République Algérienne Démocratique et Populaire
Ministère De L'enseignement Supérieur,
De La Recherche Scientifique
École Nationale Polytechnique

المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique

C.D.T.A

Département d'Automatique

Mémoire de fin d'étude en vue de l'obtention du diplôme d'ingénieur d'état en Automatique

Compréhension de scène par détection et suivi de l'être humain pour un robot mobile autonome

-Application au robot B21r-

Sifeddine BENAHMED *et* Abdelhakim DEBIB

Sous la direction de :

Mr. L. ABDELOUEL Chargé de cours *ENP*
Mr. N. OUADAH Maitre de recherche *CDTA*

Présenté et soutenu publiquement le 20/06/2016

Composition du jury :

Président : Mr. H. CHEKIREB Professeur *ENP*
Rapporteur : Mr. L. ABDELOUEL Chargé de cours *ENP*
Mr. N. OUADAH Maitre de recherche *CDTA*
Examinateur : Mr. H. ACHOUR Chargé de cours *ENP*

Promotion 2016

Ecole Nationale Polytechnique : 10, Avenue Hassen Badi 16200 EL-Harrach, Alger

République Algérienne Démocratique et Populaire
Ministère De L'enseignement Supérieur,
De La Recherche Scientifique
École Nationale Polytechnique

المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique

C.D.T.A

Département d'Automatique

Mémoire de fin d'étude en vue de l'obtention du diplôme d'ingénieur d'état en Automatique

Compréhension de scène par détection et suivi de l'être humain pour un robot mobile autonome

-Application au robot B21r-

Sifeddine BENAHMED *et* Abdelhakim DEBIB

Sous la direction de :

Mr. L. ABDELOUEL Chargé de cours *ENP*
Mr. N. OUADAH Maitre de recherche *CDTA*

Présenté et soutenu publiquement le 20/06/2016

Composition du jury :

Président : Mr. H. CHEKIREB Professeur *ENP*
Rapporteur : Mr. L. ABDELOUEL Chargé de cours *ENP*
Mr. N. OUADAH Maitre de recherche *CDTA*
Examinateur : Mr. H. ACHOUR Chargé de cours *ENP*

Promotion 2016

Ecole Nationale Polytechnique : 10, Avenue Hassen Badi 16200 EL-Harrach, Alger

ملخص

إن روبوت الخدمة المستقل يجب أن يكون قادر على إستشعار المحيط الذي يعمل فيه بطريقة صحيحة. بما أن الإنسان هو الكائن الأكثر أهمية الذي سيواجهه الروبوت، فإن وحدة كشف وتتبع الأشخاص أصبحت رئيسية لأي روبوت خدمة.
الهدف من هذا العمل هو إنشاء تطبيق يعمل تحت نظام تشغيل الروبوت ضصض للاستشعار و كشف الأشخاص والتعرف على الوجوه باستخدام المعلومات الواردة من كاميرا كينيكت.
العمل تم علي منصة تجريبية ١٢ر روبوت تابع لمركز تطوير التكنولوجيا المتقدمة.

كلمات مفاتيح:الروبوت المتنقل، كشف وتتبع الأشخاص، التعرف على الوجوه ، روبوت ، كاميرا

Abstract

An autonomous service robot must be able to perceive correctly the environment in which it operates. As people is the most important object that the robot will encounter in its navigation, a detection and tracking module of humans has become a key point for any service robot. The objective of this work is to implement under a ROS environment a detection and tracking module of people on the experimental platform "the robot B21r" using the information from a RGB-D camera (Kinect).

Key words : Mobile robot, detection of people, tracking of people, person recognition, Robot B21r, RGB-D caméra, ROS.

Résumé

Un robot de service autonome doit être capable de percevoir d'une manière correcte l'environnement dans lequel il évolue. Comme l'être humain représente l'objet le plus important que le robot va rencontrer dans sa navigation, un module de détection et de suivi de personnes est devenu un point clé pour tout robot de service.
L'objectif de ce travail est d'implémenter sous un environnement ROS un module de détection et de suivi de l'être humain sur la plateforme expérimentale "le robot B21r" du Centre de Développement des Technologies Avancée utilisant les informations issues d'une caméra RGB-D (la Kinect).

Mots clés : Robot mobile, détection de l'être humain, suivi de l'être humain, reconnaissance des personnes, Robot B21r, Caméra RGB-D, ROS.

Dédicace

Je dédie ce travail à mes très chères parents pour l'effort fournie et les immenses sacrifices consentis pour mon éducation et mon succès. Que الله *vous protège et vous garde pour moi et pour la famille.*

À ma soeur Ilhem, Pour toutes les peines et tous les sacrifices que tu as consentis pour mon éducation, sans toi je ne serais pas où j'en suis aujourd'hui.

Je dédie ce travail également:

À mes adorables sœurs,

À mon chère frère, Abdelmounaim et sa femme,

À mes amis et frères Sheikh Mohamed, Mehdi, Bassem, Tayeb, Lotfi, Cherif, Housseyn...

À Soulimane le frère que ne m'a pas donné ma mère,

À tous mes camardes automaticiens et polytechniciens,

À tous mes enseignants du primaire à l'université pour leur contribution à mon éducation.

Sifeddine,

Dédicace

Je dédie ce travail à mes très chères parents. Que الله *vous protège et vous garde pour moi et pour la famille.*

Ma mère, qui a œuvré pour ma réussite, de par son amour, son soutien, tous les sacrifices consentis et ses précieux conseils, pour toute son assistance et sa présence dans ma vie, reçois à travers ce travail aussi modeste soit-il, l'expression de mes sentiments et de mon éternelle gratitude.

Mon père, qui peut être fier et trouver ici le résultat de longues années de sacrifices et de privations pour m'aider à avancer dans la vie. Puisse Dieu faire en sorte que ce travail porte son fruit ; Merci pour les valeurs nobles, l'éducation et le soutient permanent venu de toi.

À mes très chères sœurs Hakima et Naila,

À mon très cher petit frère Dinel,

À toute la famille DEBIB et KHELFAT,

À mes amis et frères Mohammed, Abderrahmane, Djamel, Oussama, Boka, Anes, Amine, Oussama, Ahmed, Ziyad, et tous mes amis de Jijel et de l'ENP,

À tous ceux que j'aime et qui m'aiment,

Abdelhakim,

Remerciement

Tout d'abord, nous remercions الله ,le tout puissant de nous avoir accordé le savoir, le droit chemin, l'opportunité de poursuivre nos études et la force pour réaliser ce travail.

Nous remercions nos parents, qui nous ont soutenus tout au long de nos étude.

Nos remerciements les plus vifs s'adressent perticuliérement à Monsieur L.ABDELOUEL, notre promoteur à l'ENP et à Monsieur N.OUADAH et Madame Kh.CHAIB nos promoteurs au CDTA.

Nous remercions aussi, les membres du jury qui ont accepté d'évaluer notre travail.
Un grand merci à toute l'équipe de la division Productique et Robotique du CDTA auprès desquels nous avons acquis un savoir incommensurable.

Notre sincère remerciement et notre profonde gratitude vont également à toute l'équipe de l'Ecole Nationale Polytechnique pour les connaissances et le savoir faire qu'ils nous ont transmis pendant les cinq années de notre formation.

Table des matières

Résumé

Dédicaces

Remerciement

Table des matières

Liste des tableaux

Liste des figures

Liste des abréviations

Introduction Générale **18**

1 Généralité sur la robotique mobile **20**

1.1 Les Robots mobiles 20

1.2 Les types de plateforme mobile : 21

1.2.1 Les plates-formes holonomes 22

1.2.2 Les plates-formes différentielles 22

1.2.3 Les plates-formes omnidirectionnelles 23

1.2.4 Les plates-formes non holonomes 24

1.2.5 Les plates-formes à pattes 24

1.3 La robotique de service 24

1.4 La navigation autonome des robots mobiles 25

1.5 La détection de l'être humain 26

1.6 Le suivi de l'être humain 26

1.7 Conclusion 28

2 Techniques de détection de l'être humain **29**

2.1 Introduction 29

2.2 La perception dans la robotique mobile 29

2.4 La détection de l'être humain en 2D . . . 32
2.4.1 Méthode des Histogramme des Gradients Orientés . . . 33
2.4.1.1 Le choix de la fenêtre glissante : . . . 33
2.4.1.2 Le choix de la cellule . . . 33
2.4.1.3 Le calcul du gradient . . . 34
2.4.1.4 L'histogramme . . . 35
2.4.1.5 La normalisation des bloques . . . 36
2.4.1.6 La taille finale du descripteur . . . 37
2.4.1.7 Machine à vecteurs de support SVM . . . 37
2.4.2 Méthode de Viola et Jones pour la détection du visage . . . 38
2.4.2.1 Principe . . . 38
2.4.2.2 Apprentissage du classifieur . . . 39
2.4.2.3 Les caractéristiques . . . 39
2.4.2.4 L'image intégrale . . . 41
2.4.2.5 Algorithme d'apprentissage basé sur Adaboost . . . 41
2.4.2.6 Cascade de classifieurs . . . 41
2.5 Méthodologie de détection du squelette humain en 3D . . . 42
2.5.1 Etape 1 . . . 43
2.5.2 Etape 2 . . . 43
2.5.2.1 Etiquetage des parties du corps humain . . . 44
2.5.2.2 Caractéristiques de l'image de profondeur . . . 45
2.5.2.3 Méthode de classification . . . 46
2.5.2.4 Positionnement des joints (articulation) . . . 47
2.5.3 Etape 3 . . . 48
2.6 Conclusion . . . 48

3 Reconnaissance et suivi des personnes 49
3.1 Introduction . . . 49
3.2 Identification et reconnaissance faciale de personnes . . . 49
3.3 Principales difficultés de la reconnaissance de visage . . . 50
3.4 Méthodes de reconnaissance de visage . . . 52
3.4.1 Méthodes globales . . . 52
3.4.2 Méthodes locales . . . 53
3.4.3 Méthodes hybrides . . . 54
3.5 Principales taches en identification et reconnaissance . . . 54
3.5.1 Prétraitement des images . . . 55
3.5.1.1 Correction de l'illumination . . . 55
3.5.1.2 Correction de la pose de la tête . . . 56

3.5.2 Reconnaissance . . . 57
3.5.2.1 Phase apprentissage . . . 58
3.5.2.2 Phase reconnaissance . . . 58
3.5.2.3 Analyse Discriminante Linéaire (Fisher-Faces) . . . 59
3.5.3 Classification et identification des personnes . . . 59
3.6 Tracking de l'être humain . . . 60
3.6.1 Filtre de Kalman . . . 62
3.6.1.1 Algorithme du filtre de Kalman discret . . . 62
3.6.1.2 Application du filtre de Kalman pour le suivi des personnes 63
3.6.2 Suivi 3D du squelette humain . . . 64
3.7 Conclusion . . . 65

4 Implémentation et Résultats 67
4.1 Introduction . . . 67
4.2 Présentation de la plateforme expérimentale B21r du CDTA . . . 67
4.2.1 Présentation de l'environnement matériel . . . 67
4.2.2 Le Robot B21r . . . 67
4.2.3 Description de la Kinect . . . 68
4.2.3.1 Les composants de la Kinect . . . 69
4.2.3.2 Caractéristiques de la Kinect . . . 69
4.3 Présentation de l'environnement logiciel . . . 69
4.3.1 ROS . . . 69
4.3.1.1 Historique de ROS . . . 70
4.3.1.2 Architecture de ROS . . . 70
4.3.2 Outils logiciels pour la Kinect . . . 71
4.4 Les différentes tâches effectuées au niveau du CDTA . . . 72
4.4.1 Les tâches préliminaires . . . 72
4.4.2 Les tâches principales . . . 72
4.4.2.1 L'architecture prototype . . . 72
4.4.2.2 Implémentation des algorithmes . . . 73
4.4.2.3 Evaluation par des tests et des scénarios . . . 73
4.5 Détection du corps humain en 2D . . . 74
4.5.1 Cas d'une seule personne . . . 74
4.5.1.1 Scénario 1 . . . 74
4.5.1.2 Scénario 2 . . . 75
4.5.1.3 Scénario 3 . . . 75
4.5.1.4 Scénario 4 . . . 76
4.5.2 Cas de plusieurs personnes . . . 77

4.5.2.1 Scénario 1 . . . 77
4.6 Détection du visage en 2D . . . 77
4.6.0.2 Scénario 1 . . . 77
4.7 Détection du corps et du visage humain simultanément . . . 79
4.7.0.3 Scénario 1 . . . 79
4.8 Détection et suivi 3D du squelette humain . . . 80
4.8.1 Cas d'une seule personne . . . 81
4.8.1.1 Scénario 1 . . . 81
4.8.1.2 Scénario 2 . . . 82
4.8.1.3 Scénario 3 . . . 83
4.8.2 Cas de plusieurs personnes . . . 86
4.8.2.1 Scénario 1 . . . 86
4.9 Détection, suivi 3D et reconnaissance faciale des personnes . . . 88
4.9.1 Scénario 1 . . . 88
4.10 Conclusion . . . 90

