande associée à la posture (Fermeture de fenêtre, Exécution de programmes).

Figure 5.42: Images à analyser.

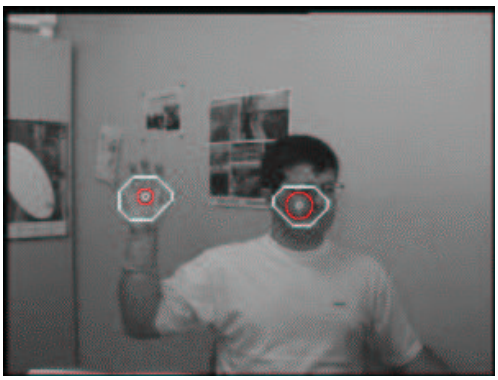


Figure 5.43: Représentation des blobs sur le visage et la main.

GHOST commence donc par capturer des images de type vidéo personnelle ou de type visioconférence (Figures 5.42).

Avant d'analyser ces images, GHOST recherche des régions d'intérêts (blobs) autour du visage et des mains (Figure 5.43). Il détecte la position et la taille du visage dans les blobs (Figure 5.44) et construit un espace corporel à partir du visage détecté (Figure 5.45).

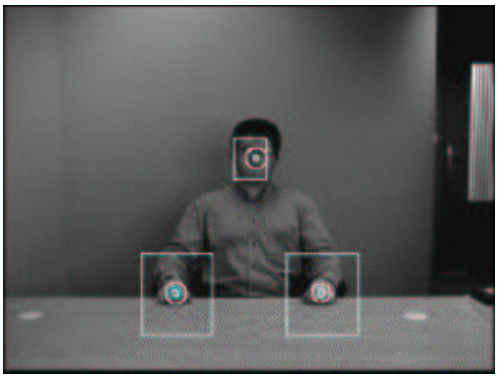


Figure 5.44: Détection du visage dans les blobs.

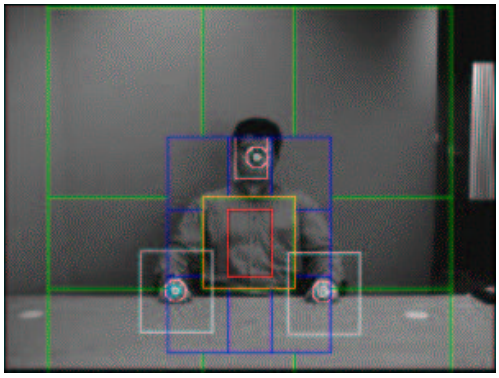


Figure 5.45: Représentation de l'espace corporel centré sur le visage.

Puis, GHOST suit les blobs qui ne sont pas des visages et utilise le GIOHMM pour reconnaître si la trajectoire du blob est un geste de la main (Figure 5.46).

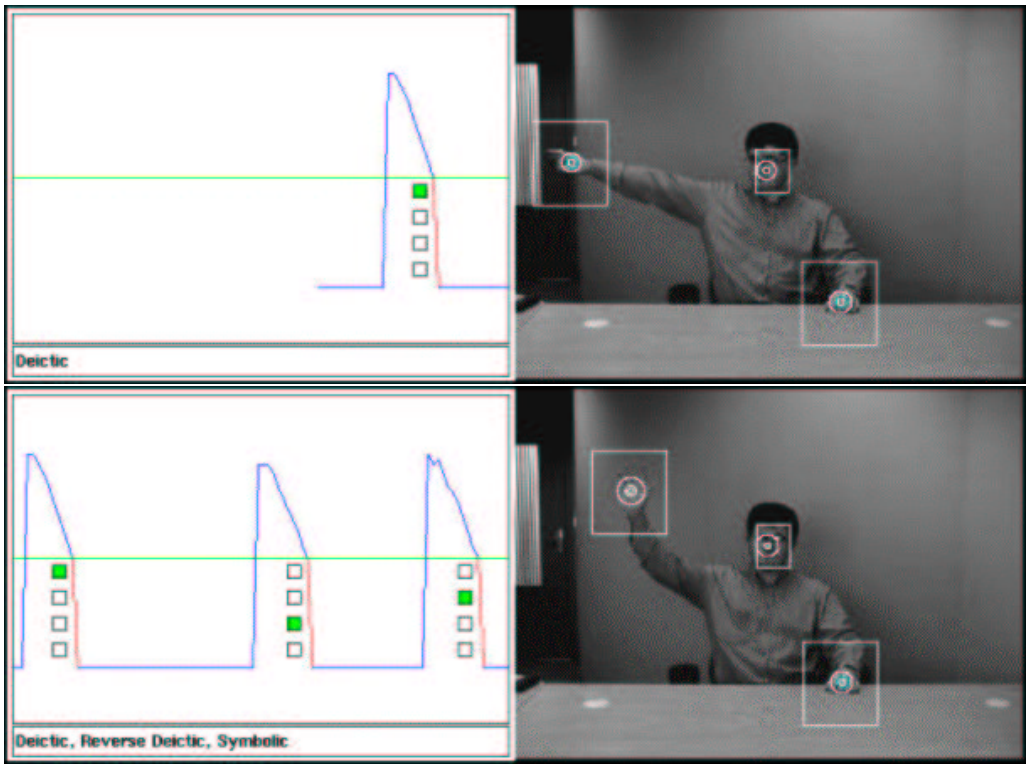


Figure 5.46: Reconnaissance d'un geste déictique (en haut) et d'un geste symbolique (en bas). La Vlikelihood (à gauche) indique la fin du geste et le marqueur l'identifie.

Finalement, GHOST rend possible la commande manuelle de son ordinateur personnel en détectant les postures de la main. L'utilisateur lève la main avec les doigts serrés pour réaliser la posture B et l'horloge s'affiche (Figure 5.47), il réalise la posture Pointe et la messagerie apparaît (Figure 5.48), il réalise la posture Compte5 et une vidéo s'enclenche (Figure 5.49).

Figure 5.47: L'utilisateur lève la main avec les doigts serrés et l'horloge s'affiche.

Figure 5.48: La messagerie s'affiche.

Figure 5.49: Une vidéo apparaît.

Figure 5.50: L'utilisateur serre le poing et les fenêtres se ferment.

Enfin, lorsque l'utilisateur ferme la main (la posture A est réalisée) toutes les fenêtres se ferment (Figure 5.50).

5.8 Discussion

Nous envisageons une application du système GHOST à la visioconférence, qui interpréte un geste de la main désignant un objet de la scène (tableau, autre personne, feuille) et qui effectue automatiquement la mise au point sur celui-ci. Il est également possible d'entrevoir une application de terminal individuel, où le système détecte et reconnaît des commandes gestuelles afin de déplacer, de modifier ou de déformer des objets virtuels ou distants.

Nous avons développé un système temps réel de reconnaissance de gestes de la main à partir de l'image fournie par une caméra. Ce système est basé sur un terminal individuel du type LISTEN, et reconnaît de façon fiable un vocabulaire de postures de mains et les gestes déictiques et symboliques.

Les prototypes actuels d'interfaces gestuelles basées sur l'image sont limités. En majorité, ce sont des applications "jouets" non utilisées en environnement courant. Par ailleurs, les systèmes existants prennent rarement en compte les interactions bimanuelles (main dominante et main dominée) [Guiard, 1987] [Buxton et Myers, 1986] [Kabbash et al., 1993] [Kabbash et al., 1994] alors que les gestes utilisent souvent les deux mains. Ultérieurement, nous tenterons d'étendre le système aux gestes pictographiques, kinégraphiques, puis aux gestes bi-manuels.

CHAPITRE 6

Conclusion et perspectives

Cette thèse s'est orientée sur deux aspects du traitement de l'information : l'aspect statique et l'aspect dynamique. Elle propose une approche générative pour le traitement des données et des séquences de données en général, et pour le traitement des images et des séquences d'images en particulier.

Les techniques proposées sont appliquées à la reconnaissance des gestes de la main en vision par ordinateur pour construire des interfaces homme-machine évoluées. Les gestes illustrent parfaitement l'aspect statique et l'aspect dynamique. En effet, il a été montré que les gestes de la main se distinguent par la ou les postures de la main exécutées et par la trajectoire de la main dans l'espace corporel.

D'un côté, les réseaux de neurones artificiels sont très efficaces pour des tâches de reconnaissance de formes, mais sont mal adaptés à la reconnaissance de données de nature temporelle. Nous avons montré l'utilisation d'un modèle génératif de réseau de neurones pour la reconnaissance statique de postures de la main dans les images.

D'un autre côté, les modèles de Markov cachés sont bien adaptés à la modélisation des séquences temporelles, mais ne permettent pas de discriminer efficacement des séquences de classes distinctes. Ainsi, un modèle hybride (GIOHMM : IOHMM génératif), qui possède les propriétés des réseaux de neurones et des modèles de Markov cachés, a été présenté pour la reconnaissance dynamique des gestes de la main dans des séquences d'images. L'intérêt majeur du GIOHMM est, outre le fait d'être capable de classifier efficacement des séquences de données comme son homologue non-génératif (IOHMM), de pouvoir distinguer une séquence observée d'une séquence inconnue.

Un ensemble de modèles génératifs de réseaux de neurones permet de reconnaître six postures de la main dans les images, et un GIOHMM est en mesure de différencier quatre trajectoires gestuelles. De plus, le principe général d'un système intégrant ces deux techniques est présenté. Ce système offre un démonstrateur temps réel de reconnaissance de postures de la main pour déclencher ou fermer des programmes.

Dans cette thèse, d'autres problématiques ont été abordées, mais n'ont pas été approfondies. La segmentation du visage et des mains dans l'image est réalisée par un modèle probabiliste gaussien, où les éléments de ce modèle (les blobs) correspondent aux objets de teinte chair de forte densité spatiale et chromatique. D'une part, dans ce modèle, il est nécessaire de s'assurer que le fond n'est pas de teinte chair.

D'autre part, ce modèle probabiliste est basé sur un nombre de blobs fixe. Or, a priori, on ne connaît pas le nombre de blobs, i.e. le nombre d'objets (visages, mains) présents dans l'image. Ainsi, il serait très intéressant d'étudier un nouveau modèle où le nombre de blobs n'est plus fixé, mais estimé dynamiquement.

Le modèle du GIOHMM a été appliqué à la reconnaissance d'un nombre limité de gestes de la main non-locaux. Il est important pour l'avenir d'envisager d'étendre le modèle à un plus grand nombre de gestes de la main. Il faudra notamment être en mesure de reconnaître des gestes locaux (enchaînements de postures) aussi bien que des gestes non-locaux.

Plus généralement, le GIOHMM peut être appliqué à la reconnaissance de l'écriture manuscrite ou à la reconnaissance de la parole. Dans ces domaines, les modèles de Markov cachés sont majoritairement utilisés et sont les méthodes les plus efficaces. Il convient donc de montrer l'intérêt majeur des IOHMM génératifs dans ces domaines.

Dans l'avenir, il est probable que nous disposerons d'interfaces homme-machine évoluées dans notre vie professionnelle et dans notre vie privée. Les interfaces évoluées comme LINEA nous apparaîtront sous la forme d'un humanoïde (reconstruction 3D d'un personnage virtuel). Nous dialoguerons avec ce dernier qui reconnaîtra nos visages, nos paroles, nos gestes et nos attitudes. Relié à Internet, ce véritable assistant virtuel nous assistera en gérant la messagerie, anticipera nos désirs en nous proposant le nouveau film vidéo de notre acteur préféré. Et peut-être même qu'il nous fera, pourquoi pas, la conversation.

Cette vision du futur est très attirante. Mais, sommes-nous prêts à nous laisser observer, même par une machine ? Sommes-nous prêts à sacrifier notre intimité au profit d'un confort de vie ? Outre un changement des mentalités et des habitudes, il faudra être en mesure de pouvoir échapper à tout moment à la surveillance "bienveillante" de notre assistant. Il sera nécessaire également de sécuriser les informations de profils et de comportements nous concernant.

ANNEXE A — La rétropropagation du gradient de l'erreur

Soit la fonction de coût quadratique instantanée suivante :

$$C_p(W) = \sum_{r=1}^{R} (d_r(p) - s_r^L(p))^2$$

On cherche donc à dériver la fonction de coût instantanée par rapport à un poids w_{ij}.

$$\begin{aligned}
\frac{\partial C_p(W)}{\partial w_{ij}^l} &= \frac{\partial C_p(W)}{\partial a_j^l} . \frac{\partial a_j^l}{\partial w_{ij}^l} \\
&= \frac{\partial C_p(W)}{\partial a_j^l} . s_i^{l-1} \\
&= \frac{\partial C_p(W)}{\partial s_j^l} . \frac{\partial s_j^l}{\partial a_j^l} . s_i^{l-1} \\
&= \Phi_j^l . f'(a_j^l) . s_i^{l-1} \qquad \text{(A.1)}
\end{aligned}$$

On pose la fonction Φ_j^l telle que :

- pour $l = L$, on est sur la couche de sortie :

$$\begin{aligned}
\Phi_j^L &= \frac{\partial C_p(W)}{\partial s_j^L} \\
&= \frac{\partial \sum_{r=1}^{R} (d_r(p) - s_r^L(p))^2}{\partial s_j^L} \\
&= 2\,(s_j^L(p) - d_j(p)) \qquad \text{(A.2)}
\end{aligned}$$

  Ainsi, on calcule pour chaque neurone de sortie j, la différence entre la sortie obtenue $s_j^L(p)$ et la sortie désirée $d_j(p)$ associée à l'exemple p.

- pour $l \neq L$, on est dans une couche cachée :

$$\Phi_j^l = \frac{\partial C_p(W)}{\partial s_j^l}$$

$$
\begin{aligned}
&= \sum_{k=1}^{N_{l+1}} \frac{\partial C_p(W)}{\partial a_k^{l+1}} \frac{\partial a_k^{l+1}}{\partial s_j^l} \\
&= \sum_{k=1}^{N_{l+1}} \frac{\partial C_p(W)}{\partial a_k^{l+1}} \frac{\partial \sum_{i=1}^{N_l} w_{ik}^{l+1} s_i^l}{\partial s_j^l} \\
&= \sum_{k=1}^{N_{l+1}} \frac{\partial C_p(W)}{\partial a_k^{l+1}} w_{jk}^{l+1} \\
&= \sum_{k=1}^{N_{l+1}} \frac{\partial C_p(W)}{\partial s_k^{l+1}} \frac{\partial s_k^{l+1}}{\partial a_k^{l+1}} w_{jk}^{l+1} \\
&= \sum_{k=1}^{N_{l+1}} \Phi_k^{l+1} f'(a_k^{l+1}) w_{jk}^{l+1} \qquad \text{(A.3)}
\end{aligned}
$$

Ainsi, Φ_j^l peut être calculé par induction à partir des couches suivantes.

Le calcul du gradient et la modification des connexions sont réalisés par récurrence depuis les neurones de sorties vers les neurones d'entrées, dans le sens inverse de la propagation, d'où le terme de rétropropagation du gradient de l'erreur. Pour chaque connexion, l'adaptation se fait à l'aide de la règle delta généralisée qui n'est rien d'autre qu'une descente du gradient :

$$w_{ij,t+1}^l = w_{ij,t}^l - \eta . \Delta w_{ij,t}^l$$

où η est appelé le pas d'apprentissage, et $\Delta w_{ij,t}^l$ est défini par :

$$\Delta w_{ij,t}^l = \begin{cases} l = L & : \quad f'(a_j^l)\, s_i^{l-1}\, 2\,(s_j^L(p) - d_j(p)) \\ l \neq L & : \quad f'(a_j^l)\, s_i^{l-1}\, \sum_{k=1}^{N_{l+1}} \Phi_k^{l+1} f'(a_k^{l+1}) w_{jk}^{l+1} \end{cases}$$

ANNEXE B L'algorithme EM

B.1 Convergence de l'algorithme EM

Nous allons donner dans cette annexe la démonstration de la convergence de l'algorithme **EM** dans le cas général vers un extremum local. En effet, **EM** ne garantit pas la convergence vers un extremum global.

B.1.1 Enoncé du problème

Soit μ une mesure et f une fonction que l'on suppose positive, on introduit alors la fonction g telle que :

$$g(\theta) = \int f(x, \theta)\, \mu(dx) \tag{B.1}$$

où θ est une variable qui appartient à l'ensemble Θ et $g(\theta)$ est non nulle pour tout $\theta \in \Theta$.

On cherche à maximiser g par rapport à θ. Pour cela, nous allons utiliser l'algorithme **EM**, un algorithme déterministe et itératif. Connaissant $\theta^{(i)}$, le paramètre à l'itération i, on cherche à déterminer $\theta^{(i+1)}$ tel que :

$$\theta^{(i+1)} = T\left(\theta^{(i)}\right) \quad \text{et} \quad g\left(\theta^{(i+1)}\right) \geq g\left(\theta^{(i)}\right)$$

On pose :

$$p(x, \theta) = \frac{f(x, \theta)}{g(\theta)}$$

Puis, on introduit la fonction $Q(\theta, \theta')$ définie par :

$$Q(\theta, \theta') = \int (\log f(x, \theta))\, p(x, \theta')\ \mu(dx)$$

où θ' est $\theta^{(i)}$ et θ est $\theta^{(i+1)}$.

Une itération de l'algorithme EM est composée de l'étape E, suivie de l'étape M.

- La première étape de l'algorithme, appelée étape E, est l'étape d'estimation de la fonction Q.

- La deuxième étape, étape M, est la suivante :

$$\theta^{(i+1)} = \quad \text{Arg}\max_{\theta} Q\left(\theta, \theta^{(i)}\right)$$

A présent, nous allons montrer que cet algorithme converge vers les maxima locaux de g. La démonstration se décompose de la manière suivante : tout d'abord, on montre d'abord que g est une fonction croissante des $\theta^{(i)}$, puis qu'un extremum local de Q en est un pour g.

B.1.2 Montrons la monotonie de g

$$\begin{aligned} Q(\theta, \theta') &= \int (\log f(x,\theta))\, p(x,\theta')\ \mu(dx) \\ &= \log\, g(\theta) + \int (\log p(x,\theta))\, p(x,\theta')\ \mu(dx) \end{aligned} \tag{B.2}$$

Ainsi :

$$Q(\theta,\theta') - Q(\theta',\theta') = \log\, g(\theta) - \log\, g(\theta') \tag{B.3}$$

$$+ \int \left(\log \frac{p(x,\theta)}{p(x,\theta')}\right) p(x,\theta')\, \mu(dx) \tag{B.4}$$

Le deuxième terme (B.4) est l'opposé de la divergence de Kullback-Leibler, notée $KL(\theta \mid\mid \theta')$. Nous allons maintenant appliquer l'inégalité de Jensen (Annexe B.2) à la fonction log. On considère la variable aléatoire $Y = p(x,\theta)/p(x,\theta')$ distribuée selon $p(x,\theta)$.

De l'inégalité de Jensen, on déduit que :

$$\log\left(E[Y]\right) \geq E[\log Y]$$

Or :

$$\begin{aligned} \log\left(E[Y]\right) &= \log\left(\int Y p(x,\theta')\ \mu(dx)\right) \\ &= \log\left(\int p(x,\theta)\, \mu(dx)\right) \\ &= \log 1 = 0 \end{aligned} \tag{B.5}$$

Et :

$$E[\log Y] = KL(\theta \mid\mid \theta')$$

Ce qui implique que :

$$KL(\theta \mid\mid \theta') \leq 0$$

De plus, on cherche à maximiser $Q(\theta,\theta')$. Par conséquent :

$$Q(\theta,\theta') - Q(\theta',\theta') \geq 0 \quad \Longrightarrow \quad \log g(\theta) - \log g(\theta') \geq 0$$

On a prouvé que la fonction g est monotone.