Annexes **96**

A Le processus de la perception **97**

B La méthode de la fenêtre glissante **99**
B.1 La méthode de la fenêtre glissante . . . 99
B.2 Algorithme de détection par fenêtre glissante . . . 100

C Le gradient d'une image **101**
C.1 Les applications de l'image gradient . . . 103

D Histogramme d'intensité **104**

E Machine à vecteurs de support SVM **106**
E.1 Principe général . . . 106
E.2 Discrimination linéaire et hyperplan séparateur . . . 106
E.3 Marge maximale . . . 107
E.4 Recherche de l'hyperplan optimal . . . 108

F image intégrale **110**

G Algorithme d'apprentissage basé sur Adaboost **112**

H Analyse Discriminante Linéaire (Fisher-Faces) **114**

I Rappels et définitions **117**
I.1 Caractéristique d'une variable aléatoire scalaire . . . 117
I.2 Caractéristique d'une variable aléatoire à plusieurs dimensions . . . 118
I.3 Signal aléatoire (processus stochastique) . . . 119
I.4 Moments d'un signal aléatoire . . . 119
I.5 Bruit blan . . . 119
I.6 Filtre de Kalman . . . 120
I.6.1 Position du problème . . . 120
I.6.2 Filtre de Kalman discret . . . 120

J ROS **123**
J.1 Un système d'exploitation OS pour robot . . . 123
J.2 l'interet d'un OS pour les robots . . . 123
J.3 Les distributions de ROS . . . 124
J.4 ROS support . . . 125
J.5 Les caractéristiques de ROS . . . 125
J.6 Feuille de route pour le développement de ROS . . . 127
J.6.1 Niveau communautaire . . . 127

K Description de la Kinect **128**
K.1 Les composants de la Kinect . . . 129
K.1.1 La camera RGB (Red Green Blue) . . . 129
K.1.2 3D depth sensor . . . 130
K.1.3 Le réseau des microphones . . . 132

L Algorithme « Mean Shift » **133**
L.0.4 Mode . . . 133
L.1 Principe de l'algorithme . . . 134
L.2 L'expression du gradient pour les noyaux simples . . . 134
L.2.1 Expression pour les noyaux composés . . . 135

Liste des tableaux

3.1 Les biométries couramment utilisées. 50
3.2 Etapes de prédiction 63
3.3 Etape de correction 63

Table des figures

1 Organisation du mémoire. 19

1.1 Voiture sans conducteur de Google; Drone prévu pour être utilisé par la police de Paris. 21
1.2 Atlas, robot marcheur conçu par une équipe de MIT, etAUV Abyss, conçu par une équipe de GEOMAR. 22
1.3 Exemple de plate-forme différentielle. Pioneer 2 DX de la société MobileRobots et Urban Robot de la société iRobot. 22
1.4 Exemple de plate-forme omnidirectionnelle à roues orientables. 23
1.5 Exemple de plate-forme omnidirectionnelle à roues suédoises. 23
1.6 Exemple de plate-forme non holonome de type Ackerman. 24
1.7 Exemple de plate-forme à pattes. 25
1.8 le prototype d'une chaîne de traitement dans un cadre de détection et de suivi de personnes. 28

2.1 La perception en robotique. 30
2.2 Le schéma de la commande en robotique. 31
2.3 Les étapes de détection $2D$. 33
2.4 Exemple sur les fenêtres originales utilisées par Dalal [16]. 34
2.5 Exemple d'une cellule de 8x8 tirée de [16]. 34
2.6 Exemple d'histogramme utilisé. 35
2.7 Le chevauchement des blocs choisi par Dalal et Trigs [15] 35
2.8 Le processus de calcul des histogrammes. 36
2.9 À gauche : le bloque R-HOG. À droite : le bloque C-HOG [16]. 36
2.10 Le processus de HOG [16]. 37
2.11 caractéristiques pseudo-haar à seulement deux caractéristiques 40
2.12 Caractéristiques pseudo-haar avec différentes orientations. 40
2.13 Illustration de l'architecture de la cascade : les fenêtres sont traitées sé- quentiellement par les classifieurs, et rejetées immédiatement si la réponse est négative (F). 42
2.14 Organigramme résumant la méthode de détection en $3D$. 42

2.15 Acquisition d'une image de profondeur avec la Kinect. 43
2.16 Technique de la lumière structurée. 43
2.17 Technique de la profondeur par focus. 43
2.18 Technique de la profondeur par stéréo. 44
2.19 A- l'image avec profondeur B- représentation des parties du corps humain C- les positions $3D$ des articulations par pixel. 44
2.20 Etiquetage des parties du corps humain. 45
2.21 Le squelette humain et la nomenclature des principales articulations. 45
2.22 Caractéristique de profondeur de pixel dans l'image. 46
2.23 Forêts de décision randomisés. 47
2.24 Optimisation par la méthode de Mean Shift. 48
2.25 Quelques configurations possibles du squelette humain à partir des articulations. 48

3.1 caractéristiques biométriques : a) ADN, b) Oreille, c) visage, d) visage infrarouge, e) thermo-gramme main, f) veine main, g) Empreintes digitales, h) marche, i) geste, j) iris, k) empreinte de la paume, I) rétine, m) signature, n) voix. 50
3.2 Les expressions faciales. 51
3.3 Effet du l'illumination sur les images de visage. 51
3.4 Classification des principaux algorithmes utilisés en reconnaissance faciale [6]. 52
3.5 Principe de fonctionnement de base et principales tâches d'un système de reconnaissance faciale. 54
3.6 Organigramme des principales tâches en reconnaissance facial. 55
3.7 (a) image RGB, (a) image niveau de gris (c) égalisation d'histogramme. . . 56
3.8 histogramme de l'image avant et après égalisation. 56
3.9 exemple de variation de pose. 57
3.10 phase d'apprentissage dans un processus de reconnaissance faciale utilisant une méthode globale. 58
3.11 Phase de reconnaissance d'un système de reconnaissance faciale. 59
3.12 llustration du principe de séparation optimale des classes par le LDA. Trois distributions 3D sont projetées sur deux sous-espaces 2D décris par les vecteurs W_1 et W_2 . Puisque le LDA essaye de trouver la plus grande séparation parmi les classes, on voit bien que W_1 est ici le vecteur optimal. 60
3.13 Taxonomie des approches de suivi d'objets [51]. 61
3.14 Le cycle du filtre de Kalman discret. 62
3.15 suivi du squelette humain. 64
3.16 Distribution de probabilité d'une partie du cops à l'instant $t = t_1$. 65
3.17 Première convergence vers le centre de masse de la zone choisit (cercle bleu). 65

3.18 Deuxième convergence vers le centre de masse de la zone choisit (cercle bleu). 65
3.19 Convergence vers l'articulation de l'instant $t = t_1$. 65

4.1 Plateforme du CDTA : Robot mobile B21r et caméras Kinect 68
4.2 La Kinect de Microsoft 68
4.3 Exemples de robots compatibles avec *ROS* 70
4.4 Architecture prototype. 73
4.5 Scénario 1: scènes a et b. 74
4.6 Scénario 1: la scène du point de vue robot. 74
4.7 A gauche la scène réel. A droit la scène du point de vue robot. 75
4.8 la scène réelle. 75
4.9 la scène du point de vue robot. 76
4.10 la scène du point de vue robot. 76
4.11 A gauche la scène réel. A droit la scène du point de vue robot. 77
4.12 Scénario 1: la scène *a*. 78
4.13 Changement de l'orientations de la tête. 78
4.14 Changement d'expressions du visage. 78
4.15 Changement d'accessoires. 78
4.16 Changement de la luminosité. 79
4.17 Scénario 1: La scène a. 79
4.18 Scénario 1: la scène du point de vue robot. 80
4.19 Modèle des 15 articulations utilisées dans les tests. 80
4.20 Représentation 3D des articulations dans Rviz en temps réel à l'instant t_1. . 81
4.21 Représentation 3D des articulations dans Rviz en temps réel à l'instant t_2. . 81
4.22 Repère de la Kinect. 82
4.23 Position 3D de la tête en temps réel de la personne. 82
4.24 A gauche image couleur (RGB), à droite image profondeur (carte de profondeur), à un instant t_1. 83
4.25 A gauche image couleur (RGB), à droite silhouette de la personne dans l'image de profondeur (carte de profondeur), instant t_2. 83
4.26 A gauche image couleur (RGB), à droite silhouette et squelette de la personne dans l'image de profondeur, à un instant t_3. 84
4.27 A gauche image couleur (RGB), à droite silhouette et squelette de la personne dans l'image de profondeur à un instant t_4. 84
4.28 A gauche image couleur (RGB), à droite silhouette et squelette de la personne dans l'image de profondeur à un instant t_5. 84
4.29 A gauche image couleur (RGB), à droite silhouette et squelette de la personne dans l'image de profondeur à un instant t_6. 85

4.30 A gauche image couleur (RGB), à droite image profondeur (carte de profondeur), à un instant t_1. 86
4.31 A gauche image couleur (RGB), à droite silhouette et squelette de plusieurs personnes dans l'image de profondeur, à un instant t_2. 86
4.32 A gauche image couleur (RGB), à droite silhouette et squelette de plusieurs personnes dans l'image de profondeur, à un instant t_3. 87
4.33 A gauche image couleur (RGB), à droite silhouette et squelette de plusieurs personnes dans l'image de profondeur, à un instant t_4. 87
4.34 Identification d'une personne comme étant inconnue. 89
4.35 Identification d'une personne comme étant connue (identifiée par son prénom). 89
4.36 Dissimulation du visage de la personne par un objet. 89

A.1 Le processus de la perception d'après [3] 97

B.1 détection par la méthode de la fenêtre glissante, les fenêtres en jaune contiennent des personne par contre celle en rouge ne contiens pas d'être humain. 99

C.1 Exemple de calcul du gradient 102
C.2 Exemple de calcul du gradient 102
C.3 représentation du vecteur gradient calculé 102
C.4 A gauche, une image d'intensité d'un chat. Au centre, une image gradient dans la direction x. A droite, une image gradient dans la direction y. 103
C.5 Changement de luminosité de l'image. 103

D.1 A gauche : l'image simple. A droite : histogramme d'intensité correspondant. 104

E.1 A gauche : une infinité d'hyperplans séparateurs. A droite :L'hyperplan optimal (en rouge) avec la marge maximale. 107

F.1 Calcul de l'image intégral 110
F.2 Illustration du calcul de la somme de la luminance des pixels 111

H.1 Passage d'une image vers un vecteur dans un espace vectoriel de grande dimension. Les coefficients a_{ij} représentent les valeurs des pixels en niveau de gris, codés de 0 à 255. 114

J.1 Montre comment les chercheurs en robotique réinventent la roue. 124
J.2 ROS indigo igloo 125

K.1 La Kinect de Microsoft 128
K.2 Un petit récapitulatif des technologies embarquées dans la Kinect. 129
K.3 les trois couleurs primaires de la lumière sont le rouge, le vert et le bleu. . . 130

K.4 Exemplaire d'un capteur photographique de type CMOS. 130
K.5 Fonctionnement d'un capteur photographique de type CMOS pour les appareils photos numériques et les cameras couleur. 130
K.6 Exemplaire d'un capteur photographique de type CMOS. 131
K.7 Une scène réelle (à gauche) et l'image recueillie par la camera Kinect (à droite) 131
K.8 Schéma résume la technologie Kinect 132

L.1 les modes d'une densité de probabilité. 133

Liste des abréviations

ROS	Robot Operating System
SLAM	Simultaneous Localization And Mapping
HOG	Histograms of Oriented Gradients
SVM	Support Vector machine
CDTA	Centre de Développement des Technologies Avancées

Introduction générale

La robotique est un très bon exemple de domaine pluridisciplinaire qui implique de nombreuses disciplines telles que l'automatique, la mécanique, la mécatronique, l'électronique, l'informatique ou l'intelligence artificielle.
Actuellement les robots investissent les environnements humains, on les trouve dans les usines, les champs, l'espace, les jardins, les salons et au fonds des mers. Ils ont une importance économique grandissante, et d'aucuns prédisent qu'ils seront au 21e siècle ce que la voiture fut au 20e siècle. En outre, ils n'ont pas seulement pénétré le monde industriel, ils sont aussi entrain de pénétrer notre vie quotidienne et notre culture, et certains d'entre eux participent au renouvellement de la vision que nous avons de nous-même.
Ce déploiement des robots dans l'entourage des humains revient à ses différentes applications, comme le nettoyage et l'entretien, l'assistance à une personne handicapée, le guidage lors d'un passage par un aéroport, etc. On parle alors, de façons générales, de "robotique de service".
Un tel cadre d'utilisation requiert une forte interaction du robot avec l'être humain ce qui représente un enjeu sociétal majeur, pour que le robot puisse effectuer ses taches d'une façon correcte tout en assurant la sécurité des personnes, il doit avoir un minimum d'autonomie avec des capacités de perception et d'information sur son environnement. La perception apparait ainsi comme la composante la plus essentielle dans l'autonomie d'un robot étant donné que les décisions et les actions qu'il peut prendre en dépendent directement.

Problématique

Dans le cadre de la perception en robotique, particulièrement dans la robotique de service, la détection et le suivi de l'être humain, est un des problèmes les plus importantes. En général, on exige que la détection et le suivi soient robustes et rapides. Ce qui est loin d'être facile à réaliser (et atteindre).
En effet, il est essentiel de détecter et d'anticiper le mouvement des personnes dans l'environnement afin de permettre au robot de raisonner et de planifier correctement ses actions en fonction de ce qui l'entoure.

Le travail présenté dans ce mémoire s'inscrit dans le cadre générale des travaux de recherche menés au *CDTA (Centre de Développement des Technologies Avancées)* visant à doter la plateforme robotique expérimentale *"le Robot B21r"* de modules lui assurant une navigation autonome complète dans un environnement dynamique.
Nous nous intéressons tout particulièrement à l'aspect compréhension de scène à travers la détection et le suivi des individus/visages.

Organisation du mémoire

19Ce mémoire s'étend sur quatre chapitres encadrés par une introduction générale et une conclusion générale. Dans le premier chapitre, on s'intéressera au contexte général dans lequel s'insère notre travail en présentant les notions de base de la robotique mobile tout en présentant la navigation autonome et son lien avec la détection et le suivi de l'être humain.
Le second chapitre expose un état de l'art des différents algorithmes proposés pour la résolution du problème de la détection du corps et du visage de l'être humain en $2D$ et en $3D$.
Le troisième chapitre a pour but de présenter les techniques utilisées pour la reconnaissance et le suivi de l'être humain.
Le dernier chapitre présente les résultats de l'implantation de quelques techniques de détection et de suivi sur la plateforme de développement *Br21* du *CDTA*.
Enfin, en conclusion, on donnera une synthèse du travail effectué et on résumera les principaux résultats obtenus ainsi que les perspectives envisagées.

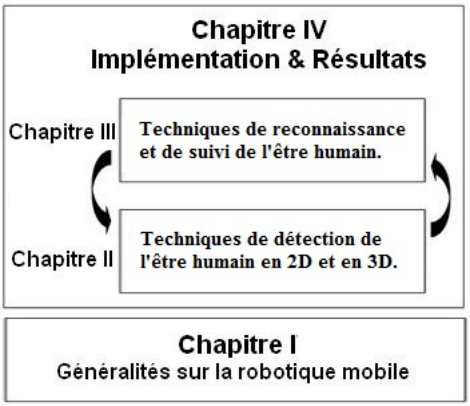

Figure 1: Organisation du mémoire.

Chapitre 1

Généralité sur la robotique mobile

En fonction du domaine d'origine des auteurs, il existe diverses définitions du terme robot, mais elles tournent en général autour de celle-ci : *Un robot est une machine équipée de capacités de perception, de décision et d'action qui lui permettent d'agir de manière autonome dans son environnement en fonction de la perception qu'il en a.*
Depuis quelques années, un intérêt croissant est porté, au sein de la communauté robotique, au développement des systèmes intelligents autonomes dans le cadre de la robotique mobile. Un tel intérêt peut être perçu comme une conséquence logique à l'apparition des applications potentiels des machines intelligentes (industriels, services, militaires, manutention ou encore de l'aide à la mobilité des personnes âgées ou handicapées, ... etc).
Dans ce chapitre, nous donnerons la définition du robot mobile avec une classification des différents types, nous exposons aussi le domaine de la robotique de service avec une petite introduction à la détection et suivi de l'être humain.

1.1 Les Robots mobiles

Pour commencer, il nous faut expliciter la notion de robot mobile. De manière générale, on regroupe sous l'appellation robot mobile l'ensemble des robots à bases mobiles, par opposition notamment aux robots manipulateurs qui eux ont une base fixe [34].

Définition 1. *Un robot mobile peut se définir comme un système mécanique capable de se déplacer dans son environnement de façon autonome. Pour cela, un robot doit être équipé de capteurs pour percevoir son environnement, d'actionneurs pour se mouvoir et d'une intelligence (algorithme ou régulateur) pour calculer, à partir des données recueillis par les capteurs, les commandes à envoyer aux actionneurs afin de réaliser une mission donnée [31].*

Du fait de cette mobilité les robot mobiles occupent une place particulière en robotique. Comme les robots manipulateurs, ils sont destinés à assister l'homme dans les tâches pénibles

(transport de charges lourdes), monotones ou en ambiance hostile (nucléaire, marine, spatiale, lutte contre l'incendie, surveillance ...etc.).
L'aspect particulier de la mobilité impose une complexité technologique et méthodologique qui s'ajoute en général aux problèmes rencontrés par les robots manipulateurs. La résolution de ces dernières passes par l'emploi de toutes les ressources disponibles tant au niveau technologique (capteurs, motricité, énergie) qu'à celui du traitement des informations par l'utilisation des techniques de l'intelligence artificielle ou de processeurs particuliers (vectoriels, cellulaires).
L'autonomie du robot mobile est une faculté qui lui permet de s'adapter ou de prendre une décision dans le but de réaliser une tâche malgré un manque d'informations préliminaires ou éventuellement erronées. Dans d'autres cas d'utilisation, comme celui des véhicules d'exploration de planètes, l'autonomie est un point fondamental puisque la télécommande est alors impossible par le fait de la durée du temps de transmission des informations [35].
On peut regrouper les problèmes clés de la robotique mobile autonome en trois points particuliers qui sont selon [39] :

- La modélisation et la commande du robot mobile ;
- la perception et l'interaction avec l'environnement local
- l'exploration et la représentation de l'environnement local

La figures 1.1 et la figure 1.2 représentent quelques exemples de robots mobiles :

Figure 1.1: Voiture sans conducteur de Google; Drone prévu pour être utilisé par la police de Paris.

1.2 Les types de plateforme mobile :

Nous présentons rapidement les différents types de bases mobiles utilisées en robotique, en nous focalisant sur les plateformes mobiles terrestres pour le milieu intérieur.

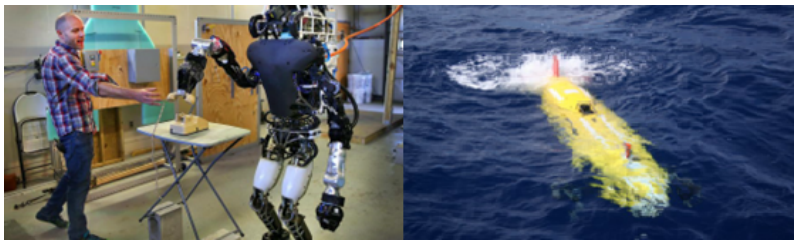

Figure 1.2: Atlas, robot marcheur conçu par une équipe de MIT, etAUV Abyss, conçu par une équipe de GEOMAR.

1.2.1 Les plates-formes holonomes

En robotique, une plateforme est dite holonome lorsque que le nombre de degrés de libertés contrôlables est égal au nombre total de degrés de liberté. Pour un robot se déplaçant sur un plan, il y a 3 degrés de liberté (deux translations et une rotation). A partir d'une position donnée, une plateforme holonome devra donc pouvoir se déplacer en avant, sur le côté et tourner sur elle-même. Cette capacité permet de contrôler très simplement le robot car tous les déplacements imaginables sont réalisables, ce qui simplifie le problème de la planification de trajectoire. De nombreuses plateformes simples ne sont pas holonomes. C'est par exemple le cas des voitures, ce qui oblige à manœuvrer pour réaliser certaines trajectoires. Par exemple, il est nécessaire de faire un créneau pour réaliser un déplacement latéral. Ces contraintes devront donc être prises en compte lors de la planification de trajectoires.

1.2.2 Les plates-formes différentielles

Une des configurations les plus utilisées pour les robots mobiles d'intérieur est la configuration différentielle qui comporte deux roues commandées indépendamment. Une ou plusieurs roues folles sont ajoutées à l'avant ou à l'arrière du robot pour assurer sa stabilité (Figure 1.3). Cette plate-forme est très simple à commander, puisqu'il suffit de spécifier les vitesses des deux roues pour réaliser la commande. Il permet de plus au robot de tourner sur place. Cette possibilité permet de traiter dans certains cas le robot comme un robot holonome.

Figure 1.3: Exemple de plate-forme différentielle. Pioneer 2 DX de la société MobileRobots et Urban Robot de la société iRobot.

1.2.3 Les plates-formes omnidirectionnelles

Les plates-formes omnidirectionnelles permettent de découpler de manière plus nette le contrôle de la rotation et de la translation d'un robot et sont donc quasiment holonomes. Il existe différents types de plateformes omnidirectionnelles. Le premier utilise trois ou quatre roues qui tournent à la même vitesse pour fournir une translation et un mécanisme qui permet d'orienter simultanément ces roues dans la direction du déplacement souhaitée (Figure 1.4).

Le corps du robot lui-même n'effectue pas de rotation mais uniquement des translations. Ce système permet un contrôle très simple et relativement rapide car les changements de direction ne concernent que les roues et peuvent donc se faire très vite. Par contre ces plates-formes sont relativement limitées en capacité de franchissement et requièrent un sol très plan.

Une deuxième catégorie de plateformes utilise des roues dites "suédoises", qui n'offrent pas de résistance au déplacement latéral (Figure 1.5). La plateforme comporte trois roues dont les axes sont fixes. Les déplacements dans toutes les directions et en rotation sont obtenus en faisant varier individuellement les vitesses des roues. La plateforme tourne sur place lorsque les trois roues tournent dans le même sens, à la même vitesse. Lorsqu'une roue est fixe, et que les deux autres tournent en sens opposé, la plateforme avance en direction de la roue fixe. Différentes combinaisons de vitesses permettent d'obtenir des déplacements quelconques.

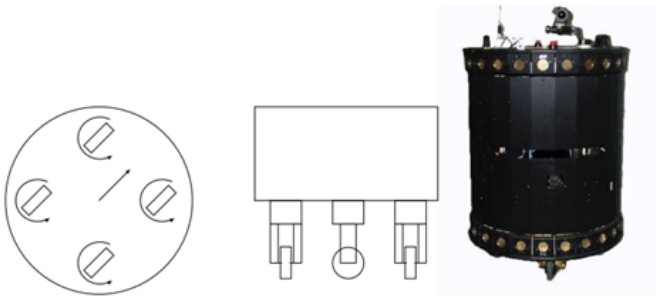

Figure 1.4: Exemple de plate-forme omnidirectionnelle à roues orientables.

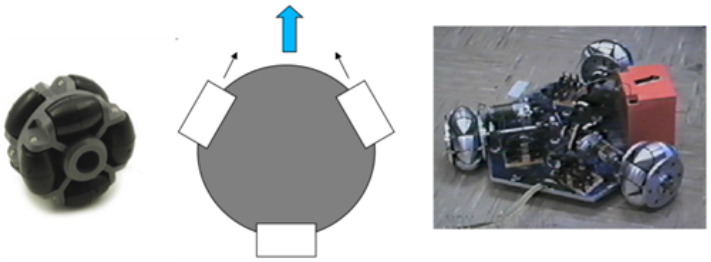

Figure 1.5: Exemple de plate-forme omnidirectionnelle à roues suédoises.

1.2.4 Les plates-formes non holonomes

Des plates-formes non holonomes, telles que les voitures, sont également utilisées en robotique mobile (Figure 1.6). C'est plus particulièrement le cas dans le domaine des véhicules intelligents. Ces plates-formes sont toutefois plus difficiles à commander car elles ne peuvent pas tourner sur place et doivent manœuvrer, ce qui peut être difficile dans des environnements encombrés. La commande de ces plates-formes pour réaliser un déplacement particulier est un problème à part entière.

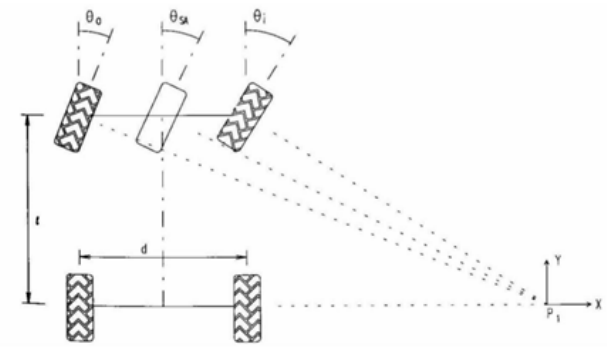

Figure 1.6: Exemple de plate-forme non holonome de type Ackerman.

1.2.5 Les plates-formes à pattes

Des plates-formes à deux, quatre ou six pattes peuvent également être utilisées. Elle ont l'avantage théorique de pouvoir se déplacer sur des terrains assez complexes, même si en pratique la plupart de ces plates-formes ne fonctionnent que sur des sols plans. Les plates-formes à six pattes sont relativement pratiques car le robot peut être en équilibre permanent sur au moins 3 pattes, ce qui facilite le contrôle. Les plates-formes à deux ou quatre pattes sont plus complexes à commander et le simple contrôle de la stabilité et d'une allure de marche correcte reste aujourd'hui difficile, ce qui les rend en général relativement lentes. L'odométrie de ce type de plates-formes est généralement d'assez faible qualité. Ces différents facteurs font que ces plates-formes sont rarement utilisées quand l'application visée a un besoin précis de positionnement et de navigation. De telles plates-formes commencent cependant à apparaître à relativement grande échelle (par exemple le robot Nao de Aldebarran Robotics,qui est au milieu de la figure 1.7).