B.1.3 Montrons qu'un extremum local de Q est un extremum local de g

Soit θ' un extremum local de $Q(.,\theta')$, comme :

$$Q(\theta,\theta') = \log g(\theta) + \int (\log p(x,\theta))\, p(x,\theta')\, \mu(dx)$$

On en déduit que (en supposant que les conditions pour dériver sous le signe intégrale soient vérifiées) :

$$\begin{aligned} \left.\frac{\partial Q(\theta,\theta')}{\partial\theta}\right|_{\theta=\theta'} &= \left.\frac{\partial \log g(\theta)}{\partial\theta}\right|_{\theta=\theta'} \\ &+ \int \left(\left.\frac{\partial \log p(x,\theta)}{\partial\theta}\right|_{\theta=\theta'} \right) p(x,\theta')\, \mu(dx) \qquad \text{(B.6)} \\ &= 0 \end{aligned}$$

Nous allons maintenant utiliser la formule de Fisher (Annexe B.3) pour montrer que :

$$\left.\frac{\partial \log g(\theta)}{\partial\theta}\right|_{\theta=\theta'} = 0$$

En utilisant la définition de g (Equation B.1), on a en dérivant sous le signe intégrale :

$$\begin{aligned} \frac{\partial g(\theta)}{\partial\theta} &= \int \frac{\partial f(x,\theta)}{\partial\theta}\, \mu(dx) \\ \Rightarrow \frac{\partial \log g(\theta)}{\partial\theta} g(\theta) &= \int \frac{\partial \log f(x,\theta)}{\partial\theta} f(x,\theta)\, \mu(dx) \end{aligned}$$

d'après la formule de Fisher appliquée aux fonctions f et g.

On en déduit donc :

$$\begin{aligned} \frac{\partial \log g(\theta)}{\partial\theta} &= \int \frac{\partial \log f(x,\theta)}{\partial\theta} p(x,\theta)\, \mu(dx) \\ &= \int \frac{\partial(\log g(\theta) + \log p(x,\theta))}{\partial\theta} p(x,\theta)\, \mu(dx) \end{aligned}$$

D'où :

$$\int \frac{\partial \log f(x,\theta)}{\partial\theta} p(x,\theta)\, \mu(dx) = \frac{\partial \log g(\theta)}{\partial\theta} + \int \frac{\partial \log p(x,\theta)}{\partial\theta} p(x,\theta)\, \mu(dx)$$

Donc :

$$\int \frac{\partial \log p(x,\theta)}{\partial\theta} p(x,\theta)\, \mu(dx) = 0$$

Et par conséquent d'après l'équation B.6:

$$\left.\frac{\partial \log g(\theta)}{\partial\theta}\right|_{\theta=\theta'} = 0$$

Le deuxième point de la démonstration est donc prouvé.

On a ainsi montré la convergence de l'algorithme EM.

B.2 Inégalité de Jensen

B.2.1 Propriété

Soit φ une fonction convexe et X une variable aléatoire. Alors, l'inégalité de Jensen est la suivante :

$$\varphi\left(E\left[X\right]\right) \leq E\left[\varphi(X)\right]$$

On suppose que ces espérances existent et soient finies.

B.2.2 Preuve

Nous allons écrire le développement en série de Taylor de φ au second ordre :

$$\exists \epsilon \in]x, y[\quad \text{tel que} \quad \varphi(x) = \varphi(y) + \varphi'(y)(x-y) + \frac{\varphi''(\epsilon)(x-y)^2}{2}$$

Or φ est une fonction convexe, donc sa dérivée seconde est positive en tout point et on a l'inégalité suivante :

$$\varphi(y) + \varphi'(y)(x-y) \leq \varphi(x)$$

Si on prend $x = X$ et $y = E[X]$, on obtient alors :

$$\varphi\left(E[X]\right) + \varphi'\left(E[X]\right)\left(X - E[X]\right) \leq \varphi(X)$$

Il suffit pour finir de prendre l'espérance de chaque membre de l'inégalité :

$$E\left[\varphi\left(E[X]\right)\right] + E\left[\varphi'\left(E[X]\right)\left(X - E[X]\right)\right] \leq E\left[\varphi(X)\right]$$

Or :

$$\begin{aligned}
& E\left[\varphi\left(E[X]\right)\right] + E\left[\varphi'\left(E[X]\right)\left(X - E[X]\right)\right] \\
= \;& \varphi(E[X]) + E[\varphi'(E[X])\, X] - E[\varphi'(E[X])\, E[X]] \\
= \;& \varphi(E[X]) + \varphi'(E[X])\, E[X] - \varphi'(E[X])\, E[E[X]] \\
= \;& \varphi(E[X]) + \varphi'(E[X])\, E[X] - \varphi'(E[X])\, E[X] \\
= \;& \varphi\left(E[X]\right)
\end{aligned}$$

Par conséquent :

$$\varphi\left(E[X]\right) \leq E\left[\varphi(X)\right]$$

L'inégalité est ainsi montrée.

B.3 Formule de Fisher

Soit f une fonction continue et dérivable sur son ensemble de définition D_f.

On a l'égalité suivante bien connue :

$$\forall x \in D_f \quad \frac{\partial \log f(x)}{\partial x} = \frac{f'(x)}{f(x)}$$

La formule de Fisher consiste à écrire :

$$\forall x \in D_f \quad f'(x) = f(x)\frac{\partial \log f(x)}{\partial x}$$

ANNEXE C

Application de l'algorithme EM aux HMM

C.1 Algorithme Forward-Backward

C.1.1 Etape Forward

On calcule la variable $\alpha_{j,t}$ telle que :

$$\begin{aligned}
\alpha_{j,t} &= Pr(\mathbf{o}_1^t, x_t = j \mid \Lambda) \\
&= \sum_{i=1}^{N} Pr(\mathbf{o}_1^t, x_t = j, x_{t-1} = i \mid \Lambda) \\
&= \sum_{i=1}^{N} Pr(\mathbf{o}_t \mid \mathbf{o}_1^{t-1}, x_t = j, x_{t-1} = i, \Lambda)\, Pr(\mathbf{o}_1^{t-1}, x_t = j, x_{t-1} = i \mid \Lambda) \\
&= \sum_{i=1}^{N} Pr(\mathbf{o}_t \mid x_t = j, \Lambda)\, Pr(x_t = j \mid \mathbf{o}_1^{t-1}, x_{t-1} = i, \Lambda)\, Pr(\mathbf{o}_1^{t-1}, x_{t-1} = i \mid \Lambda) \\
&= Pr(\mathbf{o}_t \mid x_t = j, \Lambda) \sum_{i=1}^{N} Pr(x_t = j \mid x_{t-1} = i, \Lambda)\, Pr(\mathbf{o}_1^{t-1}, x_{t-1} = i \mid \Lambda)
\end{aligned}$$

Ainsi, on obtient :

$$\alpha_{j,t} = b_j(\mathbf{o}_t) \sum_{i=1}^{N} a_{ij}\, \alpha_{i,t-1}$$

C.1.2 Etape Forward

Le calcul de $\hat{\beta}_{j,t}$ est le suivant :

$$\begin{aligned}
\hat{\beta}_{j,t} &= Pr(\mathbf{o}_{t+1}^T \mid x_t = j, \hat{\Lambda}) \\
&= \sum_{i=1}^{N} Pr(\mathbf{o}_{t+1}^T, x_{t+1} = i \mid x_t = j, \hat{\Lambda})
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{i=1}^{N} Pr(\mathbf{o}_{t+1} \mid \mathbf{o}_{t+2}^{T}, x_{t+1}=i, x_t=j, \hat{\Lambda}) Pr(\mathbf{o}_{t+2}^{T}, x_{t+1}=i \mid x_t=j, \hat{\Lambda}) \\
&= \sum_{i=1}^{N} Pr(\mathbf{o}_{t+1} \mid x_{t+1}=i, \hat{\Lambda})\, Pr(\mathbf{o}_{t+2}^{T} \mid x_{t+1}=i, x_t=j, \hat{\Lambda})\, Pr(x_{t+1}=i \mid x_t=j, \hat{\Lambda}) \\
&= \sum_{i=1}^{N} Pr(\mathbf{o}_{t+1} \mid x_{t+1}=i, \hat{\Lambda})\, Pr(\mathbf{o}_{t+2}^{T} \mid x_{t+1}=i, \hat{\Lambda})\, Pr(x_{t+1}=i \mid x_t=j, \hat{\Lambda})
\end{aligned}
$$

Finalement, on obtient :

$$\hat{\beta}_{j,t} = \sum_{i=1}^{N} b_i(\mathbf{o}_{t+1})\, \hat{\beta}_{i,t+1}\, a_{ji}$$

C.2 Algorithme EM

C.2.1 Etape E

On calcule $\hat{h}_{ij,t}$ tel que:

$$
\begin{aligned}
\hat{h}_{ij,t} &= Pr(x_t=j, x_{t-1}=i \mid \mathbf{o}_1^T, \hat{\Lambda}) \\
&= \frac{Pr(x_t=j, x_{t-1}=i, \mathbf{o}_1^T \mid \hat{\Lambda})}{Pr(\mathbf{o}_1^T \mid \hat{\Lambda})} \\
&= \frac{Pr(\mathbf{o}_{t+1}^T \mid x_t=j, x_{t-1}=i, \mathbf{o}_1^t, \hat{\Lambda})\, Pr(x_t=j, x_{t-1}=i, \mathbf{o}_1^t \mid \hat{\Lambda})}{Pr(\mathbf{o}_1^T \mid \hat{\Lambda})} \\
&= Pr(\mathbf{o}_{t+1}^T \mid x_t=j, \hat{\Lambda})\, Pr(\mathbf{o}_t \mid x_t=j, x_{t-1}=i, \mathbf{o}_1^{t-1}, \hat{\Lambda}) \\
&\quad Pr(x_t=j, x_{t-1}=i, \mathbf{o}_1^{t-1} \mid \hat{\Lambda}) \,/\, Pr(\mathbf{o}_1^T \mid \hat{\Lambda}) \\
&= \hat{\beta}_{j,t}\, Pr(\mathbf{o}_t \mid x_t=j, \hat{\Lambda})\, Pr(x_t=j \mid \mathbf{o}_1^{t-1}, x_{t-1}=i, \hat{\Lambda}) \\
&\quad Pr(x_{t-1}=i, \mathbf{o}_1^{t-1} \mid \hat{\Lambda}) \,/\, Pr(\mathbf{o}_1^T \mid \hat{\Lambda}) \\
&= \frac{\hat{\beta}_{j,t}\, b_j(\mathbf{o}_t)\, Pr(x_t=j \mid x_{t-1}=i, \hat{\Lambda})\, \hat{\alpha}_{i,t-1}}{Pr(\mathbf{o}_1^T \mid \hat{\Lambda})} \\
&= \frac{\hat{\beta}_{j,t}\, b_j(\mathbf{o}_t)\, a_{ij}\, \hat{\alpha}_{i,t-1}}{Pr(\mathbf{o}_1^T \mid \hat{\Lambda})}
\end{aligned}
$$

$$\hat{h}_{ij,t} = \frac{\hat{\beta}_{j,t}\, b_j(\mathbf{o}_t)\, a_{ij}\, \hat{\alpha}_{i,t-1}}{Pr(\mathbf{o}_1^T \mid \hat{\Lambda})}$$

C.2.2 Etape M

Les dérivées partielles a calculer sont : $\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial \pi_i}$, $\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial a_{ij}}$ et $\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial b_j(k)}$.

- Commençons par π_j :

$$\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial \pi_j} = \frac{\partial \sum_p^P \sum_{t=1}^T \sum_j^N \sum_i^N \hat{h}_{ij,t}\, log\, Pr(x_t = j \mid x_{t-1} = i, \Lambda)}{\partial \pi_j}$$

pour $t = 1$ l'équation 3.38 nous donne $Pr(x_t = j \mid x_{t-1}, \Lambda) = Pr(x_t = j \mid \Lambda)$ d'où:

$$\begin{aligned}
\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial \pi_j} &= \partial\Big(\sum_p^P \Big[\sum_j^N \hat{g}_{j,1}\, log\, Pr(x_1 = j \mid \Lambda) \\
&+ \sum_{t=2}^T \sum_j^N \sum_i^N \hat{h}_{ij,t}\, log\, Pr(x_t = j \mid x_{t-1} = i, \Lambda)\Big]\Big)/\partial \pi_j \\
&= \sum_p^P \hat{g}_{j,1} \frac{\partial log\, Pr(x_1 = j \mid \Lambda)}{\partial \pi_j} \\
&= \sum_p^P \frac{\hat{g}_{j,1}}{\pi_j}
\end{aligned}$$

On utilise la méthode des multiplicateurs de Lagrange pour résoudre l'équation ci-dessus. Soit $f(.)$ la fonction $Q(\Lambda,\hat{\Lambda})$ et $g(.)$ la fonction de contrainte (3.42) alors, au maximum de Q sous la contrainte donnée par g :

$$\begin{aligned}
&\nabla f = \lambda \nabla g \\
\Leftrightarrow\ &\sum_p^P \frac{\hat{g}_{j,1}}{\pi_j} = \lambda \frac{\partial(\sum_{i=1}^N \pi_i - 1)}{\partial \pi_j} \\
\Leftrightarrow\ &\pi_j = \frac{\sum_p^P \hat{g}_{j,1}}{\lambda} \qquad \text{(C.1)}
\end{aligned}$$

Or, on a $\sum_{i=1}^N \pi_i = 1$ (Equation 3.42) d'où:

$$\begin{aligned}
&\sum_{i=1}^N \frac{\sum_p^P \hat{g}_{j,1}}{\lambda} = 1 \\
\Leftrightarrow\ &\lambda = \sum_{i=1}^N \sum_p^P \hat{g}_{i,1} \qquad \text{(C.2)}
\end{aligned}$$

ainsi les équations (C.1) et (C.2) nous donnent :

$$\boxed{\pi_j = \frac{\sum_p^P \hat{g}_{j,1}}{\sum_{i=1}^N \sum_p^P \hat{g}_{i,1}}}$$

- Continuons par a_{ij} :

$$\frac{\partial Q(\Lambda, \hat{\Lambda})}{\partial a_{ij}} = \sum_{p}^{P} \sum_{t}^{T} \frac{\hat{h}_{ij,t}}{a_{ij}}$$

On pose $f(.)$ la fonction $Q(\Lambda, \hat{\Lambda})$ et $g(.)$ la fonction de contrainte (3.43) alors, au maximum de Q sous la contrainte donnée par g :

$$\begin{aligned} & \nabla f = \lambda_i \nabla g \\ \Leftrightarrow \quad & \sum_{p}^{P} \sum_{t}^{T} \frac{\hat{h}_{ij,t}}{a_{ij}} = \lambda_i \frac{\partial(\sum_{k=1}^{N} a_{ik} - 1)}{\partial a_{ij}} \\ \Leftrightarrow \quad & a_{ij} = \frac{\sum_p^P \sum_t^T \hat{h}_{ij,t}}{\lambda_i} \end{aligned} \tag{C.3}$$

Or, on a $\sum_{k=1}^{N} a_{ik} = 1$ (Equation 3.43) d'où:

$$\begin{aligned} & \sum_{k=1}^{N} \frac{\sum_p^P \sum_t^T \hat{h}_{ik,t}}{\lambda_i} = 1 \\ \Leftrightarrow \quad & \lambda_i = \sum_{k=1}^{N} \sum_{p}^{P} \sum_{t}^{T} \hat{h}_{ik,t} \end{aligned} \tag{C.4}$$

ainsi les équations (C.3) et (C.4) nous donnent :

$$\boxed{a_{ij} = \frac{\sum_p^P \sum_t^T \hat{h}_{ij,t}}{\sum_{k=1}^{N} \sum_p^P \sum_t^T \hat{h}_{ik,t}}}$$

- Dans le cas où les observations sont discrètes :

$$\frac{\partial Q(\Lambda, \hat{\Lambda})}{\partial b_j(k)} = \sum_{p}^{P} \sum_{t}^{T} \frac{\hat{g}_{j,t}}{b_j(k)} \delta_{\mathbf{o}_t k}$$

où δ_{ij} est le symbole de Kronecker :

$$\delta_{ij} = \begin{cases} 1 & : \quad i = j \\ 0 & : \quad i \neq j \end{cases} \tag{C.5}$$

On pose $f(.)$ la fonction $Q(\Lambda, \hat{\Lambda})$ et $g(.)$ la fonction de contrainte (3.44) alors, au maximum de la fonction Q sous la contrainte donnée par g:

$$\begin{aligned} & \nabla f = \lambda_j \nabla g \\ \Leftrightarrow \quad & \sum_{p}^{P} \sum_{t}^{T} \frac{\hat{g}_{j,t}}{b_j(k)} \delta_{\mathbf{o}_t k} = \lambda_j \frac{\partial \sum_{d=1}^{D} b_j(d) - 1}{\partial b_j(k)} \\ \Leftrightarrow \quad & b_j(k) = \frac{\sum_p^P \sum_t^T \hat{g}_{j,t} \delta_{\mathbf{o}_t k}}{\lambda_j} \end{aligned} \tag{C.6}$$

Or, on a $\sum_{d=1}^{D} b_j(d) = 1$ (Equation 3.44) d'où:

$$\sum_{d=1}^{D} \frac{\sum_p^P \sum_t^T \hat{g}_{j,t} \delta_{\mathbf{o}_t d}}{\lambda_j} = 1$$

$$\Leftrightarrow \quad \lambda_j = \sum_{d=1}^{D} \sum_{p}^{P} \sum_{t}^{T} \hat{g}_{j,t} \delta_{\mathbf{o}_t d} \tag{C.7}$$

ainsi les équations (C.6) et (C.7) nous donnent :

$$\boxed{b_j(k) = \frac{\sum_p^P \sum_t^T \hat{g}_{j,t} \delta_{\mathbf{o}_t k}}{\sum_{d=1}^{D} \sum_p^P \sum_t^T \hat{g}_{j,t} \delta_{\mathbf{o}_t d}}}$$

- Poursuivons dans le cas où les observations sont continues, on suppose que les **pdf** sont des gaussiennes. On recherche les paramètres c_{jm}, $\boldsymbol{\mu}_{jm}$ et $\boldsymbol{\sigma}_{jm}$ de Λ de sorte à maximiser la fonction auxiliaire $Q(\Lambda, \hat{\Lambda})$. Pour cela, on calcule les dérivées partielles $\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial c_{jm}}$, $\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial \boldsymbol{\mu}_{jm}}$ et $\frac{\partial Q(\Lambda,\hat{\Lambda})}{\partial \boldsymbol{\sigma}_{jm}}$.