1.3 La robotique de service

Définition 2. *Selon la Fédération Internationale de la Robotique (International Federation of Robotics), un robot de service est un robot qui fonctionne d'une façon semi ou bien*

Figure 1.7: Exemple de plate-forme à pattes.

entièrement autonome afin de fournir des services utiles pour le bien-être des humains, à l'exclusion des applications de l'automatisme dans l'industrie.

Dans la robotique de service on trouve deux catégories de robot, les robots de service personnels, utilisés pour une tâche non-commerciale, généralement dans les maisons. Par exemple les robots serviteurs domestiques, les fauteuils roulants automatisés et Les robots de compagnie pour personnes âgées. La deuxième catégorie celle des robots de service professionnels utilisés pour une tâche commerciale, généralement exploités par un opérateur bien formé, par exemples les robots de nettoyage pour les lieux publics, les robots de livraison dans les bureaux ou les hôpitaux, les robots guides ou bien suiveurs dans les musées et les aéroports, . . . etc.
Mais derrière l'immense diversité du monde des services, on retrouve une idée simple : ce qui caractérise un service, c'est l'interaction avec le monde humain, par opposition à l'industrie qui est tournée vers la matière.
Cette interaction avec l'être humain, exige que le robot soit capable de détecter les personnes afin de d'achever sa mission d'une manière correcte et en assurant la sécurité de son maitre.

1.4 La navigation autonome des robots mobiles

L'objectif de la navigation autonome est de développer des techniques permettant au robot mobile de se déplacer sans l'intervention humaine dans un environnement inconnu et en présence de l'être humain. Cela consiste en l'exécution d'un certain nombre d'actions élémentaires (déplacement, manipulation d'objet...) qui nécessite une localisation précise, ainsi qu'une détection robuste.
La détection de l'être humain est une étape essentielle et primordiale pour une multitude de fonctions en robotique mobile. Elle est le résultat de la résolution d'un problème plus général : la perception. En effet, un robot observe le monde externe à l'aide de capteurs et construit un modèle interne de l'environnement extérieur. Il met à jour continuellement ce modèle en utilisant les dernière données des capteurs [36].
Dans ce cadre, la perception peut être divisée en deux étapes : la première partie, appelée

SLAM (Simultaneous Localization And Mapping) s'intéresse à la construction d'une carte de l'environnement extérieur et à la localisation du robot lui-même dans cette carte, et la deuxième partie traite la détection et le suivi des objets mobiles qui s'y trouvent (dans notre cas l'être humain).
Ce dernier est un point clé pour le fonctionnement d'un robot autonome. En effet, il est vital de détecter et d'anticiper le mouvement des objets présents dans l'environnement ainsi que leurs propriétés cinématiques afin de permettre au robot de raisonner et de planifier correctement des actions en fonction de ce qui l'entoure. De ce fait, l'utilité d'un module de détection des objets et particulièrement l'être humain s'avère incontournable.[25]

1.5 La détection de l'être humain

La détection de l'homme en vision et robotique est l'opération qui consiste à segmenter les pixels de l'image acquise en deux classes : ceux appartenant à la classe personne et ceux à la classe non-personne (ou à l'arrière-plan). C'est une tâche difficile. Il y a deux raisons à cela, d'abord l'extrême variabilité causée par l'articulation, du point de vue de l'apparence humain, ensuite l'importante quantité d'informations contenues dans une image. En effet, l'apparence de l'homme dans une image est très variable. Elle va dépendre de la position et de l'orientation de la personne par rapport à la camera. Cela va jouer sur sa taille, sa forme et sa texture. Il y a de plus la perte d'informations due à la projection du 3D vers le 2D, qui peut entraîner certaines ambiguïtés.
L'intensité d'un pixel dépend aussi de l'éclairage de la scène : de sa position, de sa couleur, de son intensité. De son interaction avec le restant de la scène : ombres, nouvelles sources lumineuses induites par réflexion. L'autre difficulté de la détection de l'homme dans une image est l'importante quantité de données à traiter et la puissance de calcul demander car l'image de l'homme peut être composée de plusieurs centaines, voire de milliers de pixels, et chacun d'eux peut contenir une information importante dans le processus de détection.
On notera que les premières applications qui requérant un processus de détection de l'homme sont celles axées sur les visages. Dans ce cadre précis, de très nombreuses recherches ont été menées depuis les années 90. En effet, pour cette tache précise, les techniques sont très poussées et font très souvent appel à un compromis entre robustesse et rapidité comme le montre [49].

1.6 Le suivi de l'être humain

Le suivi des personnes en robotique a pour but essentiel d'estimer et prédire l'évolution de l'état de l'homme dans son enivrement suivant le temps. L'état peut être par exemple composé de ses caractéristiques cinématiques. L'estimations et la prédiction est réalisée à partir

de mesures (bruitées) sur une séquence d'image fournit par les capteurs (laser, caméra...). Donc le suivi de la personne revient à optimiser l'estimation dans son mouvement pour qu'elle corresponde au mieux aux mesures réalisées dans l'image.
Le suivi de personne fait appel à :

- *La prédiction :* calcul de la position la plus probable de la personne dans l'image courante.
- *La mesure :* consiste à estimer une ou plusieurs caractéristiques dans les images de la séquence, à la position prédite (ou autour).
- *L'observation :* consiste à déterminer la position de la personne à partir de la mesure réalisée précédemment.
- *La validation :* examine la validité de la position estimée de la personne.
- *L'estimation :* conclue le processus de suivi en fournissant une estimation de l'état de la personne.

Le suivi des personnes a un large éventail d'application, y compris la robotique, l'image et l'indexation vidéo, surveillance et la sécurité automobile. Par conséquent le suivi des gens a été étudié de manière intensive avec un rythme rapide de l'innovation au cours de cette dernière décennie.

La Figure 1.8 illustre le prototype d'une chaîne de traitement dans un tel cadre applicatif. Le premier bloc de traitement consiste en la détection et la segmentation des personnes au premier plan, afin de les isoler du reste de l'image pour des analyses ultérieures. Quand un individu est détecté, le processus de suivi est alors enclenché et les informations sur sa position, la vitesse et la direction de ses mouvements sont mises à jour tout au long des séquences d'images. Puis, l'ensemble du chemin parcouru par cet individu est extrait par une étape d'analyse de trajectoire . A partir des informations de position, de vitesse de déplacement, de direction et une analyse d'actions, des caractéristiques permettant une description sémantique pourront être extraites. On pourra alors effectuer des analyses de plus haut niveau portant sur la reconnaissance de comportements et d'activités.

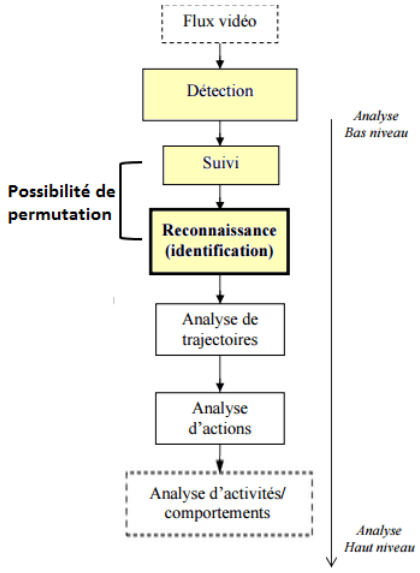

Figure 1.8: le prototype d'une chaîne de traitement dans un cadre de détection et de suivi de personnes.

1.7 Conclusion

A travers ce premier chapitre, nous avons présenté de façon générale les robots mobiles autonomes et la robotique de service, les différentes plateformes robotiques ainsi que la problématique de la détection et suivi de l'être humain. Nous avons mentionné la relation qui existe entre la perception, la navigation autonome et la détection et suivi et de l'être humain.

Ce premier chapitre sera compléter par une étude plus détaillées du problème de détection et de suivi des personnes, suivie des différentes approches existantes pour la résolution de ce problème.

Chapitre 2

Techniques de détection de l'être humain

2.1 Introduction

Dans la littérature de la robotique mobile, on retrouve plusieurs méthodes permettant d'assurer la détection des êtres humains et le suivi de leur mouvement. Etant donné l'importance de ce module dans la tache de perception, notamment en robotique de service ou la détection de personne est une affaire primordiale, les axes de recherche dans ce domaine se sont multipliés afin de répondre à toutes les problématiques posées et d'apporter une constante amélioration aux performances de ce module.
Dans ce chapitre, nous allons présenter deux méthode de détection $2D$ du corps et du visage humain et une autre méthode de détection du squelette humain en $3D$, ainsi que les notions théoriques nécessaire à leur compréhension et à leur implémentation. Cela va de la présentation des notions et des outils de base de la détection du corps et du visage humain en jusqu'à la présentation détaillée de la méthodologie de la détection du squelette en $3D$.

2.2 La perception dans la robotique mobile

Définition 3. *Littéralement la perception (du latin perceptio, percipio) est le processus de compréhension de l'environnement par l'organisation et l'interprétation des données qui proviennent des capteurs [36].*

Définition 4. *La perception en robotique désigne la capacité à obtenir des informations sur l'environnement qui entoure le robot en utilisant les capteurs, un prétraitement sur les données de ces capteurs et la mise en forme des informations utiles au robot pour agir et réagir dans le monde qui l'entoure.*

Concrètement, ceci est réalisé en obtenant les informations fondamentales suivantes :

- La position des objets par rapport aux autres objets. Ces information sont appelées carte de l'environnement (où map) ;
- La position du robot dans cette carte ;
- Distinction entre les objets statiques et les objets dynamiques ;
- Le type de chaque objet présent dans la carte, aussi appelé classification des informations (personne, voiture, etc.)

La figure 2.1, a pour but d'illustrer l'objectif de la perception en robotique, on peut voir dans cette image la représentation de l'environnement en inférant les données des capteurs. On constate d'après cette figure que le robot a inféré les formes de l'environnement et les objets qui s'y trouvent, détecter les objets mobiles et suivre leurs mouvements ce qui est représenté par les lignes bleues, vertes et violées. Une telle représentation est généralement obtenue en fusionnant les données de plusieurs capteurs.

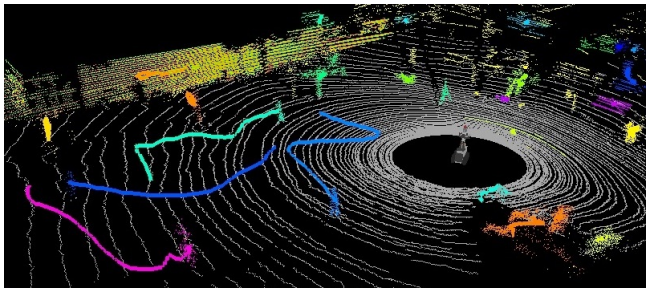

Figure 2.1: La perception en robotique.

La perception est le problème le plus important à résoudre en vue de permettre au robot de fonctionner sans intervention humaine. La perception, telle que définie plus haut, requiert la résolution des cinq sous-problèmes suivants :

- *Mapping* ; permet d'obtenir les relations géométrique entre les objets statiques présents dans l'environnement. La résolution de ce problème est nécessaire pour la construction d'une carte en temps réel en utilisant les données des capteurs.
- *Localisation* : définie comme étant le processus permettant d'obtenir la relation entre le robot mobile autonome et les objets statiques. Un robot mobile doit connaitre sa position dans la carte de l'environnement.
- *Détection des objets mobiles* : permet au robot de distinguer entre les objets statiques et les objets dynamiques.

- *Suivi des objets mobiles* : défini comme étant le processus permettant d'établir les relations géométrique et temporelles entre le robot autonome, les objets mobiles et l'environnement statique.

- *Classification des objets* : la résolution de ce problème permet de connaitre le type de chaque objet, notamment les objets dynamiques.

Ces problèmes sont regroupés dans la littérature en deux principaux problèmes : SLAM (Simultaneous Localization And Mapping) et la détection des objets [25].
Afin de bien comprendre le processus de la perception vous trouverez dans l'annexe A une explication détaillée de ce dernier.

La figure 2.2 représente le schéma standard de l'asservissent (qu'elle soit basée sur des capteurs visuelle ou bien d'autres capteurs qui donnent des informations sur l'environnement du robot), le travail présenté dans ce mémoire est concentré dans la partie de la perception (l'observation ou bien la mesure par analogie au schéma de commande dans le cas générale).

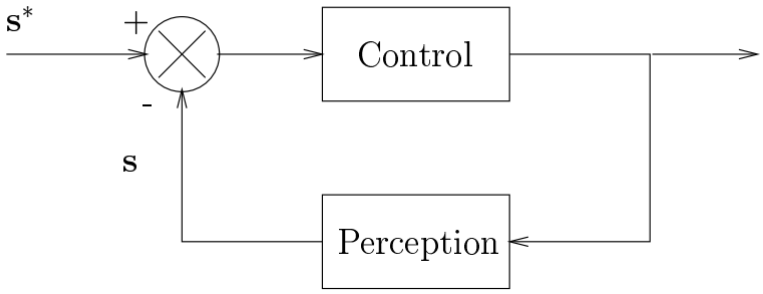

Figure 2.2: Le schéma de la commande en robotique.

2.3 Les capteurs utilisés pour la détection

Les systèmes de navigation autonome, actuellement, privilégient des capteurs fournissant des informations sur les caractéristiques de l'environnement, comme les systèmes de stéréovision, les capteurs de profondeur, les télémètres laser, les radars, les sonars et d'autres modalités sensorielles. Chaque capteur a des avantages et des inconvénients propres ; dans le contexte du développement de robots de service destinés au grand public, comme les robots guides la tendance actuelle est de privilégier les méthodes de perception fondées sur des capteurs compacts (donc faciles à intégrer dans la structure mécanique du robot), sans danger pour l'utilisateur et bas coût. Ce sont les capteurs visuels qui satisfont le mieux ces contraintes, même si ils ne sont pas toujours la solution la plus performante.
On peut distinguer deux familles de capteur, les capteurs actifs et les capteurs passifs

2.3.1 Les capteurs actifs

Un capteur est dit "actif" s'il produit une énergie envoyée sur la scène; les obstacles sont détectés par l'énergie réfléchie par leur surface. Un capteur actif possède donc à la fois un émetteur et un récepteur. En générale ces capteurs sont robustes aux variations des conditions atmosphériques (pluie, neige) ou à des conditions dégradées d'illumination. On peut citer quelques capteurs actifs comme les télémètres laser et Le radar (RAdio Detection And Ranging).

2.3.2 Les capteurs passifs

Un capteur passif ne fait que recevoir l'énergie naturellement réfléchie par les objets présents dans la scène. Plusieurs capteurs passifs sont utilisés pour la robotique ou pour la surveillance : les capteurs inertiels (détection de vibrations, de mouvement), les capteurs haptiques (contact), mais les plus connus sont les capteurs visuels, donc des caméras. En tant que capteur passif, une caméra mesure l'intensité lumineuse émise par l'environnement sous la forme d'une image digitalisée, ou d'un tableau de pixels. On peut citer plusieurs exemples sur les capteurs visuels comme les caméras IR (infra-Rouge), la stéréovision et les systèmes mono-caméras.
Un capteur visuel devient actif dès lors qu'on lui associe un projecteur ou illuminateur qui envoie sur la scène un motif lumineux (profil laser, matrice de points laser, ... etc.) comme la caméra Kinect.

L. Matthies et. al. [32] indiquent que le choix entre radar, télémétrie laser ou vision pour détecter des objets dans une scène depuis un robot, dépend de la fréquence d'apparition des objets dans l'environnement de navigation et de la vitesse du robot qui porte les capteurs. Si la fréquence est faible et les vitesses aussi, il est plus avantageux d'exploiter la vision. Par exemple, les auteurs précisent que l'utilisation d'un système Ladar[1] pour un robot d'exploration planétaire, n'est pas nécessaire vu la très faible fréquence d'apparition d'obstacles et vu sa faible vitesse.

2.4 La détection de l'être humain en 2D

Dans cette partie nous allons présenter deux algorithmes de détection de personne en $2D$. Le premier est HOG *(Histogramme des gradients orientés)* utilisé pour la détection du corps humain, et la deuxième est *Viola et Jones* utilisé pour la détection du visage, les notions du gradient d'une image et l'histogramme des intensités, ainsi que la méthode de la fenêtre glissante sont représentées dans les annexes (C, D, B) respectivement, nous invitons le lecteur

[1]Lidar (également écrit LIDAR ou LADAR) est une technologie d'arpentage qui mesure la distance en éclairant une cible avec une lumière laser.

à voir ces annexes afin de mieux comprendre ce qui suit.
Comme présenté dans [16], un détecteur de l'être humain peut être décomposé en plusieurs étapes, voir figure 2.3.

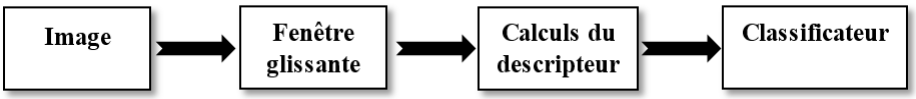

Figure 2.3: Les étapes de détection $2D$.

2.4.1 Méthode des Histogramme des Gradients Orientés

Le détecteur *HOG (Histogram of Oriented Gradient)* utilise une fenêtre glissante[1] qui est déplacée autour de l'image. A chaque position de la fenêtre, un descripteur[2] de *HOG* est calculé. Ce descripteur est ensuite passé au *SVM*[1] (un algorithme de classification) qui va le classer comme étant soit «une personne» ou «pas une personne ». Pour reconnaître les personnes à différentes échelles, l'image est sous-échantillonnée à plusieurs tailles. Chacune de ces images sous-échantillonnées est analysée.

2.4.1.1 Le choix de la fenêtre glissante :

Le détecteur de *HOG* utilise une fenêtre de détection qui est de 64 pixels de large par 128 pixels de hauteur. Voici quelques-unes des images originales utilisées pour former le détecteur, recadrées dans la fenêtre de $64 * 128$, voir figure 2.4.

2.4.1.2 Le choix de la cellule

Pour calculer le descripteur *HOG*, nous opérons sur les cellules de $8 * 8$ pixels dans la fenêtre de détection. Ces cellules seront organisées en blocs qui se chevauchent (qui sera décrite après). Voici une version zoomée de l'une des images (figure 2.5), avec une cellule de $8 * 8$

[1]La méthode de la détection par fenêtre glissante se base sur la méthodologie qui consiste à scanner une image sur toutes les positions pertinentes et sur toutes les échelles pour détecter une présence humaine, pour plus de détail vous pouvez voir l'annexe B.

[2]Dans la vision par ordinateur, les descripteurs visuels ou descripteurs d'images sont des descriptions des caractéristiques visuelles du contenu de cette image. Ils décrivent les caractéristiques élémentaires telles que la forme, la couleur, la texture ou le mouvement, entre autres. Les descripteurs visuels sont divisés en deux groupes principaux :

- Le descripteur général d'information : ils contiennent des descripteurs de bas niveau qui donnent une description de la couleur, la forme, les régions et les textures.
- Le descripteur d'image locale : est une sorte de vecteur (soit en type réel ou type binaire) utilisé comme une signature d'une image locale (une partie de l'image globale). Le but de cette représentation est de rendre l'image locale aussi distinctive que possible.

[1]Support Vector Machine, pour plus de détaille sur cet algorithme vous pouvez voir l'annexe D

Figure 2.4: Exemple sur les fenêtres originales utilisées par Dalal [16].

dessinée en rouge, pour donner une idée de la résolution de la taille des cellules et de l'image que nous travaillons avec.

Figure 2.5: Exemple d'une cellule de 8x8 tirée de [16].

2.4.1.3 Le calcul du gradient

La première étape de l'extraction caractéristique *HOG* est le calcul des gradients d'image (voir annexe C). Cela se fait en appliquant pour chaque pixel d'une cellule, le *1D centré* (un masque de dérivation discret) à la fois dans directions horizontale et verticale, ce qui est en réalité, des noyaux de filtre de la forme suivante :

$$[-1 \quad 0 \quad 1]$$

Ensuite, l'amplitude et l'orientation de chaque pixel I (x, y) est calculé par :

$$\mid G(x,y) \mid = \sqrt{G_x(x,y)^2 + (G_y(x,y)^2)} \tag{2.1}$$

$$\theta = \tan^{-1}(\frac{G_x(x,y)}{G_y(x,y)}) \tag{2.2}$$

Où $G_x(x,y)$ et $G_y(x,y)$ sont les valeurs de gradient pour chaque pixel dans la direction horizontal et la direction verticale, respectivement. Pour les images couleur, on prend

seulement le canal avec la plus grande amplitude [1].

2.4.1.4 L'histogramme

Comme chaque cellule contient 64 pixels $(8 * 8)$, nous prenons les 64 vecteurs gradients, et on les met dans un histogramme[2] $9 - bin$. L'histogramme varie de 0 à 180 degrés, donc il y a 20 degrés par bin, on peut voir dans la figure 2.6 un exemple d'histogramme utilisé.

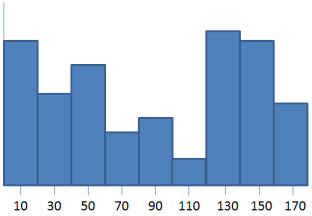

Figure 2.6: Exemple d'histogramme utilisé.

Pour chaque vecteur de gradient, sa contribution à l'histogramme est donnée par l'amplitude du vecteur (donc des gradients forts ont un impact plus important sur l'histogramme). Dalal et Triggs [15] ont partagé la contribution entre les deux bins les plus proches. Ainsi, par exemple, si un vecteur de gradient a un angle de 85 degrés, on va ajouter 1/4 de son amplitude au bin centrée à 70 degrés, et 3/4 de son amplitude au bin centrée à 90.

le but de diviser la contribution est de minimiser le problème des vecteurs gradients qui

Figure 2.7: Le chevauchement des blocs choisi par Dalal et Trigs [15]

sont à droite sur la frontière entre deux bins. Sinon, si un fort gradient était juste sur le bord d'un bin, un léger changement dans l'angle de gradient (qui va pousser le gradient dans le prochain bin) pourrait avoir un fort impact sur l'histogramme. [3]

[1] Il faut noter que: Dalal et Triggs[15] ont utilisé des gradients insignes de telle sorte que les orientations ne variait que de 0 à 180 degrés au lieu de 0 à 360. Et cela en ajoutant $\pi/2$ afin de résoudre le problème des résultats obtenus par la fonction arctan dans un intervalle entre $-\pi/2$ et $\pi/2$.

[2] L'histogramme d'intensité est la mesure du nombre de pixels dans une image pour chaque valeur d'intensité, vous trouverez plus d'explication dans l'annexe D

[3] L'idée de mettre les gradients dans cet histogramme, plutôt que d'utiliser simplement leurs valeurs directement revient au fait que l'histogramme de gradient est une forme de "quantification", où, dans ce cas, nous réduisons de 64 vecteurs avec 2 composants chacun à une chaîne de seulement 9 valeurs (les grandeurs de chaque bin) ; car la compression de la fonction descripteur peut être important pour la performance du classificateur.

2.4.1.5 La normalisation des bloques

Plutôt que de normaliser chaque histogramme individuellement, les cellules sont tout d'abord regroupées en blocs, figure 2.8. En réalité il y a plusieurs types de bloc utilisés avec *HOG* : un bloque rectangulaire *(R-HOG)*, un bloque circulaire *(C-HOG)*, ...etc. Le bloque le plus utilisé est le bloque rectangulaire représenté à gauche de la figure 2.9 . Pour plus de détails vous pouvez voir en [16].

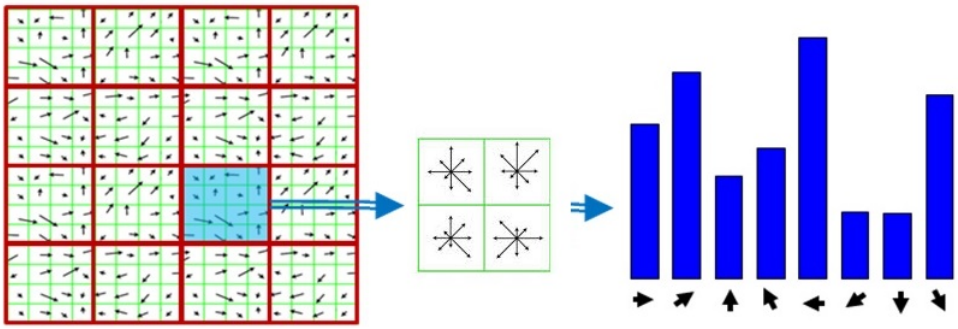

Figure 2.8: Le processus de calcul des histogrammes.