  - Commençons par dériver $Q(\Lambda, \hat{\Lambda})$ par rapport à c_{jm}. On utilise une nouvelle fois les multiplicateurs de Lagrange.

$$\frac{\partial Q(\Lambda, \hat{\Lambda})}{\partial c_{jm}} \quad = \quad \sum_{p}^{P} \sum_{t}^{T} \frac{\hat{g}_{j,t}}{b_j(\mathbf{o}_t)} \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm})$$

    On pose $f(.)$ la fonction $Q(\Lambda, \hat{\Lambda})$ et $g(.)$ la fonction de contrainte (3.45) alors, au maximum de Q sous la contrainte donnée par g :

$$\begin{aligned} & \nabla f = \lambda_j \nabla g \\ \Leftrightarrow \quad & \sum_{p}^{P} \sum_{t}^{T} \frac{\hat{g}_{j,t}}{b_j(\mathbf{o}_t)} \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm}) = \lambda_j \frac{\partial (\sum_{k=1}^{M} c_{jk} - 1)}{\partial c_{jm}} \\ \Leftrightarrow \quad & c_{jm} = \frac{\sum_p^P \sum_t^T \frac{\hat{g}_{j,t}}{b_j(\mathbf{o}_t)} \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm})}{\lambda_j} \end{aligned} \tag{C.8}$$

    Or, on a $\sum_{k=1}^{M} c_{jk} = 1$ (Equation 3.45) d'où:

$$\sum_{k=1}^{M} \frac{\sum_p^P \sum_t^T \frac{\hat{g}_{j,t}}{b_j(\mathbf{o}_t)} \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm})}{\lambda_j} = 1$$

$$\Leftrightarrow \quad \lambda_j = \sum_{k=1}^{K} \sum_{p}^{P} \sum_{t}^{T} \frac{\hat{g}_{j,t}}{b_j(\mathbf{o}_t)} \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jk}, \boldsymbol{\sigma}_{jk}) \tag{C.9}$$

    ainsi les équations (C.8) et (C.9) nous donnent :

$$c_{jm} = \frac{\sum_p^P \sum_t^T \frac{\hat{g}_{j,t}}{b_j(\mathbf{o}_t)} \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm})}{\sum_{k=1}^K \sum_p^P \sum_t^T \frac{\hat{g}_{j,t}}{b_j(\mathbf{o}_t)} \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jk}, \boldsymbol{\sigma}_{jk})}$$

- Continuons avec $\boldsymbol{\mu}_{jm}$. Cette fois-ci, nous n'utiliserons pas la méthode des multiplicateurs de Lagrange. En effet, il n'y a plus de contraintes sur $\boldsymbol{\mu}_{jm}$ et sur $\boldsymbol{\sigma}_{jm}$.

$$\begin{aligned}
\frac{\partial Q(\Lambda, \hat{\Lambda})}{\partial \boldsymbol{\mu}_{jm}} &= \sum_{p=1}^P \sum_{t=1}^T \frac{\hat{g}_{j,t}}{b_j(\mathbf{o}_t)} \frac{\partial c_{jm} \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm})}{\partial \boldsymbol{\mu}_{jm}} \\
&= -\frac{1}{2} \sum_{p=1}^P \sum_{t=1}^T \frac{\hat{g}_{j,t}\, c_{jm} \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm})}{b_j(\mathbf{o}_t)} \frac{\partial \left({}^T(\mathbf{o}_t - \boldsymbol{\mu}_{jm}) \boldsymbol{\sigma}_{jm} (\mathbf{o}_t - \boldsymbol{\mu}_{jm}) \right)}{\partial \boldsymbol{\mu}_{jm}} \\
&= \frac{1}{2} \sum_{p=1}^P \sum_{t=1}^T E_{jm} \left[\boldsymbol{\sigma}_{jm} (\mathbf{o}_t - \boldsymbol{\mu}_{jm}) + {}^T(\mathbf{o}_t - \boldsymbol{\mu}_{jm}) \boldsymbol{\sigma}_{jm} \right]
\end{aligned}$$

avec :

$$E_{jm} = \frac{\hat{g}_{j,t}\, c_{jm}\, \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm})}{b_j(\mathbf{o}_t)} \tag{C.10}$$

de plus, $\boldsymbol{\sigma}_{jm}$ est une matrice symétrique d'où :

$$\frac{\partial Q(\Lambda, \hat{\Lambda})}{\partial \boldsymbol{\mu}_{jm}} = \sum_{p=1}^P \sum_{t=1}^T E_{jm} \, (\mathbf{o}_t - \boldsymbol{\mu}_{jm}) \, \boldsymbol{\sigma}_{jm}$$

ainsi, si l'on cherche à annuler les dérivées :

$$\begin{aligned}
& \frac{\partial Q(\Lambda, \hat{\Lambda})}{\partial \boldsymbol{\mu}_{jm}} = 0 \\
\Leftrightarrow \quad & \sum_{p=1}^P \sum_{t=1}^T E_{jm} \, (\mathbf{o}_t - \boldsymbol{\mu}_{jm}) \, \boldsymbol{\sigma}_{jm} = 0 \\
\Leftrightarrow \quad & \boldsymbol{\sigma}_{jm} \sum_{p=1}^P \sum_{t=1}^T E_{jm} \, \mathbf{o}_t - \boldsymbol{\sigma}_{jm} \sum_{p=1}^P \sum_{t=1}^T E_{jm} \, \boldsymbol{\mu}_{jm} = 0
\end{aligned}$$

On obtient :

$$\boldsymbol{\mu}_{jm} = \frac{\sum_{p=1}^P \sum_{t=1}^T E_{jm} \, \mathbf{o}_t}{\sum_{p=1}^P \sum_{t=1}^T E_{jm}}$$

- A présent, il nous reste à trouver $\boldsymbol{\sigma}_{jm}$. On dérive $Q(\Lambda, \hat{\Lambda})$ par rapport à $\boldsymbol{\sigma}_{jm}$:

$$\begin{aligned}
&\frac{\partial Q(\Lambda, \hat{\Lambda})}{\partial \boldsymbol{\sigma}_{jm}} \\
&= \sum_{p=1}^{P}\sum_{t=1}^{T} \frac{\hat{g}_{j,t}\, c_{jm}}{b_j(\mathbf{o}_t)} \frac{\partial \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm})}{\partial \boldsymbol{\sigma}_{jm}} \\
&= \sum_{p=1}^{P}\sum_{t=1}^{T} \frac{\hat{g}_{j,t}\, c_{jm}}{b_j(\mathbf{o}_t)} \Big[\frac{\frac{\partial \sqrt{|\boldsymbol{\sigma}_{jm}|}}{\partial \boldsymbol{\sigma}_{jm}}}{\sqrt{(2\pi)^M}} e^{-\frac{1}{2}{}^T(\mathbf{o}_t - \boldsymbol{\mu}_{jm})\boldsymbol{\sigma}_{jm}(\mathbf{o}_t - \boldsymbol{\mu}_{jm})} \\
&+ \frac{\sqrt{|\ \boldsymbol{\sigma}_{jm}\ |}}{\sqrt{(2\pi)^M}} \frac{\partial e^{-\frac{1}{2}{}^T(\mathbf{o}_t - \boldsymbol{\mu}_{jm})\boldsymbol{\sigma}_{jm}(\mathbf{o}_t - \boldsymbol{\mu}_{jm})}}{\partial \boldsymbol{\sigma}_{jm}} \Big]
\end{aligned}$$

Dans cette expression, il est nécessaire de dériver le déterminant de la matrice $\boldsymbol{\sigma}_{jm}$.
Soit A une matrice symétrique définie positive, alors on a :

$$\frac{\partial \sqrt{|\ A\ |}}{\partial A} = \frac{1}{2\sqrt{|\ A\ |}} \frac{\partial\, |\ A\ |}{\partial A}$$

de plus, on sait que :

$$\frac{\partial\, |\ A\ |}{\partial A} = |\ A\ |\ A^{-1}$$

d'où :

$$\frac{\partial \sqrt{|\ A\ |}}{\partial A} = \frac{1}{2}\sqrt{|\ A\ |}\, A^{-1} \tag{C.11}$$

ainsi $\frac{\partial Q(\Lambda, \hat{\Lambda})}{\partial \boldsymbol{\sigma}_{jm}}$ devient :

$$\begin{aligned}
\frac{\partial Q(\Lambda, \hat{\Lambda})}{\partial \boldsymbol{\sigma}_{jm}} &= \sum_{p=1}^{P}\sum_{t=1}^{T} \frac{\hat{g}_{j,t}\, c_{jm}}{b_j(\mathbf{o}_t)} \Big[\frac{1}{2}(\boldsymbol{\sigma}_{jm})^{-1} \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm}) \\
&+ \mathcal{M}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm}) \frac{\partial -\frac{1}{2}{}^T(\mathbf{o}_t - \boldsymbol{\mu}_{jm})\boldsymbol{\sigma}_{jm}(\mathbf{o}_t - \boldsymbol{\mu}_{jm})}{\partial \boldsymbol{\sigma}_{jm}} \Big] \\
&= \frac{1}{2}\sum_{p=1}^{P}\sum_{t=1}^{T} E_{jm} \Big[(\boldsymbol{\sigma}_{jm})^{-1} - (\mathbf{o}_t - \boldsymbol{\mu}_{jm})^T (\mathbf{o}_t - \boldsymbol{\mu}_{jm}) \Big]
\end{aligned}$$

ainsi :

$$\frac{\partial Q(\Lambda, \hat{\Lambda})}{\partial \boldsymbol{\sigma}_{jm}} = 0$$

$$\Leftrightarrow \quad \frac{1}{2}\sum_{p=1}^{P}\sum_{t=1}^{T} E_{jm}\ \left[(\boldsymbol{\sigma}_{jm})^{-1} - (\mathbf{o}_t - \boldsymbol{\mu}_{jm})^{T}(\mathbf{o}_t - \boldsymbol{\mu}_{jm})\right] = 0$$

$$\Leftrightarrow \quad (\boldsymbol{\sigma}_{jm})^{-1}\sum_{p=1}^{P}\sum_{t=1}^{T} E_{jm} - \sum_{p=1}^{P}\sum_{t=1}^{T} E_{jm}(\mathbf{o}_t - \boldsymbol{\mu}_{jm})^{T}(\mathbf{o}_t - \boldsymbol{\mu}_{jm}) = 0$$

On obtient :

$$\boldsymbol{\sigma}_{jm} = \left(\frac{\sum_{p=1}^{P}\sum_{t=1}^{T} E_{jm}\,(\mathbf{o}_t - \boldsymbol{\mu}_{jm})^{T}(\mathbf{o}_t - \boldsymbol{\mu}_{jm})}{\sum_{p=1}^{P}\sum_{t=1}^{T} E_{jm}}\right)^{-1}$$

ANNEXE D

Application de l'algorithme EM aux IOHMM

D.1 Algorithme EM : Etape E

D.1.1 Calcul de $\hat{g}_{i,t}$

$$\begin{aligned}\hat{g}_{i,t} &= Pr(x_t = i \mid \mathbf{y}_1^T, \mathbf{u}_1^T) \\ &= \frac{Pr(x_t = i, \mathbf{y}_1^T \mid \mathbf{u}_1^T)}{Pr(\mathbf{y}_1^T \mid \mathbf{u}_1^T)} \\ &= \frac{Pr(x_t = i, \mathbf{y}_1^T \mid \mathbf{u}_1^T)}{L}\end{aligned}$$

Où L est donné par :

$$\begin{aligned}L &= Pr(\mathbf{y}_1^T \mid \mathbf{u}_1^T) \\ &= \sum_{i=1}^{n} Pr(\mathbf{y}_1^T, x_T = i \mid \mathbf{u}_1^T)\end{aligned}$$

Reprenons le calcul de $\hat{g}_{i,t}$:

$$\begin{aligned}\hat{g}_{i,t} &= \frac{Pr(x_t = i, \mathbf{y}_1^T \mid \mathbf{u}_1^T)}{L} \\ &= \frac{Pr(\mathbf{y}_{t+1}^T \mid x_t = i, \mathbf{y}_1^t, \mathbf{u}_1^T)\, Pr(x_t = i, \mathbf{y}_1^t \mid \mathbf{u}_1^T)}{L} \\ &= \frac{Pr(\mathbf{y}_{t+1}^T \mid x_t = i, \mathbf{u}_1^t)\, Pr(x_t = i, \mathbf{y}_1^t \mid \mathbf{u}_1^t)}{L}\end{aligned}$$

A présent, on pose :

$$\begin{aligned}\hat{\alpha}_{i,t} &= Pr(\mathbf{y}_1^t, x_t = i \mid \mathbf{u}_1^t) \\ \hat{\beta}_{i,t} &= Pr(\mathbf{y}_{t+1}^T \mid x_t = i, \mathbf{u}_t^T)\end{aligned}$$

Ainsi, $\hat{g}_{i,t}$ s'écrit :

$$\hat{g}_{i,t} = \frac{\hat{\beta}_{i,t}\,\hat{\alpha}_{i,t}}{L}$$

Et L devient :

$$L = \sum_{i=1}^{n} \hat{\alpha}_{i,T}$$

D.1.2 Calcul de $\hat{h}_{ij,t}$

$$\begin{aligned}
\hat{h}_{ij,t} &= Pr(x_t = i, x_{t-1} = j \mid \mathbf{u}_1^T, \mathbf{y}_1^T) \\
&= \frac{Pr(x_t = i, x_{t-1} = j, \mathbf{y}_1^T \mid \mathbf{u}_1^T)}{Pr(\mathbf{y}_1^T \mid \mathbf{u}_1^T)} \\
&= \frac{Pr(\mathbf{y}_{t+1}^T \mid x_t = i, x_{t-1} = j, \mathbf{y}_1^t, \mathbf{u}_1^T)\, Pr(x_t = i, x_{t-1} = j, \mathbf{y}_1^t \mid \mathbf{u}_1^T)}{L} \\
&= Pr(\mathbf{y}_{t+1}^T \mid x_t = i, x_{t-1} = j, \mathbf{y}_1^t, \mathbf{u}_1^T)\, Pr(\mathbf{y}_t \mid x_t = i, x_{t-1} = j, \mathbf{y}_1^{t-1}, \mathbf{u}_1^T) \\
&\quad Pr(x_t = i, x_{t-1} = j, \mathbf{y}_1^{t-1} \mid \mathbf{u}_1^T) / L \\
&= Pr(\mathbf{y}_{t+1}^T \mid x_t = i, x_{t-1} = j, \mathbf{y}_1^t, \mathbf{u}_1^T)\, Pr(\mathbf{y}_t \mid x_t = i, x_{t-1} = j, \mathbf{y}_1^{t-1}, \mathbf{u}_1^T) \\
&\quad Pr(\mathbf{y}_1^{t-1}, x_{t-1} = j \mid x_t = i, \mathbf{u}_1^T)\, Pr(x_t = i \mid x_{t-1} = j, \mathbf{y}_1^{t-1}, \mathbf{u}_1^T) / L \\
&= Pr(\mathbf{y}_{t+1}^T \mid x_t = i, \mathbf{u}_t^T)\, Pr(\mathbf{y}_t \mid x_t = i, \mathbf{u}_t) \\
&\quad Pr(\mathbf{y}_1^{t-1}, x_{t-1} = j \mid \mathbf{u}_1^{t-1})\, Pr(x_t = i \mid x_{t-1} = j, \mathbf{u}_t) / L \\
&= \frac{\hat{\beta}_{i,t}\, f_Y(\mathbf{y}_t; \boldsymbol{\eta}_{i,t})\, \hat{\alpha}_{j,t-1}\, \varphi_{ij,t}}{L}
\end{aligned}$$

$$\hat{h}_{ij,t} = \frac{\hat{\alpha}_{j,t-1}\,\varphi_{ij,t}\,\hat{\beta}_{i,t}\, f_Y(\mathbf{y}_t; \boldsymbol{\eta}_{i,t})}{L}$$

D.1.3 Calcul de $\hat{\alpha}_{i,t}$

$$\begin{aligned}
\hat{\alpha}_{i,t} &= Pr(\mathbf{y}_1^t, x_t = i \mid \mathbf{u}_1^t) \\
&= \sum_{j=1}^{n} Pr(\mathbf{y}_1^t, x_t = i, x_{t-1} = j \mid \mathbf{u}_1^t) \\
&= \sum_{j=1}^{n} Pr(\mathbf{y}_t \mid \mathbf{y}_1^{t-1}, x_t = i, x_{t-1} = j, \mathbf{u}_1^t)
\end{aligned}$$

$$
\begin{aligned}
& Pr(x_t = i, x_{t-1} = j, \mathbf{y}_1^{t-1} \mid \mathbf{u}_1^t) \\
= & \sum_{j=1}^{n} Pr(\mathbf{y}_t \mid \mathbf{y}_1^{t-1}, x_t = i, x_{t-1} = j, \mathbf{u}_1^t) \\
& Pr(x_t = i \mid x_{t-1} = j, \mathbf{y}_1^{t-1}, \mathbf{u}_1^t)\, Pr(x_{t-1} = j, \mathbf{y}_1^{t-1} \mid \mathbf{u}_1^t) \\
= & \sum_{j=1}^{n} Pr(\mathbf{y}_t \mid x_t = i, \mathbf{u}_t)\, Pr(x_t = i \mid x_{t-1} = j, \mathbf{y}_1^{t-1}, \mathbf{u}_t)\, Pr(\mathbf{y}_1^{t-1}, x_{t-1} = j \mid \mathbf{u}_1^{t-1}) \\
= & \; Pr(\mathbf{y}_t \mid x_t = i, \mathbf{u}_t) \sum_{j=1}^{n} \varphi_{ij,t}\, \hat{\alpha}_{j,t-1} \\
= & \; f_Y(\mathbf{y}_t; \boldsymbol{\eta}_{i,t}) \sum_{j=1}^{n} \varphi_{ij,t}\, \hat{\alpha}_{j,t-1}
\end{aligned}
$$

$$
\boxed{\hat{\alpha}_{i,t} \;=\; f_Y(\mathbf{y}_t; \boldsymbol{\eta}_{i,t}) \sum_{j=1}^{n} \varphi_{ij,t}\, \hat{\alpha}_{j,t-1}}
$$

D.1.4 Calcul de $\hat{\beta}_{i,t}$

$$
\begin{aligned}
\hat{\beta}_{i,t} &= Pr(\mathbf{y}_{t+1}^T \mid x_t = i, \mathbf{u}_t^T) \\
&= \sum_{j=1}^{n} Pr(\mathbf{y}_{t+1}^T, x_{t+1} = j \mid x_t = i, \mathbf{u}_t^T) \\
&= \sum_{j=1}^{n} Pr(\mathbf{y}_{t+1} \mid \mathbf{y}_{t+2}^T, x_{t+1} = j, x_t = i, \mathbf{u}_t^T) \\
&\quad Pr(\mathbf{y}_{t+2}^T, x_{t+1} = j \mid x_t = i, \mathbf{u}_t^T) \\
&= \sum_{j=1}^{n} Pr(\mathbf{y}_{t+1} \mid \mathbf{y}_{t+2}^T, x_{t+1} = j, x_t = i, \mathbf{u}_t^T) \\
&\quad Pr(\mathbf{y}_{t+2}^T \mid x_{t+1} = j, x_t = i, \mathbf{u}_t^T) \\
&\quad Pr(x_{t+1} = j \mid x_t = i, \mathbf{u}_t^T) \\
&= \sum_{j=1}^{n} Pr(\mathbf{y}_{t+1} \mid x_{t+1} = j, \mathbf{u}_t^T) \\
&\quad Pr(\mathbf{y}_{t+2}^T \mid x_{t+1} = j, \mathbf{u}_{t+1}^T)\, Pr(x_{t+1} = j \mid x_t = i, \mathbf{u}_{t+1}^T) \\
&= \sum_{j=1}^{n} Pr(\mathbf{y}_{t+1} \mid x_{t+1} = j, \mathbf{u}_{t+1})\, \hat{\beta}_{j,t+1}\, \varphi_{ji,t+1} \\
&= \sum_{j=1}^{n} f_Y(\mathbf{y}_{t+1}; \boldsymbol{\eta}_{j,t+1})\, \hat{\beta}_{j,t+1}\, \varphi_{ji,t+1}
\end{aligned}
$$

$$\hat{\beta}_{i,t} \;=\; \sum_{j=1}^{n} f_Y(\mathbf{y}_{t+1};\boldsymbol{\eta}_{j,t+1})\,\hat{\beta}_{j,t+1}\,\varphi_{ji,t+1}$$

D.2 Algorithme EM : Etape M

D.2.1 Calcul des dérivées partielles $\frac{\partial \varphi_{ij,t}}{\partial s_{ij}}$

$$\begin{aligned}\frac{\partial \varphi_{ij,t}}{\partial s_{qj}} &= \frac{\partial \frac{e^{s_{ij}}}{\sum_{l=1}^{n} e^{s_{lj}}}}{\partial s_{qj}} \\ &= \frac{\frac{\partial e^{s_{ij}}}{\partial s_{qj}} \sum_{l=1}^{n} e^{s_{lj}} - e^{s_{ij}} \frac{\partial \sum_{l=1}^{n} e^{s_{lj}}}{\partial s_{qj}}}{(\sum_{l=1}^{n} e^{s_{lj}})^2}\end{aligned}$$

- $q = i$:

$$\frac{\partial \varphi_{ij,t}}{\partial s_{ij}} = \frac{e^{s_{ij}} \sum_{l=1}^{n} e^{s_{lj}} - e^{s_{ij}}\, e^{s_{ij}}}{(\sum_{l=1}^{n} e^{s_{lj}})^2} = \varphi_{ij,t}\,(1 - \varphi_{ij,t})$$

- $q \neq i$:

$$\frac{\partial \varphi_{ij,t}}{\partial s_{qj}} = \frac{-e^{s_{ij}}\, e^{s_{qj}}}{(\sum_{l=1}^{n} e^{s_{lj}})^2} = -\varphi_{ij,t}\,\varphi_{qj,t}$$

D.2.2 Calcul des dérivées partielles de $\frac{\partial Q(\Theta,\hat{\Theta})}{\partial \theta_{kj}}$

$$\begin{aligned}\frac{\partial Q(\Theta,\hat{\Theta})}{\partial \theta_{kj}} &= \sum_{p=1}^{P}\sum_{t=1}^{T}\sum_{i=1}^{n} \frac{\hat{h}_{ij,t}}{\varphi_{ij,t}} \sum_{q=1}^{n} \frac{\partial \varphi_{ij,t}}{\partial s_{qj}} \frac{\partial s_{qj}}{\partial \theta_{kj}} \\ &= \sum_{p=1}^{P}\sum_{t=1}^{T}\sum_{i=1}^{n} \frac{\hat{h}_{ij,t}}{\varphi_{ij,t}} \Big[\varphi_{ij,t}\,(1-\varphi_{ij,t})\,\frac{\partial s_{ij}}{\partial \theta_{kj}} - \sum_{q\neq i}^{n} \varphi_{ij,t}\,\varphi_{qj,t}\,\frac{\partial s_{qj}}{\partial \theta_{kj}}\Big] \\ &= \sum_{p=1}^{P}\sum_{t=1}^{T}\sum_{i=1}^{n} \hat{h}_{ij,t} \Big[(1-\varphi_{ij,t})\,\frac{\partial s_{ij}}{\partial \theta_{kj}} - \sum_{q\neq i}^{n} \varphi_{qj,t}\,\frac{\partial s_{qj}}{\partial \theta_{kj}}\Big] \\ &= \sum_{p=1}^{P}\sum_{t=1}^{T}\sum_{i=1}^{n} \Big[\hat{h}_{ij,t}\,\frac{\partial s_{ij}}{\partial \theta_{kj}} - \hat{h}_{ij,t}\,\varphi_{ij,t}\,\frac{\partial s_{ij}}{\partial \theta_{kj}} - \hat{h}_{ij,t} \sum_{q\neq i}^{n} \varphi_{qj,t}\,\frac{\partial s_{qj}}{\partial \theta_{kj}}\Big] \\ &= \sum_{p=1}^{P}\sum_{t=1}^{T} \Big[\sum_{i=1}^{n} \hat{h}_{ij,t}\,\frac{\partial s_{ij}}{\partial \theta_{kj}}\Big] - \Big[\sum_{i=1}^{n} \hat{h}_{ij,t} \sum_{q=1}^{n} \varphi_{qj,t}\,\frac{\partial s_{qj}}{\partial \theta_{kj}}\Big] \\ &= \sum_{p=1}^{P}\sum_{t=1}^{T} \Big[\Big(\sum_{q=1}^{n} \hat{h}_{qj,t}\,\frac{\partial s_{qj}}{\partial \theta_{kj}}\Big) - \Big(\sum_{q=1}^{n} \varphi_{qj,t}\,\frac{\partial s_{qj}}{\partial \theta_{kj}}\Big)\Big(\sum_{i=1}^{n} \hat{h}_{ij,t}\Big)\Big]\end{aligned}$$

$$\boxed{\frac{\partial Q(\Theta,\hat{\Theta})}{\partial \theta_{kj}} = \sum_{p=1}^{P}\sum_{t=1}^{T}\sum_{q=1}^{n}\frac{\partial s_{qj}}{\partial \theta_{kj}}(\hat{h}_{qj,t} - \varphi_{qj,t}\sum_{i=1}^{n}\hat{h}_{ij,t})}$$

D.2.3 Calcul des multiplicateurs de Lagrange pour résoudre $\frac{\partial Q(\Theta,\hat{\Theta})}{\partial w_{ijk}}$

On pose $f(.)$ la fonction $Q(\Theta,\hat{\Theta})$ et $g(.)$ la fonction de contrainte (4.16), alors :

$$\begin{aligned}
& \nabla f = \lambda_{jk}\nabla g \\
\Leftrightarrow \quad & \sum_{p}^{P}\sum_{t}^{T}\frac{\hat{h}_{ij,t}}{w_{ijk}} = \lambda_{jk}\frac{\partial(\sum_{l=1}^{n} w_{ljk} - 1)}{\partial w_{ijk}} \\
\Leftrightarrow \quad & w_{ijk} = \frac{\sum_{p}^{P}\sum_{t}^{T}\hat{h}_{ij,t}}{\lambda_{jk}} \qquad \text{(D.1)}
\end{aligned}$$

Or, on a $\sum_{l=1}^{n} w_{ljk} = 1$ (Equation 4.16) d'où:

$$\begin{aligned}
& \sum_{l=1}^{n}\frac{\sum_{p}^{P}\sum_{t}^{T}\hat{h}_{lj,t}}{\lambda_{jk}} = 1 \\
\Leftrightarrow \quad & \lambda_{jk} = \sum_{l=1}^{n}\sum_{p}^{P}\sum_{t}^{T}\hat{h}_{lj,t} \qquad \text{(D.2)}
\end{aligned}$$

Ainsi les équations (D.1) et (D.2) nous donnent :

$$\boxed{w_{ijk} = \frac{\sum_{p}^{P}\sum_{t}^{T}\hat{h}_{ij,t}}{\sum_{l=1}^{n}\sum_{p}^{P}\sum_{t}^{T}\hat{h}_{lj,t}}}$$

D.2.4 Calcul des multiplicateurs de Lagrange pour résoudre $\frac{\partial Q(\Theta,\hat{\Theta})}{\partial z_{qik}}$

On pose $f(.)$ la fonction $Q(\Theta,\hat{\Theta})$ et $g(.)$ la fonction de contrainte (4.17), alors :

$$\begin{aligned}
& \nabla f = \lambda_{ik}\nabla g \\
\Leftrightarrow \quad & \sum_{p}^{P}\sum_{t}^{T}\frac{\hat{g}_{i,t}}{z_{qik}}\delta_{y_t q} = \lambda_{ik}\frac{\partial(\sum_{l=1}^{Z} z_{qik} - 1)}{\partial z_{qik}} \\
\Leftrightarrow \quad & z_{qik} = \frac{\sum_{p}^{P}\sum_{t}^{T}\hat{g}_{i,t}\delta_{y_t k}}{\lambda_{ik}} \qquad \text{(D.3)}
\end{aligned}$$

où δ_{ij} est le symbole de Kronecker :

$$\delta_{ij} = \begin{cases} 1 & : \quad i = j \\ 0 & : \quad i \neq j \end{cases}$$

Or, on a $\sum_{l=1}^{Z} z_{lik} = 1$ (Equation 4.17) d'où:

$$\begin{aligned} & \sum_{l=1}^{Z} \frac{\sum_p^P \sum_t^T \hat{g}_{i,t} \delta_{y_t l}}{\lambda_{ik}} = 1 \\ \Leftrightarrow \quad & \lambda_{ik} = \sum_{l=1}^{Z} \sum_{p}^{P} \sum_{t}^{T} \hat{g}_{i,t} \delta_{y_t l} \end{aligned} \tag{D.4}$$

Ainsi les équations (D.3) et (D.4) nous donnent :

$$z_{qik} = \frac{\sum_p^P \sum_t^T \hat{g}_{i,t} \delta_{y_t q}}{\sum_{l=1}^{Z} \sum_p^P \sum_t^T \hat{g}_{i,t} \delta_{y_t l}}$$

ANNEXE E

Application de l'algorithme EM aux GIOHMM

E.1 Algorithme EM : Etape E

E.1.1 Calcul de $\hat{g}_{i,t}$

$$\begin{aligned}\hat{g}_{i,t} &= Pr(x_t = i \mid \mathbf{y}_1^T, \mathbf{v}_1^T, \mathbf{u}_1^T) \\ &= \frac{Pr(x_t = i, \mathbf{y}_1^T, \mathbf{v}_1^T \mid \mathbf{u}_1^T)}{Pr(\mathbf{y}_1^T, \mathbf{v}_1^T \mid \mathbf{u}_1^T)} \\ &= \frac{Pr(x_t = i, \mathbf{y}_1^T, \mathbf{v}_1^T \mid \mathbf{u}_1^T)}{L}\end{aligned}$$

Où L est donné par :

$$\begin{aligned}L &= Pr(\mathbf{y}_1^T, \mathbf{v}_1^T \mid \mathbf{u}_1^T) \\ &= \sum_{i=1}^{n} Pr(\mathbf{y}_1^T, \mathbf{v}_1^T, x_T = i \mid \mathbf{u}_1^T)\end{aligned}$$

Reprenons le calcul de $\hat{g}_{i,t}$:

$$\begin{aligned}\hat{g}_{i,t} &= \frac{Pr(x_t = i, \mathbf{y}_1^T, \mathbf{v}_1^T \mid \mathbf{u}_1^T)}{L} \\ &= \frac{Pr(\mathbf{y}_{t+1}^T, \mathbf{v}_{t+1}^T \mid x_t = i, \mathbf{y}_1^t, \mathbf{v}_1^t, \mathbf{u}_1^T)\, Pr(x_t = i, \mathbf{y}_1^t, \mathbf{v}_1^t \mid \mathbf{u}_1^T)}{L} \\ &= \frac{Pr(\mathbf{y}_{t+1}^T, \mathbf{v}_{t+1}^T \mid x_t = i, \mathbf{u}_t^t)\, Pr(x_t = i, \mathbf{y}_1^t, \mathbf{v}_1^t \mid \mathbf{u}_1^t)}{L}\end{aligned}$$

A présent, on pose :

$$\begin{aligned}\hat{\alpha}_{i,t} &= Pr(\mathbf{y}_1^t, \mathbf{v}_1^t, x_t = i \mid \mathbf{u}_1^t) \\ \hat{\beta}_{i,t} &= Pr(\mathbf{y}_{t+1}^T, \mathbf{v}_{t+1}^T \mid x_t = i, \mathbf{u}_t^T)\end{aligned}$$

Ainsi, $\hat{g}_{i,t}$ s'écrit :

$$\hat{g}_{i,t} = \frac{\hat{\beta}_{i,t}\,\hat{\alpha}_{i,t}}{L}$$

Et L devient :

$$L = \sum_{i=1}^{n} \hat{\alpha}_{i,T}$$

D'autre part, $Pr(\mathbf{y}_t, \mathbf{v}_t \mid x_t = i, \mathbf{u}_t)$ est calculé par :

$$\begin{aligned} Pr(\mathbf{y}_t, \mathbf{v}_t \mid x_t = i, \mathbf{u}_t) &= Pr(\mathbf{y}_t \mid \mathbf{v}_t, x_t = i, \mathbf{u}_t)\, Pr(\mathbf{v}_t \mid x_t = i, \mathbf{u}_t) \\ &= Pr(\mathbf{y}_t \mid x_t = i, \mathbf{u}_t)\, Pr(\mathbf{v}_t \mid x_t = i, \mathbf{u}_t) \end{aligned}$$

On obtient :

$$Pr(\mathbf{y}_t, \mathbf{v}_t \mid x_t = i, \mathbf{u}_t) = f_Y(\mathbf{y}_t; \boldsymbol{\eta}_{i,t})\, d_i(\mathbf{v}_t)$$

E.1.2 Calcul de $\hat{h}_{ij,t}$

$$\begin{aligned} \hat{h}_{ij,t} &= Pr(x_t = i, x_{t-1} = j \mid \mathbf{u}_1^T, \mathbf{v}_1^T, \mathbf{y}_1^T) \\ &= \frac{Pr(x_t = i, x_{t-1} = j, \mathbf{y}_1^T, \mathbf{v}_1^T \mid \mathbf{u}_1^T)}{Pr(\mathbf{y}_1^T, \mathbf{v}_1^T \mid \mathbf{u}_1^T)} \\ &= \frac{Pr(\mathbf{y}_{t+1}^T, \mathbf{v}_{t+1}^T \mid x_t = i, x_{t-1} = j, \mathbf{y}_1^t, \mathbf{v}_1^t, \mathbf{u}_1^T)\, Pr(x_t = i, x_{t-1} = j, \mathbf{y}_1^t, \mathbf{v}_1^t \mid \mathbf{u}_1^T)}{L} \\ &= Pr(\mathbf{y}_{t+1}^T, \mathbf{v}_{t+1}^T \mid x_t = i, \mathbf{u}_t^T)\, Pr(\mathbf{y}_t, \mathbf{v}_t \mid x_t = i, x_{t-1} = j, \mathbf{y}_1^{t-1}, \mathbf{v}_1^{t-1}, \mathbf{u}_1^T) \\ &\quad Pr(x_t = i, x_{t-1} = j, \mathbf{y}_1^{t-1}, \mathbf{v}_1^{t-1} \mid \mathbf{u}_1^T) \,/\, L \\ &= Pr(\mathbf{y}_{t+1}^T, \mathbf{v}_{t+1}^T \mid x_t = i, \mathbf{u}_t^T)\, Pr(\mathbf{y}_t, \mathbf{v}_t \mid x_t = i, \mathbf{u}_t) \\ &\quad Pr(\mathbf{y}_1^{t-1}, \mathbf{v}_1^{t-1}, x_{t-1} = j \mid x_t = i, \mathbf{u}_1^T)\, Pr(x_t = i \mid x_{t-1} = j, \mathbf{y}_1^{t-1}, \mathbf{v}_1^{t-1}, \mathbf{u}_1^T) \,/\, L \\ &= Pr(\mathbf{y}_{t+1}^T, \mathbf{v}_{t+1}^T \mid x_t = i, \mathbf{u}_t^T)\, Pr(\mathbf{y}_t, \mathbf{v}_t \mid x_t = i, \mathbf{u}_t) \\ &\quad Pr(\mathbf{y}_1^{t-1}, \mathbf{v}_1^{t-1}, x_{t-1} = j \mid \mathbf{u}_1^T)\, Pr(x_t = i \mid x_{t-1} = j, \mathbf{u}_t) \,/\, L \\ &= Pr(\mathbf{y}_{t+1}^T, \mathbf{v}_{t+1}^T \mid x_t = i, \mathbf{u}_t^T)\, Pr(\mathbf{y}_t, \mathbf{v}_t \mid x_t = i, \mathbf{u}_t) \\ &\quad Pr(\mathbf{y}_1^{t-1}, \mathbf{v}_1^{t-1}, x_{t-1} = j \mid \mathbf{u}_1^{t-1})\, Pr(x_t = i \mid x_{t-1} = j, \mathbf{u}_t) \,/\, L \\ &= \frac{\hat{\beta}_{i,t}\, d_i(\mathbf{v}_t)\, f_Y(\mathbf{y}_t; \boldsymbol{\eta}_{i,t})\, \hat{\alpha}_{j,t-1}\, \varphi_{ij,t}}{L} \end{aligned}$$

$$\hat{h}_{ij,t} = \frac{\hat{\alpha}_{j,t-1}\,\varphi_{ij,t}\,\hat{\beta}_{i,t}\,f_Y(\mathbf{y}_t;\boldsymbol{\eta}_{i,t}\,d_i(\mathbf{v}_t))}{L}$$

E.1.3 Calcul de $\hat{\alpha}_{i,t}$

$$\begin{aligned}
\hat{\alpha}_{i,t} &= Pr(\mathbf{y}_1^t, \mathbf{v}_1^t, x_t = i \mid \mathbf{u}_1^t) \\
&= \sum_{j=1}^{n} Pr(\mathbf{y}_1^t, \mathbf{v}_1^t, x_t = i, x_{t-1} = j \mid \mathbf{u}_1^t) \\
&= \sum_{j=1}^{n} Pr(\mathbf{y}_t, \mathbf{v}_t \mid \mathbf{y}_1^{t-1}, \mathbf{v}_1^{t-1}, x_t = i, x_{t-1} = j, \mathbf{u}_1^t) \\
&\quad Pr(x_t = i, x_{t-1} = j, \mathbf{y}_1^{t-1}, \mathbf{v}_1^{t-1} \mid \mathbf{u}_1^t) \\
&= \sum_{j=1}^{n} Pr(\mathbf{y}_t, \mathbf{v}_t \mid \mathbf{y}_1^{t-1}, \mathbf{v}_1^{t-1}, x_t = i, x_{t-1} = j, \mathbf{u}_1^t) \\
&\quad Pr(x_t = i \mid x_{t-1} = j, \mathbf{y}_1^{t-1}, \mathbf{v}_1^{t-1}, \mathbf{u}_1^t)\, Pr(x_{t-1} = j, \mathbf{y}_1^{t-1}, \mathbf{v}_1^{t-1} \mid \mathbf{u}_1^t) \\
&= \sum_{j=1}^{n} Pr(\mathbf{y}_t, \mathbf{v}_t \mid x_t = i, \mathbf{u}_t)\, Pr(x_t = i \mid x_{t-1} = j, \mathbf{u}_t)\, Pr(\mathbf{y}_1^{t-1}, \mathbf{v}_1^{t-1}, x_{t-1} = j \mid \mathbf{u}_1^{t-1}) \\
&= Pr(\mathbf{y}_t, \mathbf{v}_t \mid x_t = i, \mathbf{u}_t) \sum_{j=1}^{n} \varphi_{ij,t}\,\hat{\alpha}_{j,t-1} \\
&= f_Y(\mathbf{y}_t;\boldsymbol{\eta}_{i,t})\,d_i(\mathbf{v}_t) \sum_{j=1}^{n} \varphi_{ij,t}\,\hat{\alpha}_{j,t-1}
\end{aligned}$$

$$\hat{\alpha}_{i,t} = f_Y(\mathbf{y}_t;\boldsymbol{\eta}_{i,t})\,d_i(\mathbf{v}_t) \sum_{j=1}^{n} \varphi_{ij,t}\,\hat{\alpha}_{j,t-1}$$