Les blocs utilisés par Dalal et Triggs consistaient en 2 cellules par 2 cellules. Les blocs ont "50 % de chevauchement", qui est le mieux décrit par l'illustration ci-dessous. Ce bloc

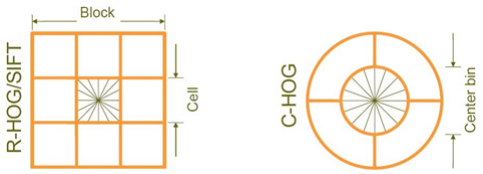

Figure 2.9: À gauche : le bloque R-HOG. À droite : le bloque C-HOG [16].

de normalisation est réalisé en enchaînant les histogrammes des quatre cellules à l'intérieur du bloc dans un vecteur de 36 éléments (4 histogrammes x 9 bacs par histogramme), et de diviser ce vecteur par son amplitude.
L'effet du chevauchement de bloc est que chaque cellule apparaît plusieurs fois dans le descripteur final, mais normalisée par un autre ensemble de cellules voisines. (Spécifiquement, les cellules d'angle apparaissent une fois, les autres cellules de bord apparaissent deux fois, et les cellules intérieures apparaissent quatre fois chacun).
Trois types de normalisation sont explorées:

$$L_2 - norme : f = \frac{v}{\sqrt{\| v \|_2 + \epsilon^2}} \tag{2.3}$$

$$L_1 - norme : f = \frac{v}{(\| v \|_1 + \epsilon)} \tag{2.4}$$

$$L_1 - racine : f = \sqrt{\frac{v}{(\| v \|_1 + \epsilon)}} \tag{2.5}$$

Le vecteur non normalisé contenant tous les histogrammes d'un seul bloc est désigné par v , sa $k-norme$ par $\| v \|$ et ϵ est une constante de faible valeur à fin d'éviter la division par 0, $L_2-norme$, et $L_1-racine$ obtiennent des performances similaires, tandis que $L_1-norme$ obtient de moins bons résultats, mais toutefois bien meilleurs que l'absence de normalisation.

2.4.1.6 La taille finale du descripteur

La fenêtre de détection $64 * 128$ pixels sera divisée en 7 blocs horizontalement et 15 blocs verticalement, pour un total de 105 blocs. Chaque bloc contient 4 cellules avec un histogramme $9-bin$ pour chaque cellule, pour un total de 36 valeurs par bloc. Cela porte la taille du vecteur final à 7 blocs horizontalement et 15 blocs verticalement avec 4 cellules par bloc et $9-bin$ par histogramme, ce qui nous donne le descripteur final *HOG* qui contient 3.780 valeurs. La figure 2.10 résume le processus de l'histogramme des gradients orientés. Le descripteur final est passé vers le classifieur linéaire[1], dans notre cas "SVM" (*Support*

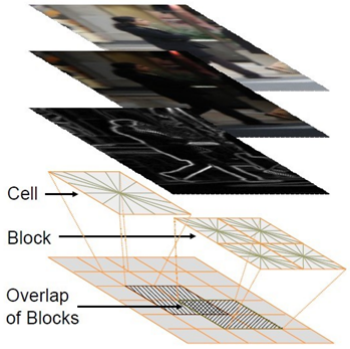

Figure 2.10: Le processus de HOG [16].

Vector Machine), qui va s'occuper de la tâche de la classification, "une personne" ou "pas une personne".

2.4.1.7 Machine à vecteurs de support SVM

Les machines à vecteurs de support ou séparateurs à vaste marge (en anglais Support Vector Machine, SVM) sont un ensemble de techniques d'apprentissage supervisé destinées à résoudre des problèmes de discrimination et de régression. Les SVM sont une généralisation

[1]En apprentissage automatique, le terme de classifieur linéaire représente une famille d'algorithmes de classement statistique. Le rôle d'un classifieur est de classer dans des groupes (des classes) les échantillons qui ont des propriétés similaires, mesurées sur des observations. Un classifieur linéaire est un type particulier de classifieur, qui calcule la décision par combinaison linéaire des échantillons.

des *classifieurs linéaires*. Ils ont été développés dans les années 1990 à partir des considérations théoriques de *Vladimir Vapnik* [45] sur le développement d'une théorie statistique de l'apprentissage.

Principe général Les *SVM* peuvent être utilisés pour résoudre des problèmes de *discrimination*, c'est-à-dire décider à quelle classe appartient un échantillon, ou de *régression*, c'est-à-dire prédire la valeur numérique d'une variable. La résolution de ces deux problèmes passe par la construction d'une fonction h qui, à un vecteur d'entrée x, fait correspondre une sortie y :

$$y = h(x) \tag{2.6}$$

On se limite dans ce mémoire à un problème de discrimination à deux classes (discrimination binaire), c'est-à-dire $y\epsilon\{-1, 1\}$, le vecteur d'entrée x étant dans un espace X muni d'un produit scalaire. On peut prendre par exemple $X = R^N$, ce qui représente dans notre cas le descripteur visuel. Pour plus de détail vous pouvez voir l'annexe E.

2.4.2 Méthode de Viola et Jones pour la détection du visage

Une avancée majeure dans le domaine a été réalisée par les chercheurs *Paul Viola* et *Michael Jones* en 2001 [47]. Ces derniers ont proposé une méthode basée sur l'apparence (Appearance-based methods).

La méthode de *Viola et Jones* est une méthode de détection d'objet dans une image numérique, elle fait partie des toutes premières méthodes capables de détecter efficacement et en temps réel des objets dans une image. La méthode de *Viola et Jones* est l'une des méthodes les plus connues et les plus utilisées, en particulier pour la détection du visage.

En tant que procédé d'apprentissage supervisé, la méthode de *Viola et Jones* nécessite de quelques centaines à plusieurs milliers d'exemples de l'objet que l'on souhaite détecter, pour entraîner un classifieur. Une fois son apprentissage réalisé, ce classifieur est utilisé pour détecter la présence éventuelle de l'objet dans une image en parcourant celle-ci de manière exhaustive, à toutes les positions et dans toutes les tailles possibles.

Considérée comme étant l'une des plus importantes méthodes de détection d'objet, la méthode de *Viola et Jones* est notamment connue pour avoir introduit plusieurs notions reprises ensuite par de nombreux chercheurs en vision par ordinateur, à l'exemple de la notion d'image intégrale ou de la méthode de classification construite comme une cascade de classifieurs boostés [10].

2.4.2.1 Principe

La méthode de *Viola et Jones* consiste à balayer une image à l'aide d'une fenêtre de détection de taille initiale $24 * 24$ et de déterminer si un visage y est présent. Lorsque l'image a été parcourue entièrement, la taille de la fenêtre est augmentée et le balayage recommence, jusqu'à

ce que la fenêtre fasse la taille de l'image. L'augmentation de la taille de la fenêtre se fait par un facteur multiplicatif de 1.25. Le balayage, quant à lui, consiste simplement à décaler la fenêtre d'un pixel. Ce décalage peut être changé afin d'accélérer le processus, mais un décalage d'un pixel assure une précision maximale.

Cette méthode est une approche basée sur l'apparence, qui consiste à parcourir l'ensemble de l'image en calculant un certain nombre de caractéristiques dans des zones rectangulaires qui se chevauchent. Elle a la particularité d'utiliser des caractéristiques très simples mais très nombreuses. La méthode *Viola et Jones* est caractérisée par :

- l'utilisation d'images intégrales qui permettent de calculer plus rapidement les caractéristiques.
- la sélection par boosting des caractéristiques.
- la combinaison en cascade de classifieurs boostés, apportant un net gain de temps d'exécution.

2.4.2.2 Apprentissage du classifieur

Une étape préliminaire et très importante est l'apprentissage du classifieur. Il s'agit d'entraîner le classifieur afin de le sensibiliser à ce que l'on veut détecter, ici des visages. Pour cela, il est mis dans deux situations :
La première où une énorme quantité de cas positifs lui sont présentés et la deuxième où, à l'inverse, une énorme quantité de cas négatifs lui sont présentés. Concrètement, une banque d'images contenant des visages de personnes est passée en revue afin d'entraîner le classifieur. Ensuite, une banque d'images ne contenant pas de visages humains est passée. Dans le cas présent, *Viola et Jones* ont entraîné leur classifieur à l'aide d'une banque d'images du *MIT*[1].
Il en résulte un classifieur sensibles aux visages humain.
Dans l'absolu, on serait en mesure de détecter n'importe quel signe distinctif à partir d'un classifieur entrainé à cela [10].

2.4.2.3 Les caractéristiques

Une caractéristique est une représentation synthétique et informative, calculée à partir des valeurs des pixels. Les caractéristiques utilisées ici sont les caractéristiques *pseudo-haar*. Elle sont calculées par la différence des sommes de pixels de deux ou plusieurs zones rectangulaires adjacentes.
Prenons un exemple de la figure 2.11 qui représente deux zones rectangulaires adjacentes, la première en blanc, la deuxième en noire.

[1] Massachusetts Institute of Technology

Les caractéristique seraient calculées en soustrayant la somme des pixels noirs à la somme des pixels blancs. Les caractéristiques sont calculées à toutes les positions et à toutes les échelles dans une fenêtre de détection de petite taille, typiquement de $24 * 24$ pixels ou de $20 * 15$ pixels. Un très grand nombre de caractéristiques par fenêtre est ainsi généré, *Viola et Jones* donnant l'exemple d'une fenêtre de taille $24 * 24$ qui génère environ 160000 caractéristiques.

La figure 2.11 présente des caractéristiques pseudo-haar *(Haar-like features)* à seulement

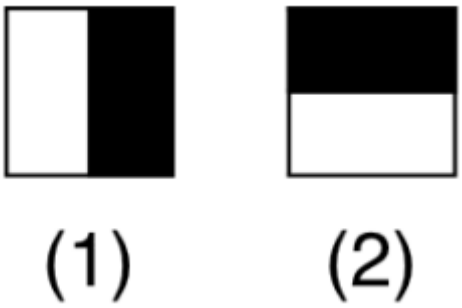

Figure 2.11: caractéristiques pseudo-haar à seulement deux caractéristiques

deux caractéristiques mais il en existe d'autres, allant de 4 à 14, et avec différentes orientations, voir figure 2.12. Malheureusement, le calcul de ces caractéristiques de manière classique coûte cher en terme de ressources processeur, c'est là qu'interviennent les images intégrales.

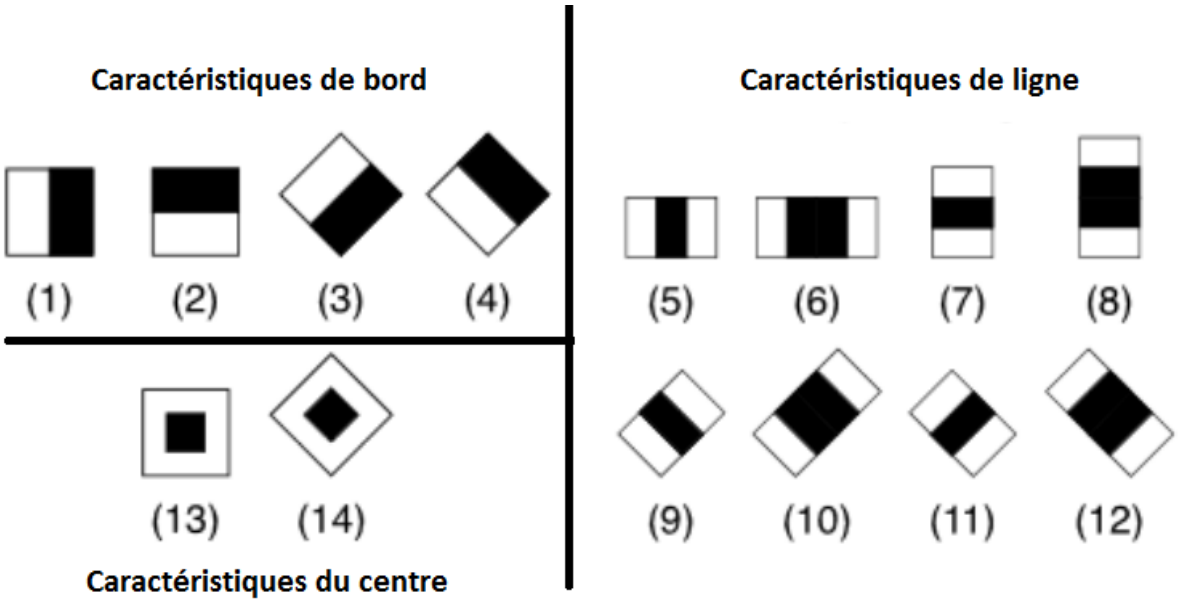

Figure 2.12: Caractéristiques pseudo-haar avec différentes orientations.

2.4.2.4 L'image intégrale

Pour calculer rapidement et efficacement ces caractéristiques sur une image, les auteurs proposent également une nouvelle méthode, qu'ils appellent image intégrale. C'est une représentation sous la forme d'une image, de même taille que l'image d'origine, elle contient en chacun de ses points la somme des pixels situés au-dessus et à gauche du pixel courant. Plus formellement, l'image intégrale i_i au point (x, y) est définie à partir de l'image i par :

$$i_i(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y') \tag{2.7}$$

Pour plus de détaille vous pouvez voir l'annexe F.

2.4.2.5 Algorithme d'apprentissage basé sur Adaboost

Le deuxième élément clé de la méthode de *Viola et Jones* est l'utilisation d'une méthode de boosting[1] afin de sélectionner les meilleures caractéristiques. *Viola et Jones* adaptent ce principe en assimilant une caractéristique à un classifieur faible, en construisant un classifieur faible qui n'utilise qu'une seule caractéristique. L'apprentissage du classifieur faible consiste alors à trouver la valeur seuil de la caractéristique qui permet de mieux séparer les exemples positifs des exemples négatifs. Le classifieur se réduit alors à un couple (caractéristique, seuil), (voir annexe G).