E.1.4 Calcul de $\hat{\beta}_{i,t}$

$$\begin{aligned}
\hat{\beta}_{i,t} &= Pr(\mathbf{y}_{t+1}^T, \mathbf{v}_{t+1}^T \mid x_t = i, \mathbf{u}_t^T) \\
&= \sum_{j=1}^{n} Pr(\mathbf{y}_{t+1}^T, \mathbf{v}_{t+1}^T, x_{t+1} = j \mid x_t = i, \mathbf{u}_t^T) \\
&= \sum_{j=1}^{n} Pr(\mathbf{y}_{t+1}, \mathbf{v}_{t+1} \mid \mathbf{y}_{t+2}^T, \mathbf{v}_{t+2}^T, x_{t+1} = j, x_t = i, \mathbf{u}_t^T) \\
&\quad Pr(\mathbf{y}_{t+2}^T, \mathbf{v}_{t+2}^T, x_{t+1} = j \mid x_t = i, \mathbf{u}_t^T) \\
&= \sum_{j=1}^{n} Pr(\mathbf{y}_{t+1}, \mathbf{v}_{t+1} \mid x_{t+1} = j, \mathbf{u}_{t+1})
\end{aligned}$$

$$
\begin{aligned}
& Pr(\mathbf{y}_{t+2}^T, \mathbf{v}_{t+2}^T \mid x_{t+1} = j, x_t = i, \mathbf{u}_t^T)\, Pr(x_{t+1} = j \mid x_t = i, \mathbf{u}_t^T) \\
= & \sum_{j=1}^{n} Pr(\mathbf{y}_{t+1}, \mathbf{v}_{t+1} \mid x_{t+1} = j, \mathbf{u}_{t+1}) \\
& Pr(\mathbf{y}_{t+2}^T, \mathbf{v}_{t+2}^T \mid x_{t+1} = j, \mathbf{u}_{t+1}^T)\, Pr(x_{t+1} = j \mid x_t = i, \mathbf{u}_{t+1}) \\
= & \sum_{j=1}^{n} Pr(\mathbf{y}_{t+1}, \mathbf{v}_{t+1} \mid x_{t+1} = j, \mathbf{u}_{t+1})\, \hat{\beta}_{j,t+1}\, \varphi_{ji,t+1} \\
= & \sum_{j=1}^{n} f_Y(\mathbf{y}_{t+1}; \boldsymbol{\eta}_{j,t+1})\, d_j(\mathbf{v}_t)\, \hat{\beta}_{j,t+1}\, \varphi_{ji,t+1}
\end{aligned}
$$

$$
\hat{\beta}_{i,t} = \sum_{j=1}^{n} f_Y(\mathbf{y}_{t+1}; \boldsymbol{\eta}_{j,t+1})\, d_j(\mathbf{v}_t)\, \hat{\beta}_{j,t+1}\, \varphi_{ji,t+1}
$$

ANNEXE F Les moments de Zernike

F.1 Introduction

On cherche à obtenir des paramètres décrivant une fonction image f et respectant les propriétés d'un ensemble de fonctions mathématiques. La décomposition en moments répond à ces deux critères. Elle s'effectue selon une base de fonctions orthogonales choisie au préalable. Si les fonctions mathématiques utilisées possèdent des propriétés d'invariances de formes, on conserve ces propriétés par la décomposition. On présentera deux ensembles de fonctions orthogonales possédant de telles propriétés celles de Zernike et celles de Fourier-Mellin. Cette annexe explique les différentes propriétés des moments associées à ces fonctions et leurs mises en œuvres dans un système de reconnaissance de formes.

F.2 Rappel sur les moments

L'intérêt du calcul des moments vient du fait qu'ils décrivent de manière unique l'image décomposée et sont facilement comparables. De plus, dans le cas des fonctions de bases utilisées ici, ils permettent de minimiser les effets des translations, des rotations et des changements d'échelles. Tout d'abord, il faut introduire quelques principes nécessaires à l'étude de ces moments.

Soit D^2 l'espace des fonctions complexes définies sur le disque unité :

$$D^2 = \left\{(x,y) \in \mathcal{R}^2 : x^2 + y^2 \leq 1\right\}$$

et muni d'un produit euclidien propre : $\langle f, g\rangle = \int\int_{D^2} f(x,y)\, g^*(x,y) dxdy$.

Etant donné les fonctions d'une base orthogonale $\{V_r\} \in D^2$ et une image caractérisée par sa fonction f, on peut décomposer cette image par les moments correspondants à la base donnée (Equation F.1). On obtient ainsi une décomposition unique décrivant entièrement la fonction d'origine f.

$$Mom^f_{V_r} = \frac{\langle f, V_r\rangle}{\langle V_r, V_r\rangle} \tag{F.1}$$

La reconstruction de l'image f à partir des moments calculés à l'ordre N est donné par la relation de l'équation F.2.

$$recon^f_{V_r} = \sum_r^N Mom^f_{V_r}.V_r \tag{F.2}$$

F.3 Fonctions définies sur le disque unité

Afin de pouvoir appliquer cette décomposition à une fonction image f, les différentes fonctions employées pour le calcul doivent répondre au critère d'appartenance au disque unité. Il faut tout d'abord définir ce disque unité, son centre, son rayon et la transformation associée. Pour cela, on fait appel aux moments géométriques pour obtenir le centre de gravité (centroïde) de l'image. Le centre va permettre de définir le disque unité, c'est-à-dire la zone d'intérêt incluant la forme à décomposer.

Les moments géométriques d'ordre p et de répétitions q par rapport au point de coordonnées $(\bar{x}, \bar{y})$ de la fonction f sont donnés par la relation suivante :

$$g^f_{pq} = \int\int_{\mathcal{R}^2} (x-\bar{x})^p (y-\bar{y})^q f(x,y) dxdy \tag{F.3}$$

Soit $g^f_{00}, g^f_{01}, g^f_{10}$, les moments géométriques par rapport à l'origine de l'image $(\bar{x}, \bar{y}) = (0,0)$, on définit le centroïde (x_c, y_c) de l'image par :

$$\begin{aligned} x_c &= \frac{g^f_{10}}{g^f_{00}} = \int\int_{\mathcal{R}^2} (x) f(x,y) dxdy \\ y_c &= \frac{g^f_{01}}{g^f_{00}} = \int\int_{\mathcal{R}^2} (y) f(x,y) dxdy \end{aligned}$$

Le disque unité ou zone d'intérêt est défini à partir du centroïde. Il représente le plus petit disque incluant l'objet ou la forme recherchée.

Soit f une fonction image et (x_c, y_c) son centroïde, le rayon d'étendue de la fonction f est :

$$\sigma^f_{(\bar{x},\bar{y})} = sup\left\{\rho : \rho = \sqrt{(x-\bar{x})^2 + (y-\bar{y})^2} \text{ avec } f(x,y) \geq 0\right\} \tag{F.4}$$

A partir de ce rayon d'étendue, on effectue la normalisation suivante sur les coordonnées polaires :

$$\begin{aligned} \rho &= \frac{\sqrt{(x-x_c)^2 + (y-y_c)^2}}{\sigma^f_{(x_c,y_c)}} \\ \theta &= \arctan\left(\frac{(y-y_c)}{(x-x_c)}\right) \end{aligned} \tag{F.5}$$

On obtient ainsi une nouvelle fonction $\hat{f}$: $\forall (x,y) \in \mathcal{R}^2$, $x^2 + y^2 \leq 1$ soit $\rho \leq 1$, ce qui implique que la fonction $\hat{f}$ soit définie sur le disque.

F.4 Invariance de forme

De manière à prouver les différentes invariances pour différentes fonctions orthogonales, on adopte une forme commune pour les différentes fonctions complexes constituant la base:

$$V_r(\rho,\theta) = L_r(\rho)\exp(jr\theta) \text{ avec } \rho \leq 1$$

F.4.1 Invariance en translation

Soit la translation T : $(x',y') \longmapsto (x+a, y+b)$, et f'(x',y') le résultat de la transformation T sur la fonction f, on obtient :

$$\begin{aligned}\sqrt{(x'-x'_c)^2+(y'-y'_c)^2} &= \sqrt{(x+a-x_c+a)^2+(y+b-y_c+b)^2} \\ &= \sqrt{(x-x_c)^2+(y-y_c)^2}\end{aligned}$$

Ainsi, l'équation F.4 nous donne :

$$\sigma^{f'}_{(x'_c,y'_c)} = \sigma^{f}_{(x_c,y_c)}$$

Soit :

$$\rho' = \rho$$

Et :

$$\begin{aligned}\theta' &= \arctan\left(\frac{(y'-y'_c)}{(x'-x'_c)}\right) \\ &= \arctan\left(\frac{(y+b-y_c-b)}{(x+a-x_c-a)}\right) \\ &= \arctan\left(\frac{(y-y_c)}{(x-x_c)}\right) \\ &= \theta\end{aligned}$$

Finalement, on obtient :

$$V_r^{'}(\rho,\theta) = L_r(\rho')\exp(jr\theta') = L_r(\rho)\exp(jr\theta) = V_r(\rho,\theta)$$

Puis, l'équation F.1 nous donne :

$$\begin{aligned}\langle f', V_r^{'}\rangle &= \int\int_{D^2} f'(x',y')V_r^{'*}(x',y')dx'dy' \\ &\quad \text{(Le jacobien de la transformation T est 1)} \\ &= \int\int_{D^2} f(x,y)V_r^{*}(x,y)dxdy \\ &= \langle f, V_r\rangle\end{aligned}$$

De plus :

$$\langle V_r^{'}, V_r^{'} \rangle = \langle V_r, V_r \rangle \tag{F.6}$$

La relation d'invariance se traduit par :

$$Mom_{V_r^{'}}^{f'} = Mom_{V_r}^{f} \tag{F.7}$$

On a donc bien une invariance concernant la translation sur la valeur des moments.

F.4.2 Invariance en changement d'échelle

Un problème se pose concernant la définition des fonctions sur le disque unité. Si l'on effectue un changement d'échelle selon la transformation $S : (x', y') \longmapsto (\alpha x, \alpha y)$ de la fonction f définie sur D^2, alors la fonction f' résultante n'est plus définie sur le disque unité. Un des moyen de pallier ce problème est d'ajuster le disque unité à l'image transformée grâce à la transformation F.5 (le rayon d'étendue croît en même temps que l'échelle). Dans ce cas, on perd toutes les informations concernant l'échelle. On essaie pourtant de conserver ce facteur de manière à ce qu'il apparaisse dans la formulation des moments, cette information s'avère fortement utile dans le système, en marge du traitement des moments. Pour cela, on utilise les moments géométriques pour calculer un coefficient traduisant fidèlement le changement d'échelle.

Calcul du facteur d'échelle

Afin de déterminer un facteur d'échelle cohérent, on utilise le moment géométrique d'ordre zéro qui traduit la "masse" de l'image. On peut donc considérer que par la transformation S, le moment géométrique d'ordre zéro traduira fidèlement la variation de "masse" et donc le changement d'échelle.

$$g_{00}^{f} = \int\int_{\mathcal{R}^2} f(x, y) dx dy$$

Pour la fonction f' résultant de la transformation S, on a :

$$\begin{aligned} g_{pq}^{f'} &= \int\int_{\mathcal{R}^2} (x' - \bar{x}')^p (y' - \bar{y}')^q f'(x', y') dx' dy' \\ &= \int\int_{\mathcal{R}^2} (\alpha x - \alpha \bar{x})^p (\alpha y - \alpha \bar{y})^q f(x, y) \alpha^2 dx dy \\ &= \alpha^{p+q+2} g_{pq}^{f} \end{aligned}$$

On définit ainsi le facteur d'échelle α par:

$$\alpha = \sqrt{\frac{g_{00}^{f'}}{g_{00}^{f}}} \tag{F.8}$$

Calcul des moments à partir du facteur d'échelle

Pour conserver l'information d'échelle, on utilise la relation suivante. La fonction $V_r^{'}$ est définie sur l'intervalle $\alpha(D^2) \longmapsto \mathcal{C}$ ainsi que la fonction image f'. On utilise par contre pour calculer les moments la relation $\langle V_r, V_r \rangle$ définie par rapport à l'image d'origine f définie sur D^2.

Le calcul des moments s'obtient par :

$$Mom_{V_{r'}}^{f'} = \frac{\langle f', V_r^{'} \rangle}{\langle V_r, V_r \rangle}$$

On conserve ainsi toutes les informations relatives au changement d'échelle. On considère maintenant la fonction $f'(x', y') \longmapsto f(\alpha x, \alpha y)$, on montre facilement :

$$\begin{aligned}
\sigma_{(x_c', y_c')}^{f'} &= \sup \left\{ \rho' : \rho' = \sqrt{(x' - x_c')^2 + (y' - y_c')^2} \text{ avec } f'(x', y') \geq 0 \right\} \\
&= \sup \left\{ \rho' : \rho' = \sqrt{\alpha^2 (x - x_c)^2 + \alpha^2 (y - y_c)^2} \text{ avec } f(x, y) \geq 0 \right\} \\
&= \sup \{ \rho' : \rho' = \alpha(\rho) \text{ avec } f(x, y) \geq 0 \} \\
&= \alpha(\sigma_{(x_c, y_c)}^{f})
\end{aligned}$$

Comme pour la translation, on a:

$$\begin{aligned}
\rho' &= \frac{\sqrt{(x' - x_c')^2 + (y' - y_c')^2}}{\sigma_{(x_c', y_c')}^{f'}} \\
&= \frac{\alpha \sqrt{(x - x_c)^2 + (y - y_c)^2}}{\alpha \sigma_{(x_c, y_c)}^{f}} \\
&= \rho
\end{aligned}$$

Et :

$$\begin{aligned}
\theta' &= \arctan \left(\frac{(y' - y_c')}{(x' - x_c')} \right) \\
&= \arctan \left(\frac{\alpha (y - y_c)}{\alpha (x - x_c)} \right) \\
&= \theta
\end{aligned}$$

Finalement :

$$V_r^{'}(\rho, \theta) = L_r(\rho') \exp(jr\theta') = L_r(\rho) \exp(jr\theta) = V_r(\rho, \theta)$$

D'où :

$$\begin{aligned}\langle f', V_r' \rangle &= \int\int_{D^2} f'(x',y') V_r'^*(x',y') dx'dy' \\ &\quad \text{(Le jacobien de la transformation S est } \alpha^2\text{)} \\ &= \int\int_{D^2} f(x,y) V_r^*(x,y) \alpha^2 dxdy \\ &= \alpha^2 \langle f, V_r \rangle\end{aligned}$$

La relation d'invariance se traduit par :

$$Mom_{V_r'}^{f'} = \alpha^2 Mom_{V_r}^{f} \tag{F.9}$$

On démontre ainsi qu'il existe une relation indépendante du couple (p,q) entre les moments de l'image f et ceux de l'image f' ayant subi la transformation S selon le facteur d'échelle défini par la relation F.8.

F.4.3 Invariance en rotation

L'intérêt principal dans l'étude de ces moments est leur comportement lors d'une rotation de l'image.

Si l'on effectue une rotation selon la transformation : $R_\varphi^{(\bar{x},\bar{y})} : (x',y') \longmapsto (x,y)$ telle que :

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = R_\varphi^{(\bar{x},\bar{y})} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix} \begin{pmatrix} (x-\bar{x}) \\ (y-\bar{y}) \end{pmatrix} + \begin{pmatrix} \bar{x} \\ \bar{y} \end{pmatrix}$$

Alors :

$$\begin{aligned}& (x'-x_c')^2 + (y'-y_c')^2 \\ =\ & (\cos\varphi(x-\bar{x}) - \sin\varphi(y-\bar{y}) + \bar{x} - \cos\varphi(x_c-\bar{x}) + \sin\varphi(y_c-\bar{y}) - \bar{x})^2 + \\ & (\sin\varphi(x-\bar{x}) + \cos\varphi(y-\bar{y}) + \bar{y} - \sin\varphi(x_c-\bar{x}) - \cos\varphi(y_c-\bar{y}) - \bar{y})^2 \\ =\ & (\cos\varphi(x-x_c) - \sin\varphi(y-y_c))^2 + (\sin\varphi(x-x_c) + \cos\varphi(y-y_c))^2 \\ =\ & \cos^2\varphi(x-x_c)^2 - 2\cos\varphi(x-x_c)\sin\varphi(y-y_c) + \sin^2\varphi(y-y_c)^2 + \\ & \sin^2\varphi(x-x_c)^2 + 2\cos\varphi(x-x_c)\sin\varphi(y-y_c) + \cos^2\varphi(y-y_c)^2 \\ =\ & (x-x_c)^2 + (y-y_c)^2\end{aligned}$$

En prenant la racine carrée de chaque terme, on obtient :

$$\sigma_{(x_c',y_c')}^{f'} = \sigma_{(x_c,y_c)}^{f}$$

Soit :

$$\rho' = \frac{\sqrt{(x'-x_c')^2 + (y'-y_c')^2}}{\sigma_{(x_c',y_c')}^{f'}}$$

$$\begin{aligned} &= \frac{\alpha\sqrt{(x-x_c)^2+(y-y_c)^2}}{\alpha\sigma^f_{(x_c,y_c)}} \\ &= \rho \end{aligned}$$

Pour l'angle θ, le calcul est légèrement différent:

$$\begin{aligned} (y'-y'_c) &= \sin\varphi(x-x_c)+\cos\varphi(y-y_c) \\ (x'-x'_c) &= \cos\varphi(x-x_c)-\sin\varphi(y-y_c) \end{aligned}$$

$$\begin{aligned} \theta' &= \arctan\left(\frac{(y'-y'_c)}{(x'-x'_c)}\right) \\ &= \arctan\left(\frac{\frac{\sin\varphi(x-x_c)}{\cos\varphi(x-x_c)}+\frac{\cos\varphi(y-y_c)}{\cos\varphi(x-x_c)}}{\frac{\cos\varphi(x-x_c)}{\cos\varphi(x-x_c)}-\frac{\sin\varphi(y-y_c)}{\cos\varphi(x-x_c)}}\right) \\ &\quad \text{avec } \tfrac{(y-y_c)}{(x-x_c)} = \tan\theta \\ &= \arctan\left(\frac{\tan\varphi+\tan\theta}{1-\tan\varphi\tan\theta}\right) \\ &= \arctan(\tan(\theta+\varphi)) \\ &= \theta+\varphi \end{aligned}$$

On retrouve donc la rotation d'angle φ. On remarque aussi que le centre de la rotation $(\bar{x},\bar{y})$ n'intervient pas dans les coordonnées polaires pour le calcul des moments.

Finalement, on obtient :

$$\begin{aligned} V'_r(\rho',\theta') &= L_r(\rho')\exp(jr\theta') = L_r(\rho)\exp(jr(\theta+\varphi)) \\ V'_r(\rho',\theta') &= \exp(jr\varphi)V_r(\rho,\theta) \end{aligned}$$

$$\begin{aligned} \langle f', V'_r\rangle &= \int\int_{\alpha(D^2)} f'(x',y')V'^*_r(x',y')dx'dy' \\ &\quad \text{(Le jacobien de la transformation R est 1)} \\ &= \int\int_{D^2} f(x,y)V^*_r(x,y)\exp(-jr\varphi)dxdy \\ &= \exp(-jr\varphi)\langle f, V_r\rangle \end{aligned}$$

On a aussi :

$$\langle V'_r, V'_r\rangle = \langle V_r, V_r\rangle$$

La relation d'invariance se traduit par :

$$Mom^{f'}_{V'_r} = \exp(-jr\varphi) Mom^{f}_{V_r} \tag{F.10}$$

On obtient donc, comme pour le changement d'échelle, un facteur multiplicatif indépendant du couple (p,q). Cependant, on peut supprimer ce terme et obtenir une invariance en rotation en prenant le module des moments :

$$\mid Mom^{f'}_{V'_r} \mid = \mid Mom^{f}_{V_r} \mid \tag{F.11}$$

F.5 Calcul à partir d'images numériques

Afin de pouvoir calculer les moments dans le cas d'images numériques, il est nécessaire d'appliquer les résultats suivants dans le cas particulier de données discrètes.

F.5.1 Calcul du centroïde

Il s'agit d'appliquer à l'image le calcul des moments géométriques d'ordre p et de répétition q (voir équation F.3). On utilise la relation de calcul des moments suivante.

$$\begin{aligned} g^f_{pq} = & \frac{\Delta_x^{p+1}}{p+1}\frac{\Delta_y^{q+1}}{q+1}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n). \\ & \left\{(m-\bar{\mu})^{p+1} - (m-1-\bar{\mu})^{p+1}\right\}.\left\{(n-\bar{\nu})^{q+1} - (n-1-\bar{\nu})^{q+1}\right\} \end{aligned}$$

où Δ_x, Δ_y sont les valeurs représentant les résolutions des différents axes introduites par la discrétisation des données et où μ et ν sont les coordonnées discrètes définies par : $\mu = \frac{x}{\Delta_x}$ et $\nu = \frac{y}{\Delta_y}$.

Le calcul des moments géométriques pour $(\bar{x}, \bar{y}) = (0,0)$ donne :

$$\begin{aligned} g^f_{00} &= \Delta_x\Delta_y \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n) \\ g^f_{10} &= \frac{\Delta_x^2}{2}\Delta_y \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)(2m-1) \\ g^f_{01} &= \Delta_x\frac{\Delta_y^2}{2} \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)(2n-1) \end{aligned}$$

On peut ainsi calculer le centroïde en effectuant le rapport suivant :

$$x_c = \frac{g^f_{10}}{g^f_{00}} \qquad y_c = \frac{g^f_{01}}{g^f_{00}}$$

F.5.2 Calcul du rayon d'étendue

Soit $C(x_m, y_n)$ une cellule de résolution (ou pixel) de l'image où (x_m, y_n) définissent le centre du pixel et (x_c, y_c) son centroïde, on a :

$$\sigma^f_{(x_c,y_c)} = max\left\{r^{C(x_m,y_n)} : m = 1...M, n = 1...N, f(x_m, y_n) \geq 0\right\}$$

avec : $r^{C(x_m,y_n)} = \sqrt{(x_m - x_c)^2 + (y_n - y_c)^2}$

On obtient ainsi le rayon minimum incluant tous les pixels à valeur non nulle. A partir du centroïde et de ce rayon d'étendue, on définit dans l'image la zone correspondant au disque unité, autrement dit, le domaine de définition des moments.

F.5.3 Calcul des moments

Le moments A_r est définit par :

$$A_r = \langle f, V_r \rangle \quad = \quad \int\int_{D^2} f(x,y) V_r^*(x,y) dxdy \tag{F.12}$$

Soit (x_m, y_n) les coordonnées discrètes de l'image de taille M*N pixels avec $0 \leq m \leq M$ et $0 \leq n \leq N$, la version discrète des moments est donnée par :

$$\hat{A}_r \quad \approx \quad \sum \sum_{x_m{}^2 + y_n{}^2 \leq 1} f(x_m, y_n) h_r(x_m, y_n) \tag{F.13}$$

Avec :

$$h_r(x_m, y_n) = \int_{x_m - \frac{\Delta x_m}{2}}^{x_m + \frac{\Delta x_m}{2}} \int_{y_n - \frac{\Delta y_n}{2}}^{y_n - \frac{\Delta y_n}{2}} V_r^*(x,y) dxdy \tag{F.14}$$

où $h_r(x_m, y_n)$ représente l'intégration de $V_r^*(x,y)$ sur le pixel (m,n).

Un des problèmes principaux dans le calcul des moments est l'évaluation de cette intégrale. N'ayant pas trouvé de forme simple pour le calcul de l'intégrale, on utilisera une approximation correspondant à l'évaluation d'une région circulaire selon le modèle de Gauss [Abramowitz et Stegun, 1970]:

L'approximation de Gauss consiste en une somme de points pondérés:

$$\int\int f(x,y) dxdy = \Delta_{x_m} \Delta_{y_m} \sum_{\substack{s,t=1 \\ s+t=l}}^{l=L} w_l . f(\mu_{ms}, \nu_{nt}) + \varepsilon$$

Ici on prend: $\forall l$, $w_l = \frac{1}{4}$ et $1 \leq (s,t) \leq 2$, soit L=4. On dispose de 4 points d'évaluation (μ_{ms}, ν_{nt}) définis par (Figure F.1) :

$$\mu_{ms} \quad = \quad \Delta_{x_m} \left(x_m + \frac{1}{2} + \frac{1}{2\sqrt{3}} (-1)^s \right)$$

$$\nu_{nt} \quad = \quad \Delta_{y_n} \left(y_n + \frac{1}{2} + \frac{1}{2\sqrt{3}} (-1)^t \right)$$

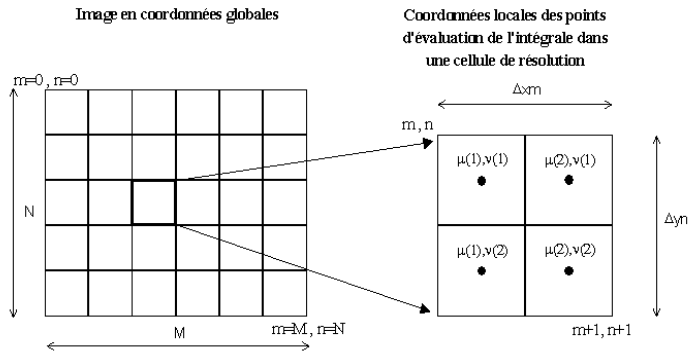


Figure F.1: Evaluation des points d'approximation de l'intégrale

L'équation F.14 devient :

$$h_r(x_m, y_n) = \left\{ \Delta_{x_m} \Delta_{y_n} \sum_{s,t=1}^{2} \frac{1}{4} V_r^*(\mu_{ms}, \nu_{nt}) \right\} + \varepsilon \tag{F.15}$$

Cette approximation introduit une erreur mathématique ε. La suite de cet annexe détaille ces différentes erreurs dues aux approximations et aux hypothèses de calcul.

Finalement en combinant l'équation F.13 et l'équation F.15, on obtient :

$$\hat{A}_r \approx \sum \sum_{{x_m}^2+{y_n}^2 \leq 1} f(x_m, y_n) \left\{ \Delta_{x_m} \Delta_{y_n} \sum_{s,t=1}^{2} \frac{1}{4} V_r^*(\mu_{ms}, \nu_{nt}) \right\} \tag{F.16}$$

De plus, on utilise la forme polaire de V_r^* (équation F.5) qui intègre les éléments relatifs à la définition sur le disque unité. On obtient ainsi une forme implémentable pour le calcul des moments.

F.5.4 Analyse des différentes erreurs

Reprenons l'expression F.13, avec $0 \leq (m,n) \leq k$:

$$\begin{aligned} \hat{A}_r &= \sum \sum_{{x_m}^2+{y_n}^2 \leq 1} f(x_m, y_n) h_r(x_m, y_n) \\ &= \sum \sum_{x_m^2+y_n^2 \leq 1} f(m,n) \int_{x_m - \frac{\Delta x_m}{2}}^{x_m + \frac{\Delta x_m}{2}} \int_{y_n - \frac{\Delta y_n}{2}}^{y_n + \frac{\Delta y_n}{2}} V_{pq}^*(x,y) dx dy \end{aligned}$$

La relation précédente est valable uniquement pour les pixels dont le centre est contenu dans le disque unité. On remarque que même si $f(x,y) = 1$, $\forall (x,y) \in D^2$ on a (Equation F.12):

$$\hat{A}_r \neq A_r$$

On montre ici une des différences principales entre les moments géométriques et les moments orthogonaux définis sur le disque unité. On retrouve donc une erreur inhérente au type de support utilisé (le disque unité) que l'on nommera erreur géométrique E_{pq}^{g}. De même, une erreur de type numérique est introduite due à l'évaluation de l'intégrale [Liao et Pawlak, 1998].

L'erreur globale est donc:

$$E_r = \hat{A}_r - A_r = E_r^g + E_r^n$$

Erreur géométrique

L'expression suivante montre bien la présence de cette erreur géométrique. A partir des relations F.12 et F.16, en posant : $f(x,y) = 1 \ \forall (x,y) \in D^2$ et sachant que $V_{00}^*(x,y) = 1$, on a:

$$\int\int_{D^2} V_{00}^*(x,y)dxdy = \pi$$

On détermine le terme G(k) par:

$$G(k) = \hat{A}_{00} - A_{00} = \frac{4}{k^2}\left[\sum \sum_{(x_m,y_n)\in D^2} 1\right] - \pi$$

On suppose que la région est composée de k^2 pixels: on a donc : $\Delta x_m = \Delta y_n = \frac{2}{k}$. Le terme G(k) est non nul car, comme le montre la figure F.2, certains pixels ne sont pas entièrement inclus dans le disque unité, tandis que d'autres ne sont pas pris en compte. Le terme $\left(\sum \sum_{(x_m,y_n)\in D^2} 1\right)$ représente le nombre de pixels inclus dans le disque unité.

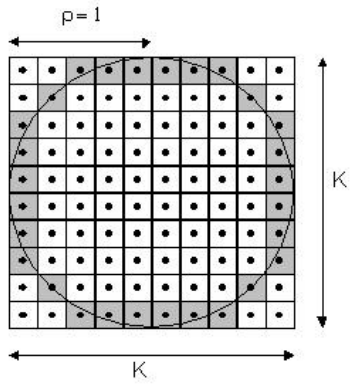


Figure F.2: Approximation d'une région circulaire discrète

Le problème reste dans la quantification du terme G(k). Plusieurs résultats ont été formulés à ce sujet dont notamment [Liao et Pawlak, 1998] : $G(n) = O(k^{-2(1-\theta)})$. Gauss (1834) propose $\theta = 0.5$ et Sierpinski (1906) propose $\theta = \frac{1}{3}$. Mais les plus récents résultats sont dus à Iwaniec et Mozzochi [Iwaniec et Mozzochi, 1988] qui

proposent d'utiliser $\theta = \frac{7}{22}$ ce qui donne $G(k) = O(k^{-\frac{15}{11}})$.

On peut aussi mettre en évidence cette erreur en décomposant l'expression des moments. Soit p_{mn} le pixel correspondant au coordonnées (x_m, y_n) et O(k) la région d'intersection du disque unité responsable de l'erreur géométrique (seuls les pixels inclus dans le disque unité sont pris en compte pour le calcul des moments).

$$\begin{aligned} A_r &= \sum \sum_{(x_m,yn)\in D^2} \int\int_{p_{mn}} f(x,y)V_r^*(x,y)dxdy \\ &- \int\int_{O(n)} f(x,y)V_r^*(x,y)dxdy \qquad \text{(F.17)} \\ E_r &= \hat{A}_r - A_r \\ &= \sum \sum_{(x_m,yn)\in D^2} \int\int_{p_{nm}} (f(x_m,y_n) \\ &- f(x,y))V_r^*(x,y)dxdy - \int\int_{O(n)} f(x,y)V_r^*(x,y)dxdy \end{aligned}$$

Le premier terme défini l'erreur numérique:

$$E_r^n = \sum \sum_{(x_m,yn)\in D^2} \int\int_{p_{ij}} (f(x_i,y_j) - f(x,y))V_r^*(x,y)dxdy$$

et le second terme défini l'erreur géométrique:

$$E_r^g = -\int\int_{O(n)} f(x,y)V_r^*(x,y)dxdy$$

En appliquant l'inégalité de Cauchy-Schwartz au premier terme de l'expression F.17, on a :

$$\begin{aligned} \mid E_r^k \mid &\leq \sum \sum_{(x_m,yn)\in D^2} \left\{ \int\int_{p_{mn}\cap D^2} (f(x_m,y_n - f(x,y))^2 \int\int_{p_{mn}} \mid V_r^*(x,y) \mid^2 dxdy \right\}^{\frac{1}{2}} \\ &\leq \left\{ \sum \sum_{(x_m,yn)\in D^2} \int\int_{p_{mn}\cap D^2} (f(x_m,y_n - f(x,y))^2 \int\int_{p_{mn}} \mid V_r^*(x,y) \mid^2 dxdy \right\}^{\frac{1}{2}} \end{aligned}$$

Ainsi, selon les résultats démontrés dans [Liao et Pawlak, 1996]:

$$\sum \sum_{(x_m,yn)\in D^2} \int\int_{p_{mn}\cap D^2} (f(x_m,y_n - f(x,y)))^2 \leq \frac{4f_{max}}{k^2} V(f)$$

où $f_{max} = \sup_{x,y}(f(x,y))$ et V(f) représente la variance de la fonction f.

Posons à présent :

$$\int\int_{D^2} \mid V_r^*(x,y) \mid^2 dxdy = \beta(j) \qquad \text{(F.18)}$$

où $\beta(j)$ est un terme dépendant de la valeur de j uniquement. La valeur de $\beta(j)$ sera calculée plus loin dans l'étude des différents moments.

On a donc finalement :

$$\mid E_r^n \mid \leq \left\{ \frac{4\beta(j) f_{max}}{k^2} V(f) \right\}^{\frac{1}{2}}$$

En reprenant le deuxième membre de l'équation F.17, on obtient:

$$\mid E_r^g \mid \leq \left\{ \int\int_{O(n)} (f(x,y))^2 \int\int_{D^2} \mid V_r^*(x,y) \mid^2 dxdy \right\}^{\frac{1}{2}} \leq f_{max} \{\beta(j) \mid O(k) \mid\}^{\frac{1}{2}}$$

Selon les résultats précédents, on a $G(k) = O(k^{-\frac{15}{11}})$. Donc :

$$\mid E_r^g \mid \leq f_{max} \beta(j)^{\frac{1}{2}} O^{\frac{1}{2}}(n) \leq f_{max} \beta(j)^{\frac{1}{2}} G^{\frac{1}{2}}(k)$$

L'erreur totale E_r obtenue est donnée par:

$$E_r = E_r^g + E_r^n = f_{max} \beta(j)^{\frac{1}{2}} G^{\frac{1}{2}}(k) + \{4 f_{max} \beta(j) V(f)\}^{\frac{1}{2}} k^{-1}$$

Selon les recherches menées par [Liao et Pawlak, 1998], l'erreur totale est dominée par l'erreur géométrique et peut finalement être approximée par :

$$E_r = \mid \hat{A}_r - A_r \mid \leq O(k^{-\frac{15}{22}})$$

Les éléments de calcul étant définis, il reste maintenant à introduire les différentes fonctions V_r qui forment les bases orthogonales et donc les moments recherchés.

F.6 Les moments

Dans cette section, nous allons décrire les moments les plus connus et les plus utilisés.

F.6.1 Les moments géométriques

[Ming-Kuel, 1962] a été le premier à utiliser les moments géométriques pour la reconnaissance des formes, et a introduit les invariants de Hu. Ces derniers forment un ensemble de 7 descripteurs invariants par translation, par homothétie et par rotation. Cependant, ces moments se sont rapidement montrés limités pour des problèmes de reconnaissance des formes.

F.6.2 Les moments de Zernike

Les fonctions de Zernike forment une base orthogonale présentant des caractéristiques d'invariance en forme. La partie suivante explique en détail la construction de cette base ainsi que ses propriétés.

$$V_{pq}(x,y) = V_{pq}(\rho,\theta) = R_{pq}(\rho) \exp(jq\theta)$$

- p: ordre p $\in \mathcal{Z}^*$

- q: répétition q $\in \mathcal{N}$, avec p-$\mid q \mid$ pair et p-$\mid q \mid \leq n$

- ρ, θ est la forme en coordonnées polaires de (x,y)$\in \mathcal{R}^2$

- $R_{pq}(\rho)$ est le polynôme radial défini par :

$$R_{pq}(\rho) = \sum_{s=0}^{(p-|q|)/2} \frac{(-1)^s (q-s)!}{s! \left(\frac{p+|q|}{2} - s\right)! \left(\frac{p-|q|}{2} - s\right)!} \rho^{p-2s}$$

On remarque que $R_{p,q}(\rho) = R_{p,-q}(\rho)$. De plus si $\rho \in [0,1]$, alors $R_{p,q}(\rho) \leq 1$. La figure F.3 donne une idée de la forme de ce polynôme à l'ordre 10.

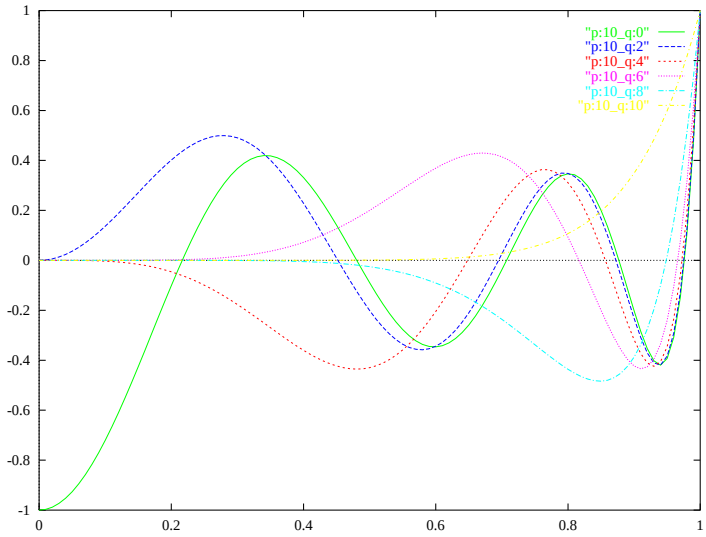


Figure F.3: Polynôme radial $R_{pq}(\rho)$ à l'ordre 10 pour toutes ses répétitions q

Les moments de Zernike sont le résultat de la projection de la fonction image f dans cette base. Ces moments d'ordre p et de répétition q pour une fonction image continue sont définis par la relation F.1, ainsi :

$$Z_{pq}^f = \frac{\langle f, V_{pq} \rangle}{\langle V_{pq}, V_{pq} \rangle} = \frac{\int\int_{D^2} f(x,y)[V_{pq}(x,y)]^* dxdy}{\langle V_{pq}, V_{pq} \rangle}$$

On retrouve la relation $\beta(p)$ utilisée dans le calcul des erreurs (Equation F.18), ainsi :

$$\begin{aligned} \langle V_{pq}, V_{p'q'} \rangle &= \int\int_{(x^2+y^2)\leq 1} [V_{pq}(x,y)]^* V_{p'q'}(x,y) dxdy = \frac{\pi}{p+1} \delta_{pp'} \delta_{qq'} \\ \beta(p) &= \langle V_{pq}, V_{pq} \rangle = \frac{\pi}{p+1} \end{aligned}$$

avec $\delta_{ab} = \begin{cases} 1 & : \quad a = b \\ 0 & : \quad a \neq b \end{cases}$, on obtient donc :

$$Z_{pq}^{f} = \frac{p+1}{\pi} \int\int_{(x^2+y^2)\leq 1} f(x,y) V_{pq}^{*}(x,y) dxdy$$

Après discrétisation, on a :

$$Z_{pq}^{f} = \frac{p+1}{\pi} (\Delta_x \Delta_y \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n)) \int\int_{\substack{(m,m+1)*(n,n+1) \\ x_m^2 + y_n^2 \leq 1}} V_{pq}^{*}(x,y) dxdy$$

On effectue la transformation en coordonnées polaires par l'équation F.5 :

$$V_{pq}^{*}(x,y) = R_{pq}(\rho) \exp(-jq\theta)$$

Puis, on utilise le résultat de l'équation F.16 :

$$\begin{aligned} \int\int_{(m,m+1)*(n,n+1)} V_{pq}^{*}(x,y) dxdy &\approx \left\{ \frac{1}{4} \sum_{(s,t)=1}^{2} V_{pq}^{*}(\mu_s, \nu_s) \right\} \\ &\approx \left\{ \frac{1}{4} \sum_{(s,t)=1}^{2} R_{pq}(\rho_{st}) \exp(-jq\theta_{st}) \right\} \end{aligned}$$

Finalement, l'expression du moment d'ordre p et de répétition q s'écrit :

$$Z_{pq}^{f} = \frac{p+1}{4\pi} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) h_{pq}(x_m, y_n) \qquad \text{(F.19)}$$

$$\text{où} h_{pq}(x_m, y_n) = \left\{ (\Delta_{x_m} \Delta_{y_n}) \sum_{(s,t)=1}^{2} R_{pq}(\rho_{st}) \exp(-jq\theta_{st}) \right\}$$

Le calcul des moments est effectué selon le résultat précédent à partir de l'image représentée par sa fonction f et d'une image représentant les différents poids associés à chacun pixel $h_{pq}(x_m, y_n)$.