2.4.2.6 Cascade de classifieurs

La méthode de *Viola et Jones* est basée sur une approche par recherche exhaustive sur l'ensemble de l'image, qui teste la présence de l'objet dans une fenêtre à toutes les positions et à plusieurs échelles. Cette approche est cependant extrêmement coûteuse en calcul. L'une des idées-clés de la méthode pour réduire ce coût réside dans l'organisation de l'algorithme de détection en une cascade de classifieurs. Appliqués séquentiellement, ces classifieurs prennent une décision d'acceptation; la fenêtre contient l'objet et l'exemple est alors passé au classifieur suivant, ou de rejet; la fenêtre ne contient pas l'objet et dans ce cas l'exemple est définitivement écarté. L'idée est que l'immense majorité des fenêtres testées étant négatives (c.-à-d. ne contiennent pas l'objet), il est avantageux de pouvoir les rejeter avec le moins possible de calculs. Ici, les classifieurs les plus simples, donc les plus rapides, sont situés au début de la cascade, et rejettent très rapidement la grande majorité des exemples négatifs. Cette structure en cascade peut également s'interpréter comme un arbre de décision dégénéré,

[1] Le boosting est un principe qui consiste à construire un classifieur fort à partir d'une combinaison pondérée de classifieurs faibles, c'est-à-dire, donnant en moyenne une réponse meilleure qu'un tirage aléatoire

puisque chaque nœud ne comporte qu'une seule branche [21].

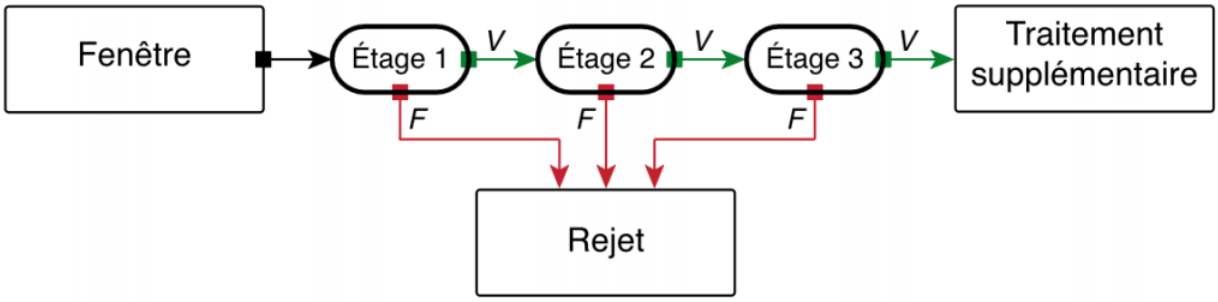

Figure 2.13: Illustration de l'architecture de la cascade : les fenêtres sont traitées sé- quentiellement par les classifieurs, et rejetées immédiatement si la réponse est négative (F).

2.5 Méthodologie de détection du squelette humain en 3D

La détection du squelette humain est une des méthodes les plus utilisées pour reconnaitre, analyser ou reproduire les activités humaines, elle permet d'obtenir une localisation $3D$ des articulations et une représentation du squelette humain.

Dans ce travail, on suppose que le corps est complètement visible (la partie occlusions ne fait pas partie de notre travail). Le capteur *Kinect* est statique par rapport au robot mobile et le mouvement de l'homme le long de l'axe de la caméra est limitée à une plage qui doit être supérieur à 1.5 *metre*. L'organigramme représenté dans la figure 2.14 résume les différentes étapes de cette méthode.

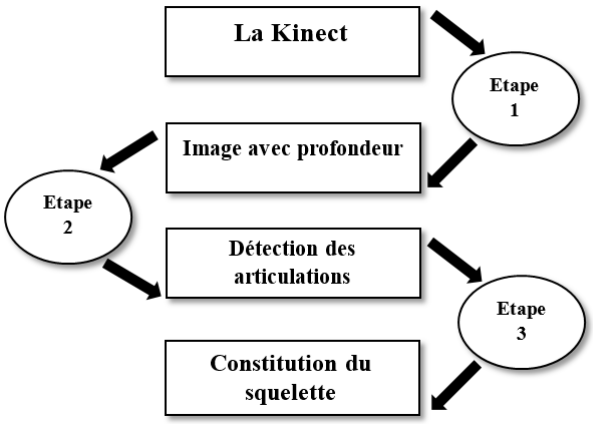

Figure 2.14: Organigramme résumant la méthode de détection en $3D$.

2.5.1 Etape 1

Dans cette étape on va expliquer comment se fait l'acquisition d'une image (carte) de profondeur dans un processus de détection, à partir d'un capteur RGB-D qui est la Kinect. La

Figure 2.15: Acquisition d'une image de profondeur avec la Kinect.

carte de profondeur est réalisée par analyse d'une forme particulière (motif) de la lumière laser infrarouge, la technique utilisée est la lumière structurée qui consiste en la projection d'un motif connu sur la scène et en déduire la profondeur grâce à la déformation de ce dernier (figure 2.16). La Kinect combine la lumière structurée avec deux techniques classiques de

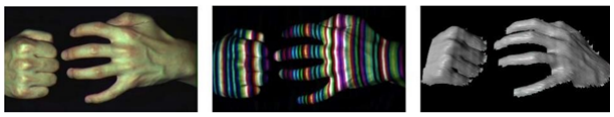

Figure 2.16: Technique de la lumière structurée.

la vision par ordinateur : la profondeur par focus (depth from focus) qui se base sur un principe selon lequel les objets les plus flous dans une image sont les plus loins (figure 2.17), et la profondeur par stéréo qui se base sur le principe suivant: si on regarde la scène sous un autre angle, les objets qui sont proches seront décalés vers le côté plus que les objets qui sont loins figure 2.18.

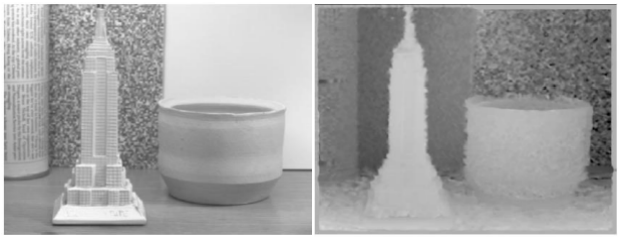

Figure 2.17: Technique de la profondeur par focus.

2.5.2 Etape 2

Dans cette partie on fera le développement d'une méthode de détection de plusieurs personnes grâce à la prédiction rapide et précise des positions $3D$ des articulations du corps à partir

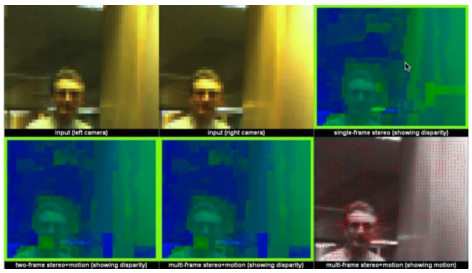

Figure 2.18: Technique de la profondeur par stéréo.

d'une seule image de profondeur, qui est fournie par la Kinect, On va adopter une approche de reconnaissance d'objets et on fera la conception d'une représentation intermédiaire des parties du corps qui transformera le difficile problème d'estimations de pose humaine en un simple problème de classification par pixel. L'approche illustrée sur la figure 2.19 est

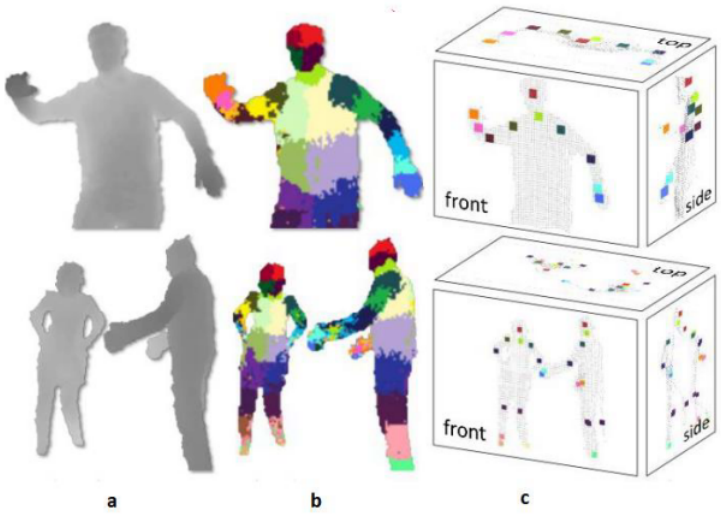

Figure 2.19: A- l'image avec profondeur B- représentation des parties du corps humain C- les positions $3D$ des articulations par pixel.

inspirée par les travaux récents de reconnaissance d'objets qui divise les objets en parties (par exemple [38] [50]).

2.5.2.1 Etiquetage des parties du corps humain

Tous d'abord il faut définir plusieurs étiquettes des parties localisées sur le corps et qui couvrent densément l'anatomie humaine, comme un code de couleur sur le corps, voir figure 2.20. Certaines de ces parties sont définies pour localiser directement les joints d'intérêt particuliers du squelette, tandis que d'autres combleront les lacunes ou pourrait être utilisé en combinaison pour prévoir d'autres articulations. Pour avoir une très bonne détection il faut utiliser 31 parties du corps : les parties gauches, droites, hauts, bas de la tête, le cou, l'épaule, le bras, le coude, le poignet, la main, le torse, la jambe, le genou, la cheville et

Figure 2.20: Etiquetage des parties du corps humain.

le pied, etc. On notera que la précision de définition de ces parties peut être modifiée en

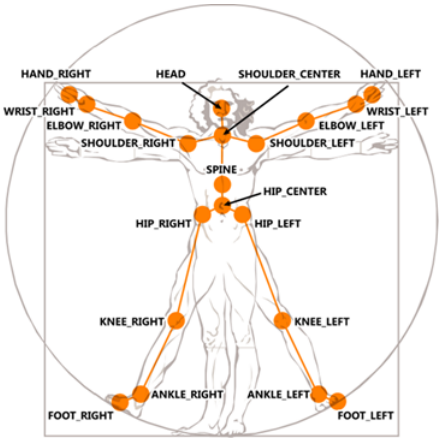

Figure 2.21: Le squelette humain et la nomenclature des principales articulations.

fonction d'une application particulière. Par exemple, dans un scénario de suivi du haut du corps, toutes les parties inférieures du corps pourront être fusionnées. Les parties doivent être suffisamment petites pour localiser avec précision les articulations du corps, mais pas trop nombreuses pour ne pas perdre la capacité de la classification.

2.5.2.2 Caractéristiques de l'image de profondeur

on utilise une simple méthode de comparaison de profondeur, inspirée par ceux dans [44]. À un pixel donné x, on définit la caractéristique suivante :

$$f_\theta(I,x) = d_I(x + \frac{u}{d_I(x)}) - d_I(x + \frac{v}{d_I(x)}) \tag{2.8}$$

avec :

- $d_I(x)$: est la profondeur de l'image I en niveau du pixel x,
- $\theta = (u, v)$: décrit le décalage(le dépassement) selon u et v,

- $\frac{1}{d_I(x)}$: est la normalisation du décalage.

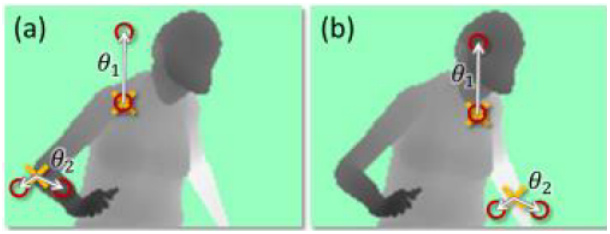

Figure 2.22: Caractéristique de profondeur de pixel dans l'image.

Dans la figure 2.22 les cercles rouges indiquent les pixels de décalage. En (a), les deux dispositifs d'exemple donnent une réponse importante de la différence de profondeur. En (b), les deux mêmes caractéristiques à de nouveaux emplacements d'image donnent une réponse beaucoup plus petite, donc en dépit que le corps humain est symétrique mais dans une image de profondeur la réponse et les informations du côté gauche et droit de la personne sont complétement différentes.
Si la différence de profondeur entre un pixel appartenant aux corps humains et un autre est au-dessus d'un certain seuil cela veut dire que ce dernier fait partie de l'arrière-plan. Il est vrai qu'individuellement, ces caractéristiques ne fournissent qu'un signal faible sur le pixel qui appartient à une partie du corps, mais en les combinant dans une forêt de décision cette méthode devient suffisante pour clarifier avec précision toutes les parties formées. La conception de ces caractéristiques a été fortement motivée par leur calcul efficace : aucun prétraitement est nécessaire, chaque caractéristique a besoin de lire au plus 3 pixels de l'image et d'effectuer au plus 5 opérations arithmétiques.