F.6.3 Les moments de Fourier-Mellin

Les moments de Fourier-Mellin de l'image, une autre catégorie de moments possédant les mêmes propriétés que ceux de Zernike, ont été introduits par [Arsenaud et Sheng, 1986]. Ces moments sont issus du polynôme de Fourier-Mellin.

$$\begin{aligned} U_{pq}(x,y) &= Q_p(\rho) \exp(jq\theta) \\ Q_p(\rho) &= \sum_{s=0}^{p} \frac{(-1)^s (2p+1-s)!}{s!(p-s)!(p+1-s))!} \rho^{p-2s} \end{aligned}$$

où p est l'ordre, p $\in \mathcal{Z}$ et q est la répétition, q $\in \mathcal{N}$

La figure F.4 donne la forme du polynôme de Fourier-Mellin à l'ordre 10 à titre de comparaison avec celui de Zernike.

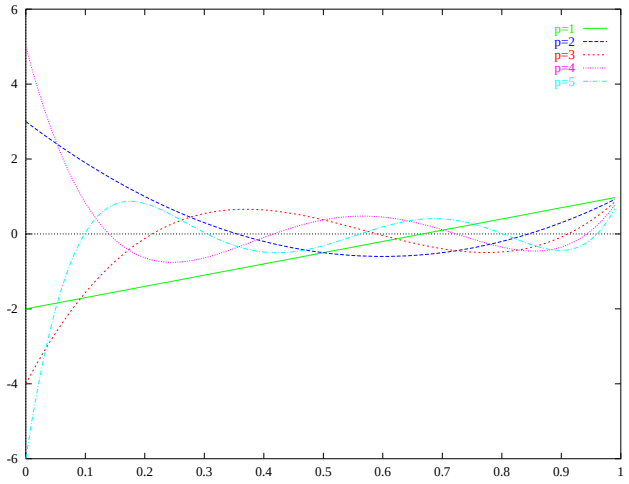


Figure F.4: Polynôme radial $Q_p(\rho)$ à l'ordre 1, 2, 3, 4 et 5

Le comparatif d'efficacité entre ces deux bases de fonctions n'a pas été effectué. Néanmoins, on se basera sur plusieurs études [Belkasim et al., 1991] [Teh et Chin, 1988] dont les résultats montrent que les moments de Zernike ont des performances de reconnaissance et de classification de formes supérieures aux moments géométriques et aux moments complexes. De plus, ils sont robustes au bruit [Khotanzad et Lu, 1990]. Ainsi, nous utiliserons les moments de Zernike pour détecter les changements de postures dans nos séquences d'images.

ANNEXE G

Les séquences de postures de la main

Figure G.1: Exemple d'une séquence Clic

Figure G.2: Exemple d'une séquence Stop-Grasp-OK

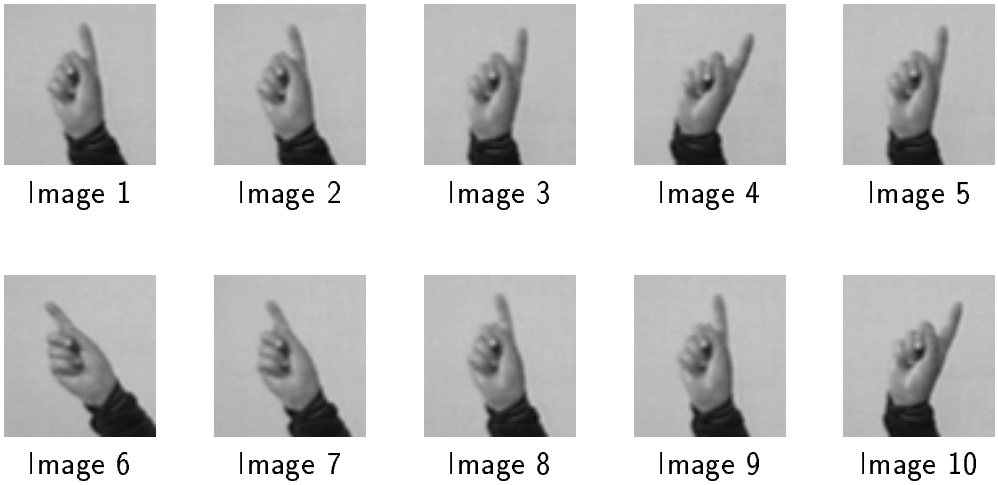


Figure G.3: Exemple d'une séquence No

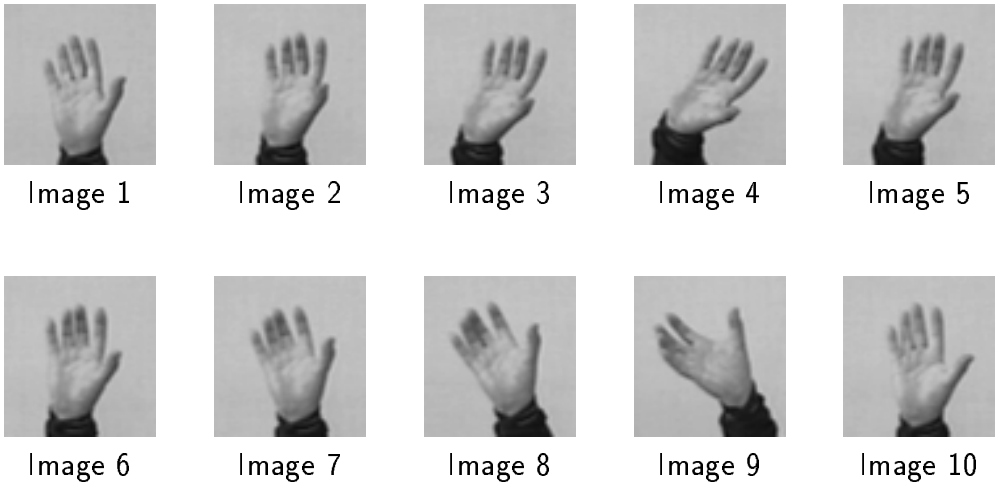


Figure G.4: Exemple d'une séquence Rotate

Bibliographie

Abramowitz, M. et Stegun, I. (1970). *Handbook of Mathematical Functions*. Dover Publishers Inc.,New York.

Ackley, D., Hinton, G., et T., S. (1985). A learning algorithm for boltzmann machines. In *Cognitive Science*, volume 9, pages 147–169.

Ahmad, S. (1994). A usable real-time 3d hand tracker. *Asilomar Conference on Signals, Systems and Computer*, (28). Interval Research Corporation, www.interval.com.

Ahmad, S. et Tresp, V. (1993). Classification with missing and uncertain inputs. In *International Conference on Neural Networks*, volume 3, pages 1949–1954.

Arsenaud, H. et Sheng, Y. (1986). Properties of the circular harmonic expansion for rotation-invariant pattern recognition. In *Applied optics*, volume 25 of *18*, pages 3225–3229.

Baldi, P. et Chauvin, Y. (1996). Hybrid modeling, HMM/NN architecture, and protein applications. *Neural Computation*, 8(7):1541–1565.

Baudel, T. et Beaudouin-Lafon, M. (1993). Charade: Remote control of objects using free-hand gestures. *Communications of the ACM*, 36(7):28–35.

Baudel, T. et Braffort, A. (1993). Reconnaissance de gestes de la main en environnement reel. *Actes de Informatique'93 - L'interface des mondes reels et virtuel*, pages 207–216. Montpellier.

Baum, L. (1972). An inequality and associated maximization technique in statistical estimation of probabilistic functions of markov processes. *Inequalities*, 3:1–8.

Belkasim, S., Shridhar, M., et Ahmadi, M. (1991). Pattern recognition with moment invariants: A comparative study and new results. In *Pattern Recognition*, volume 12, pages 1117–1138.

Bengio, Y.and Frasconi, P. (1995). An Input Output HMM Architecture. In *Advances in Neural Information Processing Systems 7*, San Mateo CA. Morgan Kaufmann.

Bengio, Y., De Mori, R., Flammia, G., et Kompe, R. (1992). Global optimization of Neural Network-Hidden Markov Model hybrid. *IEEE Transactions on Neural Networks*, 3(2):252–259.

Bengio, Y., LeCun, Y., Nohl, C., et Burges, C. (1995). Lerec: A NN/HMM hybrid for on-line handwriting recognition. *Neural Network*, 7(6):1289–1303.

Bengio, Y. et Simard, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, (5):157–166.

Binford, T. et Nalwa, V. (1986). On detecting edges. *IEEE Pattern Analysis and Machine Intelligence*, 8(6).

Blayo, F. et Verleysen, M. (1996). *Les réseaux de neurones artificiels*. Presses Universitaires de France, que sais-je ? edition.

Bobick, A. et Davis, J. (1996). Real-time recognition of activities using temporal templates. In *International Conference on Automatic Face and Gesture Recognition*.

Boehm, K., Broll, W., et Sokolewicz, M. (1994). Dynamic gesture recognition using neural networks: A fundament for advanced interaction construction. *SPIE, Conference Electronic Imaging Science and Technology*. San Jose California, USA.

Bottou, L. (1991). *Une approche théorique de l'apprentissage connexionniste: applications à la reconnaissance de la parole*. PhD thesis, Université de Paris 11, Orsay, France.

Bourlard, H. et Morgan, N. (1993). *Connectionnist Speech Recognition. A Hybrid Approach*, volume 247. The Kluwer international series in engineering and computer science, boston: kluwer academic publisher edition.

Bourlard, H. et Wellekens, C. (1990). Links between Hidden Markov Models and Multilayer Perceptrons. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:1167–1178.

Braffort, A. (1996). *Reconnaissance et comprehension de gestes, application a la langue des signes*. PhD thesis, Paris XI.

Bröckl-Fox, U. (1995). Real-time 3d interaction with up to 16 degrees of freedom form monocular video image flow. In *International Workshop on Automatic Face and Gesture Recognition*, pages 172–178. Zurich, Switzerland.

Breiman, L. (1996). Bagging predictor. *Machine Learning*, 26(2):123–140.

Bridle, J. (1990). Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. *Advances in Neural Information Processing Systems*, 2:211–217. Morgan Kaufmann.

Brown, P. (1987). *The Acoustic-Modeling problem in Automatic Speech Recognition*. PhD thesis, Dept. Of Computer Science, Carnegie-Mellon University.

Broyden, C. (1970). The convergence of a class of double-rank minimization algorithms. *J. Inst. Maths. Applics.*, 6:76–90.

Buxton, W. et Myers, B. (1986). A study in two-handed input. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 321–326.

Cadoz, C. (1994). Le geste canal de communication Homme/Machine - la communication instrumentale. *Techniques et Sciences Informatiques*, 13:31–61.

Cagin, J. (1993). *Une étude sur la reconnaissance de formes dynamiques - Application à la langue des signes*. ENSTA Paris. Projet de fin d'étude.

Campbell, L. et Bobick, A. F. (1995). Recognition of human body motion using phase constraints. In *International Conference on Computer Vision*, pages 624–630. Cambridge MA, MIT TR number 309.

Campbell, L. W., Becker, D. A., Azarbayejani, A., Bobick, A. F., et Pentland, A. (1996). Invariant features for 3d gesture recognition. In *International Workshop on Automatic Face and Gesture Recognition*, pages 157–162. Killington, Vermont.

Canny, J. (1986). A computational approach to edge detection. *IEEE Pattern Analysis and Machine Intelligence*, 8(6).

Cassell, J., Bickmore, T., Billinghurst, M., Campbell, L., Chang, K., Vilhjálmsson, H., et Yan, H. (1999). Embodiment in conversational interfaces: Rea. In *ACM Conference on Human Factors in Computing Systems CHI'99 Conference Proceedings, Pittsburgh, PA*.

Changeux, J.-P. (1983). *L'homme neuronal*. Fayard, pluriel edition.

Cho, S.-B. et Kim, J. H. (1995). An HMM/MLP architecture for sequence processing. *Neural Computation*, 7(2):358–369.

Cipolla, R., Okamoto, Y., et Kuno, Y. (1993). Robust structure form motion using motion parallax. In *International Conference on Computer Vision*, pages 374–382.

Codella, C., Jalili, R., Koved, L., et al. (1992). Interactive simulation in a multi-person virtual world. In *ACM Conference on Human Factors in Computing Systems*, volume 35, pages 329–334. CHI'92.

Cohen, C. J., Conway, L., et Koditschek (1996). Dynamical system representation, generation, and recognition of basic oscillatory motion gestures. In *International Workshop on Automatic Face and Gesture Recognition*, pages 60–65. Killington, Vermont.

Collobert, M., Feraud, R., Le Tourneur, G., Bernier, O., Viallet, J., Mahieux, Y., et Collobert, D. (1996). Listen: A system for locationg and tracking individual speakers. In *International Workshop on Automatic Face and Gesture Recognition*, pages 283–288. France Telecom, CNET.

Cootes, T., Taylor, C., Cooper, D., et Graham, J. (1995). Active shape models - their training and applications. *Computer Vision and Image Understanding*, 61:38–59.

Cybenko, G. (1989). Approximation by superposition of sigmoïdal functions. *Mathematics of Control, Signal and Systems*, vol 2:303–314.

Darrell, T. et Pentland, A. (1993). Space-time gestures. In *Computer Vision and Pattern Recognition CVPR'93*, pages 335–340.

Darrell, T. J. et Pentland, A. P. (1992). Recognition of space-time gestures using a distributed representation. Technical Report 197, M.I.T. Media Lab Perceptual Computing. Annee a verifier.

Davis, J. et Shah, M. (1993). Gesture recognition. Technical report, Department of Computer Science, University of Central Florida. CS-TR-93-11.

Davis, J. et Shah, M. (1994). Recognizing hand gestures. In *ECCV94*, pages A:331–340.

de l'Epée, A. (1784). *La véritable manière d'instruire les sourds et muets.* Fayard, Paris, 1984 edition. Corpus des oeuvres de philosophie en langue française.

Dempster, A., Laird, N., et Rubin, D. (1977). Maximum-likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistical Society*, 39:1–38.

Deriche, R. (1987). Using canny's criteria to derive a recursively implemented optimal edge detector. *International Conference on Computer Vision*, 1(2):167–187.

DeVries, B. et Principe, J. (1992). The gamma model: A new neural model for temporal processing. *Neural Computation*, (5):565–576.

Downton, A. et Drouet, H. (1991). Image analysis for model-based sign language coding. *International Conference on Image Analysis and Processing*, (6):637–644. Progress in image analysis and processing II.

Essa, I. et Pentland, S. (1995). Facial expression recognition using a dynamic model and motion energy. In *IEEE International Conference on Computer Vision.*

Etoh, M., Tomono, A., et Kishino, F. (1991). Stéréo-based description by generalized cylinder complexes from occluding contours. *Systems and Computers in Japan*, 22(12):79–89.

Fahlman, S. et Lebiere, C. (1991). The cascade-correlation learning architecture. Technical report, School of Computer Science, Carnegie Mellon University.

Fahlman, S. E. et Lebiere, C. (1990). The cascade-correlation learning architecture. *Advances in Neural Information Processing Systems*, (2):524–532. NIPS 90, Los Altos CA.

Fels, S. et Hinton, G. (1993). Glove-talk: A neural networks interface between a data-glove and a speech synthesizer. *IEEE Transactions on Neural Networks*, 4:2–8.

Feraud, R. (1997). Pca, neural networks and estimation for face detection. *NATO ASI, Face Recognition: from Theory to Applications.*

Figueiredo, M., Böhm, K., et Teixera, J. (1993). Advanced interaction techniques in virtual environments. *Computers and Graphics*, 17(6):655–661.

Fletcher, R. (1970). A new approach to variable metric algorithm. *Computer Journal*, 13:317–322.

Fletcher, R. et Reeves, C. (1964). Function minimization by conjugate gradients. *Computer Journal*, 7:149–154.

Frasconi, P., Gori, M., et Soda, G. (1992). Local feedback multilayered networks. *Neural Computation*, (4):120–130.

Féraud, R., O., B., Viallet, J., et Collobert, M. (2000). A fast and accurate face detector for indexation of faces images. In *FG'2000 Conference on Automatic Face and Gesture Recognition.* Grenoble, 28-30 mars 2000.

Freeman, W. et Roth, M. (1995). Orientation histograms for hand gesture recognition. In *International Workshop on Automatic Face and Gesture Recognition*, pages 296–301. Zurich, Switzerland.

Freeman, W. T. et Adelson, E. H. (1991). The design and use of steerable filters. In *IEEE Transaction Pattern Analysis and Machine Intelligence*, volume 13 of *9*, pages 891–906. MIT - Media Lab and Dept of Brain and Cognitive Sciences.

Freeman, W. T. et Weissman, C. D. (1995). Television control by hand gestures. In *International Workshop on Automatic Face and Gesture Recognition*, pages 179–183. Zurich, Switzerland.

Freund, Y. et Schapire, R. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and Systems Sciences*, 55(1).

Frosini, P. (1991). Measuring shapes by size functions. In *Proceedings SPIE on Intelligent Robotic Systems.* Boston.

Golfarb, D. (1970). A family of variable metric updates derived by variational means. *Mathematics of Computing*, 24:23–26.

Gravila, D. (1996). Hermite deformable contours. In *International Conference on Pattern Recognition.* ICPR'96, Vienna.

Gravila, D. et Davis, L. (1995). Towards 3-d model-based tracking and recognition of human movement: a multi-view approach. In *International Conference on Automatic Face and Gesture Recognition*, number 1, pages 272–277.

Grimes, G. J. (1987). Digital data entry glove interface device. Technical report, Bell Telephone Laboratories. United States Patent 4, 414, 537, Murray Hill, NJ, November 8, 1983.

Guiard, Y. (1987). Asymetric division of labor in human skilled bimanual action : The kinematic chain as a model. *Journal of Motor Behavior*, 19(4):486–481.

Guyon, I., Vapnik, V., Boser, B., Bottou, L., et Solla, S. (1992). Structural risk minimization for character recognition. *Advances in Neural Information Processing Systems*, (4):471–479. NIPS 91, San Mateo CA, J. E. Moody et al., editor, Morgan Kaufmann.

Hagan, M., Demuth, H., et Beale, M. (1996). *Neural Network Design*. Boston, MA:PWS Publishing.

Harling, P. (1993). Gesture input using neural networks. Technical report, Dept of Computer Science University of York. BSc degree in Computer Science.

Hebb, D. (1949). The organization of behaviour. *Wiley*.

Hinton, G. (1986). Learning distributed representations of concepts. In *Annual Conference of the Cognitive Science Society*, volume 8, pages 1–12.

Hopfield, J. (1982). Neural networks and physical systems with emergent collective abilities. In *Proceedings of the National Academy of Sciences, USA*, volume 81, pages 3088–3092.

Hornik, K., Stinchombe, M., et White, H. (1989). Multilayer feedforward networks are approximators. *Neural Networks*, vol 2:359–366.

Huang, T. S. et Pavlovic, V. I. (1995). Hand gesture modeling, analysis and synthesis. In *International Workshop on Automatic Face and Gesture Recognition*, pages 73–79. Zurich, Switzerland.

Huang, T. S., Pavlovic, V. I., et Sharma, R. (1996). Gestural interface to a visual computing environment for molecular biologists. In *International Workshop on Automatic Face and Gesture Recognition*, pages 30–35. Killington, Vermont.

Hunter, E., Schlenzig, J., et Jain, R. (1995). Posture estimation in reduced-model gesture input systems. In *International Workshop on Automatic Face and Gesture Recognition*, pages 290–295. Zurich, Switzerland.

Industries, W. (1991). *Press release*. W Industries. Leicester, UK.

Iwaniec, H. et Mozzochi, C. (1988). On the Divisor and Circle Problems. *J. Number Theory*, 29:60–63.

Jacobs, R. A. et Jordan, M. I. (1991). A competitive modular connexionist architecture. In *Neural Information Processing System 3*.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., et Hinton, G. E. (1991). Adaptative mixtures of local experts. *Neural Computation*, 3:79–87.

Kabbash, P., Buxton, W., et Sellen, A. (1994). Two-handed input in a compond task. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 417–423.

Kabbash, P., MacKenzie, I., et Buxton, W. (1993). Human performance using computer input devices in the preferred and non-preferred hands. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 474–481.

Kandel, E. et Schwartz, J. (1991). *Principle of Neural Science*. New York: Elsevier/North Holland.

Kass, M., Witkin, A., et Terzopoulos, D. (1997). Snakes: Active contour models. *International Conference on Computer Vision*, pages 259–268.

Kendon, A. (1982). *The study of gesture : some remarks on its history*, volume 2, pages 45–62. Recherche Sémiotique/Semiotic Inquiry.

Kendon, A., Nespoulous, J.-L., Peron, P., et Lecours, A. (1986). Current issues in the study of gesture. *The Biological Foundations of Gestures: Motor and Semiotic Aspects*, pages 23–47. Lawrence Erlbaum Assoc.

Kervrann, C. et Heitz, F. (1995). Learning structure and deformation modes of nonrigid objects in long image sequences. In *International Workshop on Automatic Face and Gesture Recognition*.

Khotanzad, A. et Lu, J. (1990). Classification of invariant image representations using a neural network. In *IEEE transaction on Acoustics, Speech and Signal Processing*, volume 38 of *6*, pages 1028–1190.

Kjeldsen, R. (1994). Visual hand gesture interpretation. *IEEE Computer Society Workshop on Non-Rigid and Articulate Motion*. Austin, TX.

Koch, R. (1993). Dynamic 3d scene analysis through synthetic feedback control. In *IEEE Pattern Analysis and Machine Intelligence*, volume 15, pages 556–569.

Kohonen, T. (1995). *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer, Berlin, Heidelberg, New York, second extended edition edition.

Kramer, J. et Leifer, L. (1989). The talking glove: An expressive and receptive "verbal" communication aid for deaf, deaf-blind and nonvocal. Technical report, Department of Electrical Engineering. Stanford University.

Krueger, M. (1993). Environmental technology: Making the real world virtual. *Communications of the ACM*, 36:36–37.

Kuch, J. et Huang, T. (1995). Vision based hand modeling and tracking. In *International Conference on Computer Vision*. Cambridge, MA.

Lanitis, A., Taylor, C., Cootes, T., et Ahmed, T. (1995). Automatic interpretation of human faces and hand gestures using flexible models. In *International Workshop on Automatic Face and Gesture Recognition*, pages 98–103. Zurich, Switzerland.

Le Cun, Y., Jackel, D., Boser, B., Denker, J., et al (1989). Handwritten digit recognition: Application of neural netwok ships and automatic learning. *IEEE Communications Magazine*, 27:41–46.

LeCun, Y. (1985). Une procédure d'apprentissage pour réseau a seuil assymétrique. *Cognitiva: A la frontière de l'Intelligence Artificielle des Sciences de la Connaissance des Neurosciences*, pages 599–604.

LeCun, Y. (1987). *Modèle Connexionnistes de l'Apprentissage*. PhD thesis, Universite de Paris 6.

LeCun, Y., Bottou, L., Orr, G., et Müller, K. (1998). Neural Networks: Tricks of the trade. *Lecture Notes in Computer Science, Springer Verlag*, (1524). Orr, Genevieve B. and Müller Klaus-Robert editors.

LeCun, Y., Denker, J., et Solla, S. (1990). Optimal brain damage. In *Advances in Neural Networks Information Processing Systems*, volume 2, pages 598–605.

Lee, J. et Kunii, T. (1993). *Constraint-based hand animation*, pages 110–127. Models and techniques in computer animation. Tokyo: Springer-Verlag.

Leighton, R. et Conrad, B. (1991). The autoregressive backpropagation algorithm. In *Proceedings of the International Joint Conference on Neural Networks*.

Lemaire, V. (1999). *Une nouvelle fonction de coût régularisante dans les réseaux de neurones artificiels: Application à l'estimation des temps de blocage dans une nœud ATM*. PhD thesis, Université Paris VI.

Levenberg, K. (1944). A method for the solution of certain problems in least squares. *Quart. Apl. Math.*, 2:164–168.

Liao, S. et Pawlak, M. (1996). On Image Analysis by Moments. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18:254–266.

Liao, S. et Pawlak, M. (1998). On the Accuracy of Zernike Moments for Image Analysis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20:1358–1364.

Maggioni, C. (1993). A novel gestural input device for virtual reality. In *IEEE Annual Virtual Reality International Symposium*, pages 118–124.

Magnenat-Thalmann, N. et Thalman, D. (1990). *Computer Animation: Theory and Practice*. New York: Springer Verlag, 2nd rev edition.

Marcel, S. (1999). Hand posture recognition in a body-face centered space. In *Conference on Human Factors in Computing Systems CHI'99*.

Marcel, S., Bernier, O., et Collobert, D. (1999). Reconnaissance de la main pour les interfaces gestuelles. In *CORESA'99*.

Marquardt, D. (1963). An algorithm for least-squares estimation of nonlinear parameters. *SIAM J. Appl. Math.*, 11:431–441.

Martin, J., Devin, V., et Crowley, J. L. (1998). Active hand tracking. *3rd IEEE Conference on Automatic Face and Gesture Recognition*, 1(3):573–578. FG'98, Nara, Japan, 14-16 April 1998.

McCulloch, W. et Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bull Math. Biophysics*, 5:115–133.

McNeill, D. (1992). *Hand and Mind: What gestures reveal about thought.* Chicago Press.

Ming-Kuel, H. (1962). Visual pattern recognition by moment invariants. *IRE Transaction on Information Theory*, 2:179–187.

Minsky, M. et Papert, S. (1969). *Perceptrons.* Cambridge, MIT press.

Moghaddam, B. et Pentland, A. (1995). Maximum likelihood detection of faces and hands. In *International Workshop on Automatic Face and Gesture Recognition*, pages 122–128. Zurich, Switzerland.

Moghaddam, B. et Pentland, A. (1997). Probalistic visual learning for object representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):696–710.

Moody, B. (1983). *La langue des signes. Histoire et grammaire*, volume 1. Paris.

Morrel-Samuels, P. (1990). Clarifying the distinction between lexical and gestural commands. *International Journal of Man-Machine Studies*, 32:581–590.

Mozer, M. (1989). A focused back-propagation algorithm for temporal pattern recognition. *Complex Systems*, 3:349–381.

Murakami, K. et Taguchi, H. (1991). Gesture recognition using recurrent neural networks. In *Conference on Human Interaction*, pages 237–242. Fujitsu Lab, Conference on Human Factors in Computing Systems CHI'91, ACM, New Orlean, Louisiana.

Napier, J. (1980). *Hands.* Pantheon Books, new york edition.

O'Rourke et Badler, N. (1980). Model-based image analysis of human motion using constraint propagation. In *IEEE Transaction Pattern Analysis and Machine Intelligence*, volume 2, pages 522–536.

Parker, D. (1985). Learning logic. Technical report, Center for Computational Research in Economics and Managment Science, MIT, Cambridge, MA.

Pavlovic, V. I., Sharma, R., et Huang, T. S. (1997). Visual interpretation of hand gestures for human-computer interation: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):677–695.

Pentland, A., Azarbayejani, A., et Christopher, W. (1996). Real-time 3d tracking of human body. Technical Report 374, M.I.T. Media Lab Perceptual Computing. Appears in Proceedings of IMAGE'COM 96, Bordeaux, France.

Pentland, A. et Liu, A. (1995). Modeling and prediction of human behavior. Technical Report 433, M.I.T. Media Lab Perceptual Computing.

Quek, F. K. (1994). Toward a vision-based hand gesture interface. Technical report, University of Illinois at Chicago. Virtual Reality Software and Technology Conference.

Rabiner, L. R. (1989). A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–285.

Rehg, J. et Kanade, T. (1995). Model-based tracking of self-occluding articulated objects. In *ICCV95*, pages 612–617.

Rehg, J. M. (1995). *Visual Analysis of High DOF Articulated Objects Application to Hand Tracking*. PhD thesis, Carnegie Mellon University.

Rehg, J. M. et Kanade, T. (1993). Digiteyes: Vision-based human hand tracking. Technical report, Carnegie Mellon University. Proceedings of European Conference on Computer Vision, May 1994, Stockholm, Sweden.

Rosenblatt, F. (1958). The perceptron : a probabilistic model for information storage and organization in the brain. *Psychological Review*, vol 65:386–408.

Rosenblatt, F. (1962). *Principles of Neurodynamics*. New York: Spartan.

Rubine, D. (1991). *The automatic recognition of gestures*. PhD thesis, Carnegie Mellon University.

Rumelhart, D., Hinton, G., et R.J., W. (1986). Learning internal representations by error propagation. In *Parallel Distributed Processing Explorations in the Microstructure of Cognition*, volume 1, pages 318–362. Rumelhart, D.E. and McClelland, J.L., Cambridge MA, MIT press.

Sagawa, H., Sakou, H., et Abe, M. (1992). Sign language translation system using continuous dp matching. In *MAV'92 - IAPR Workshop on Machine Vision Applications*, pages 339–342. Tokyo.

Sastry, P., Santharam, G., et Unnikrishnan, K. (1994). Memory neuron networks for identification and control of dynamical systems. *IEEE Transactions on Neural Networks*, (5):306–319.

Saxe, D. et Foulds, R. (1996). Toward robust skin identification in video images. In *International Workshop on Automatic Face and Gesture Recognition*, pages 379–384. Killington, Vermont.

Schlenzig, J., Hunter, E., et Jain, R. (1994a). Recursive identification of gesture inputs using hidden markov models. In *WACV94*, pages 187–194.

Schlenzig, J., Hunter, E., et Jain, R. (1994b). Vision based hand gesture interpretation using recursive estimation. In *Proceedings of the 28thAsilomar Conference on Signals, Systems and Computer*.

Segen, J. (1993). Controlling computers with gloveless gestures. In *Virtual Reality Systems*.

Shanno, D. (1970). Conditioning of quasi-newton methods for function minimization. *Mathematics of Computing*, 24:647–656.

Shen, J. et Castan, S. (1986). An optimal linear operator for edge detection. *Image Vision and Computing*.

Srinivasan, B., Prasad, U., et Rao, N. (1994). Back propagation through adjoints for the identification of nonlinear dynamic systems using recurrent neural networks. *IEEE Transactions on Neural Networks*, (5):213–228.

Starner, T. et Pentland, A. (1995). Real-time american sign language recognition form video using hidden markov models. Technical Report 375, M.I.T. Media Lab Perceptual Computing. ISCV'95.

Sturman, D. J. (1992). *Whole-hand Input*. PhD thesis, Massachusetts Institute of Technology.

Su, S. et Furuta, R. (1993). Virtual panel architecture: a 3d gesture framework. *IEEE Annual Virtual Reality International Symposium*, pages 387–393.

Teague, M. (1980). Image analysis via the general theory of moments. *Journal of the Optical Society of America*, 70:920–930.

Teh, C. et Chin, R. (1988). On image analysis by the method of moment. *IEEE transaction on Pattern Analysis and Machine Intelligence*, 10(4):496–513.

Thieffry, S. (1981). *"Hand gestures"in The Hand*, pages 488–492. Tubiana, R. Philadelphia, PA: Sanders, 1981.

Thompson, D. (1981). Biomechanics of the hand. *Perspectives in Computing*, 1:12–19.

Torige, A. et Kono, T. (1992). Human-interface by recognition of human gestures with image processing recognition of gesture to specify moving directions. *IEEE International Workshop on Robot and Human Communication*, pages 105–110.

Triesch, J. et Malsburg, C. (1996). Robust classification of hand posture against complex backgrounds. In *Proceedings of the second International Conference on Automatic Face and Gesture Recognition*, pages 170–175. Killington, Vermont, USA.

Triesch, J. et Malsburg, C. (1998). A gesture interface for human-robot-interaction. In *Proceedings of the third International Conference on Automatic Face and Gesture Recognition*, pages 546–551. Nara, Japan.

Tsoi, C. et Back, A. (1994). Locally recurrent globally feedforward networks: a critical review of architectures. *IEEE Transactions on Neural Networks*, (5):229–239.

Tubiana, R. (1981). *The Hand*, volume 1. Philadelphia, PA: Sanders, 1981.

Uras, C. et Verri, A. (1995). Hand gesture recognition form edge maps. In *International Workshop on Automatic Face and Gesture Recognition*, pages 116–121. Zurich, Switzerland.

Vaillant, R. et Darmon, D. (1995). Vision based hand pose estimation. In *International Workshop on Automatic Face and Gesture Recognition*, pages 356–361. Zurich, Switzerland.

Vapnik, V. (1982). Estimation of dependencies based on empirical data. New-York, NY, USA.

Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer Verlag, new york heidelberg berlin edition.

Vapnik, V. et Chervonenkis, A. (1971). On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its applications*, vol 16:264–271.

Vapnik, V. et Chervonenkis, A. (1974). *Theory of Pattern Recognition*. Nauka, moscow edition. in Russian.

VPL Research, I. (1987). *DataGlove model2 users manual*. VPL Research, Inc. Redwood City, CA, 1987.

Waibel, A. (1989). Modular construction of Time Delay Neural Networks for speech processing. In *Neural Computation*, volume 1.

Waibel, A., Hanazawa, T., G., H., Shikano, K., et Lang, K. (1989). Phoneme recognition using time-delay neural netwoks. *IEEE transactions on Acoustics, Speech and Signal Processing*, 37:328–339.

Waibel, A. et Vo, M. T. (1993). A multimodal human-computer interface: Combination of gesture and speech recognition. In *Proceedings of Inter Conference on Human Factors in Computing Systems CHI'93*. Amsterdam.

Wan, E. (1993). *Finite Impulse Response Neural Networks with Applications in Time Series Prediction*. PhD thesis, Stanford University.

Wang, C. et Cannon, D. (1993). A virtual end-effector pointing system in point-and-direct robotics for inspection of surface flaws using a neural network based skeleton transform. In *IEEE International Conference on Robotics and Automation*, volume 3, pages 784–789.

Watanabe, T. et Yachida, M. (1998). Real-time gesture recognition using eigenspace from multi input image sequences. In *Int. Conf. on Automatic Face and Gesture Recognition*, pages 428–433. Nara, Japan.

Werbos, P. (1974). *Beyond Regression: New Tools for Prediction and Analysis in Behavioral Sciences*. PhD thesis, Harvard University, Cambridge MA.

Widrow, B. et Hoff, M. (1960). Adaptative switching circuits. *IRE WESCON Convention Record*, vol 4:96–104.

Williams, R. et Zipser, D. (1989). Experimental analysis of the real-time recurrent learning algorithm. In *Connection Science*, volume 1, pages 87–111.

Wilson, A. D. et Bobick, A. F. (1995). Configuration states for the representation and recognition of gesture. In *International Workshop on Automatic Face and Gesture Recognition*, pages 129–134. Zurich, Switzerland.

Winter, D. (1979). *Biomechanics and Motor*. John Wiley and Sons. Control of Human Mouvement.

Wren, C., Azarbayejani, A., Darrell, T., et Pentland, A. (1997). Pfinder:real-time tracking of the human body. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 19 of *7*, pages 780–785.

Wren, C. et Pentland, A. (1998). Dynamic models of human motion. *3rd IEEE International Conference on Automatic Face and Gesture Recognition*, 1(3):22–27. April 14-16, 1998, Nara, Japan.

Yang, J., Lu, W., et Waibel, A. (1998). Skin color modeling and adaptation. In *Proceedings of the 3^{rd} Asian Conference on Computer Vision*.

Yang, M.-H. et Ahuja, N. (1999). Gaussian mixture model for human skin color and its applications in image and video databases. In *SPIE, Conference on Storage and Retrieval for Image and Video Databases*.

Zernike, F. (1934). Begungstheorie des Schneidenverfahrens und seiner verbesserten Form,der Phasenkontrastmethode. *Physica*, 1:689–701.

Résumé

La détection et l'analyse des personnes est un problème fondamental en vision par ordinateur et plus spécifiquement dans la conception des interfaces gestuelles. Les interfaces gestuelles basées sur l'image constituent la voie la plus naturelle pour la construction d'interfaces homme-machine évoluées. Ainsi, les machines seraient plus faciles à utiliser en associant la commande gestuelle à la commande vocale.

Cette thèse a pour ambition de présenter des techniques d'apprentissage statistique pour la reconnaissance des gestes de la main, et plus généralement pour le traitement d'images fixes et de séquences d'images animées. En effet, il est nécessaire de distinguer deux aspects des gestes de la main :

- l'aspect statique est caractérisé par une posture de la main dans une image,
- l'aspect dynamique est défini soit par la trajectoire de la main, soit par l'enchaînement de postures de la main dans une séquence d'images.

Les techniques d'apprentissage proposées se basent sur une approche générative des réseaux de neurones artificiels (RN) et des modèles de Markov cachés (HMM). En effet, en apprentissage statistique, on possède presque toujours des données relatives aux observations que l'on souhaite modéliser (les exemples). En revanche, on ne connaît pas toujours le reste de l'univers, c'est-à-dire toutes les observations qui ne sont pas celles que l'on désire (les contre-exemples). Il est donc difficile d'apprendre à séparer les exemples et les contre-exemples. L'approche générative se fonde sur la modélisation de la distribution de probabilité des exemples. L'évaluation de la distance entre une observation inconnue et cette distribution, permet de décider si cette observation est un exemple ou n'en est pas un.

D'un côté, les RN sont très efficaces pour des tâches de reconnaissance de formes, mais sont mal adaptés à la reconnaissance de données de nature temporelle. Nous montrons l'utilisation d'un modèle génératif de RN pour la reconnaissance statique des postures de la main dans des images.

D'un autre côté, les HMM sont bien adaptés à la modélisation des séquences de données temporelles, mais ne permettent pas de discriminer efficacement des séquences de classes distinctes. Il semble donc naturel d'étudier un modèle hybride qui possède les propriétés des RN et des HMM. Ainsi, nous proposons une approche générative neuro-markovienne (IOHMM génératif) pour la reconnaissance dynamique des gestes de la main dans des séquences d'images.

Pour finir, ce mémoire présente un démonstrateur de reconnaissance des gestes de la main (GHOST) qui utilise les techniques décrites.

Mots clés : Réseaux de neurones artificiels, modèles de Markov cachés, IOHMM génératifs, vision par ordinateur, reconnaissance des gestes de la main.