2.5.2.3 Méthode de classification

La représentation intermédiaire transforme le problème en un seul qui peut être facilement résolu par des algorithmes de classification efficaces. Les paires des parties du corps des images de profondeur sont utilisées comme données entièrement étiquetées pour l'apprentissage du classificateur (voir la figure 2.23). On définit une forêt comme étant un ensemble d'arbres. Chaque arbre est constitué de nœuds diviseur (bleu) et les nœuds de feuille (vert). Les flèches rouges indiquent les différents chemins qui pourraient être prises dans différents arbres par une entrée particulière. Les forêts et les arbres de décision randomisés [41, 37, 4, 12] ont fait leurs preuves comme classificateurs multi-classes rapides et efficaces pour de nombreuses tâches [44, 26, 30], et peut être mis en œuvre efficacement sur le GPU [40]. Comme on a cité précédemment et illustré sur la figure 2.23, une forêt est un ensemble de T arbres de décision, chacun étant constitué de nœuds diviseurs et de feuilles. Chaque nœud diviseur est constitué d'une fonction f_θ et un seuil τ.

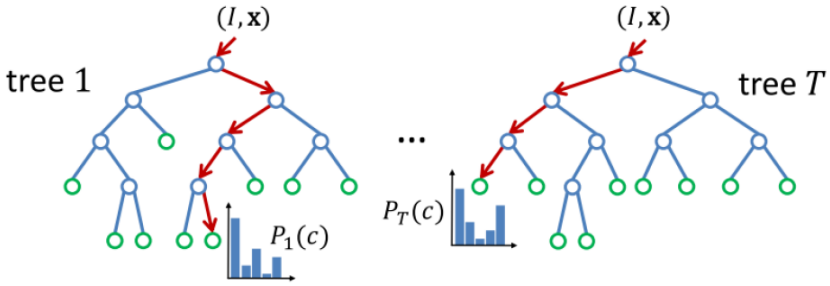

Figure 2.23: Forêts de décision randomisés.

Pour classer un pixel x dans l'image I on commence par la racine et on évalue de façon répétée l'équation (2.8), le choix de la branche (gauche ou droite) s'effectue en fonction de la comparaison avec le seuil τ. Au noeud feuille dans l'arbre t, une distribution $P_t(C/I, x)$ sur une partie du corps des étiquettes C est stockée. Les distributions sont moyennées ensemble pour tous les arbres de la forêt pour donner la classification finale.

$$P_t(C/I, x) = \frac{1}{T}.\sum_{t=1}^{T} P(C/I, x) \tag{2.9}$$

2.5.2.4 Positionnement des joints (articulation)

L'information obtenu précédemment doit être maintenant regrouper sur les pixels pour générer des propositions fiables pour les positions $3D$ des articulations squelettiques. Ces propositions sont le résultat final de cette méthode. Une option simple est d'accumuler les centres $3D$ de masse des probabilités pour chaque partie. Cependant, les pixels périphériques dégradent fortement la qualité d'une telle estimation globale. Au lieu de cela, on utilise l'approche *mean shift* (vous trouverez plus de détaille sur cette approche dans l'annexe L) [14] avec un noyau Gaussien pondéré.

$$f_c(\hat{x}) = \sum_{i=1}^{N} \omega_{ic}.\exp(-||\frac{\hat{x} - \hat{x_i}}{b_c}||^2) \tag{2.10}$$

Où $\hat{x}$ est une coordonnée dans l'espace de la scène réelle $3D$, N est le nombre des pixels de l'image, ω_{ic} est une pondération de pixel, $\hat{x}_i$ est la re-projection du pixel $\hat{x}_i$ dans l'espace de la scène réelle avec la profondeur $d_l(x_i)$, et b_c est une bande passante d'apprentissage par-partie. La pondération de pixel ω_{ic} considère à la fois la probabilité et la profondeur $d_l(x_i)$:

$$\omega_{ic} = P(c|I, x).d_l(x_i)^2 \tag{2.11}$$

Donc le problème de détection des articulations revient à un problème d'optimisation de la densité de probabilité sur une région d'intérêt qui sera résolus par l'algorithme du *Mean*

Shift, voir figure 2.24.

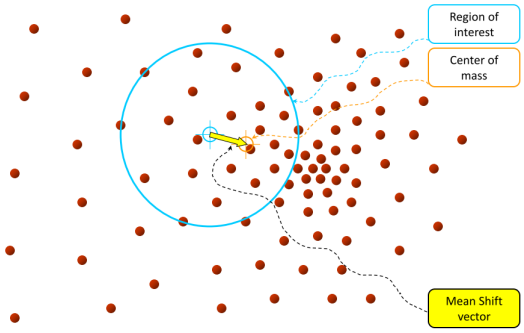

Figure 2.24: Optimisation par la méthode de Mean Shift.

2.5.3 Etape 3

Cette étape consiste à choisir parmi les 31 articulations détectées par la méthode de l'étape précédente celle qui seront utilisées pour constituer un modèle du squelette humain, ce choix dépendra de l'application et du problème pour lequel est destinée la détection. Puisque les positions $3D$ des joints des personnes sont connues, la constitution du squelette se fait en reliant par des droites les articulations d'intérêt.

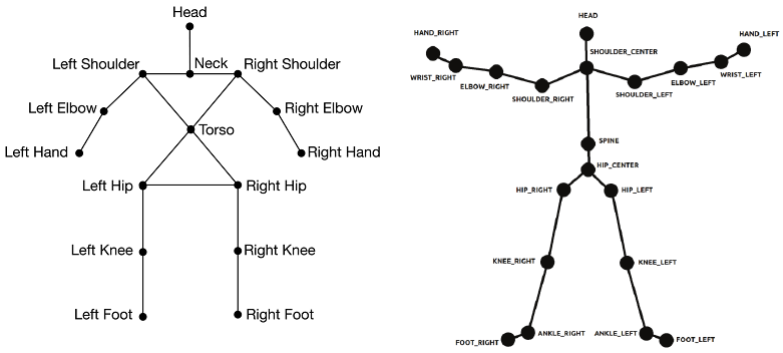

Figure 2.25: Quelques configurations possibles du squelette humain à partir des articulations.

2.6 Conclusion

Ce chapitre nous a permis de faire surgir non seulement l'aspect théorique de la méthode de détection de personne 2D et celle de la détection 3D du squelette humain en faisant le développement sur le plan mathématique, mais aussi l'aspect pratique en expliquant le principe des méthodes utilisées.

On a pu constater que l'importance du choix adéquat et convenable de la méthode de classification, cette dernière sera un des éléments déterminant dans la réussite du processus de la détection.

Chapitre 3

Reconnaissance et suivi des personnes

3.1 Introduction

De nombreuses méthodes de reconnaissance de visages ont été proposées au cours des trente dernières années. La reconnaissance faciale automatique est un challenge tel qu'il a suscité de nombreuses recherches dans des disciplines différentes. Tandis que le suivi des personnes est devenu une compétence essentielle que les robots de service modernes doivent être fournis avec. Comme ils sont censés fonctionner dans des environnements humains et partager l'espace de navigation avec des gens normaux, les robots doivent être en mesure de détecter, reconnaitre et suivre les personnes d'une manière rapide et fiable.

Dans ce chapitre on va présenter une approche de reconnaissance et de suivi qui est parmi les tentatives qui ont comme but de combiner les méthodes de la détection $2D$ et $3D$ afin de gagner plus de robustesse et de rapidité (temps de calcul), en effet cette méthode utilise la détection du squelette humain en $3D$ afin d'extraire seulement la position de la tète, une fois cette position est enregistrée, on fait appelle à la méthode de détection en $2D$ en lui précisant la position de la tête c'est à dire une zone limitée dans la quelle la détection doit être fait ce qui va permettre d'éviter de faire la recherche dans toutes les autres zones de l'image. Après cet étape de la détection vient l'étape de la reconnaissance et de suivi qu'on va expliquer dans ce qui suit.

3.2 Identification et reconnaissance faciale de personnes

L'identification d'une personne à partir de son visage est une tâche aisée pour les humains. En est-il de même pour un robot ? Ceci définit la problématique de la reconnaissance du visage humain, qui a engendré un grand nombre de travaux de recherche au cours des dernières années.

Biométries physiologiques	ADN, visage, empreinte, forme de la main, rétine, odeur, voix
Biométries comportementales	Démarche, visage, voix, écriture, signature

Table 3.1: Les biométries couramment utilisées.

Le visage peut être considéré comme une donnée biométrique[1]. Le tableau 3.1 fournit une liste des indices biométriques couramment utilisés. on remarque que, seules les données: visage et parole appartiennent à la fois aux deux catégories physiologique et comportementale. Cela veut dire entre autres que la biométrie faciale nous permet d'exploiter de nombreuses informations relatives à une personne. Dans la vie quotidienne, le visage est probablement le trait biométrique le plus utilisé par les humains afin de reconnaitre les autres, le visage a de grands avantages par rapport aux autres biométries, parce qu'il est naturel, et facile à acquérir. Dans cette partie du chapitre nous allons présenter une méthode d'identification

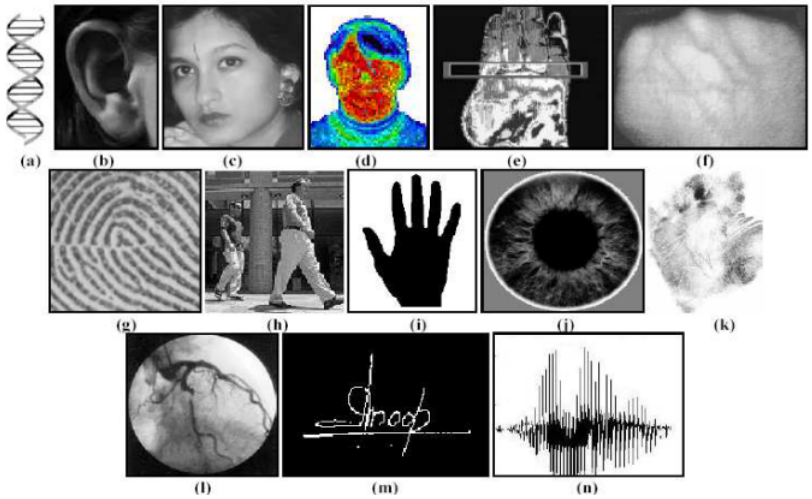

Figure 3.1: caractéristiques biométriques : a) ADN, b) Oreille, c) visage, d) visage infrarouge, e) thermo-gramme main, f) veine main, g) Empreintes digitales, h) marche, i) geste, j) iris, k) empreinte de la paume, I) rétine, m) signature, n) voix.

et de reconnaissance faciale de la personne.

3.3 Principales difficultés de la reconnaissance de visage

Bien que les êtres humains puissent détecter et identifier des visages dans une scène sans beaucoup de peine, construire un système automatique qui accomplit de telles tâches représente

[1]Une donnée biométrique est une caractéristique ou bien une information qui permet l'identification et la reconnaissance d'une personne sur la base de ce qu'il est (caractéristiques physiologiques ou comportementales)

un sérieux défi. Ce défi est d'autant plus grand lorsque les conditions d'acquisition des images sont très variables. Il existe deux types de variations associées aux images de visages : inter et intra sujet. La variation inter sujet est limitée à cause de la ressemblance physique entre les individus. Par contre la variation intra sujet est plus vaste. Elle peut être attribuée à plusieurs facteurs. Chaque visage individuel peut générer une grande variété d'images différentes. Cette grande diversité d'images de visages rend l'analyse difficile. Outre les différences générales entre les faces des variations dans l'apparence d'images de visage posent de grands problèmes à l'identification. Ces variations sont recensées comme suit:

- Changements d'éclairage influencent l'apparition d'un visage, même si la pose de la face est fixée,
- Variations de pose peuvent entraîner des changements dramatiques dans les images,
- Les expressions faciales : une autre source de variations dans les images. Seuls quelques points de repère du visage qui sont directement couplés avec la structure osseuse du crâne, comme la distance interoculaire ou la position générale de l'oreille sont constants dans un visage. La plupart des autres caractéristiques peuvent changer leur configuration spatiale ou position en raison de l'articulation de la mâchoire ou à l'action des muscles, comme les sourcils mobiles, les lèvres ou les joues.

Figure 3.2: Les expressions faciales.

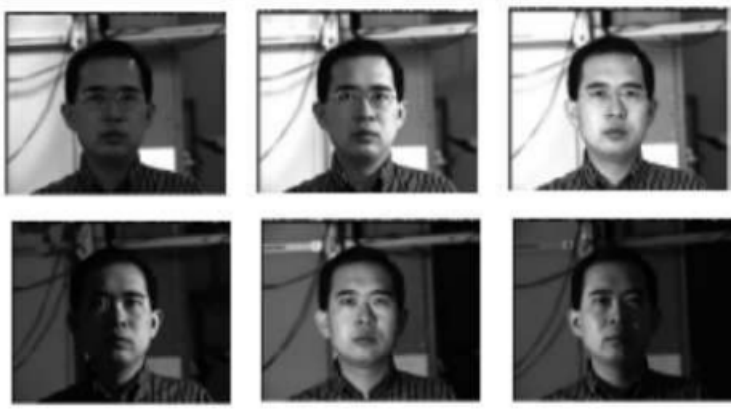

Figure 3.3: Effet du l'illumination sur les images de visage.

3.4 Méthodes de reconnaissance de visage

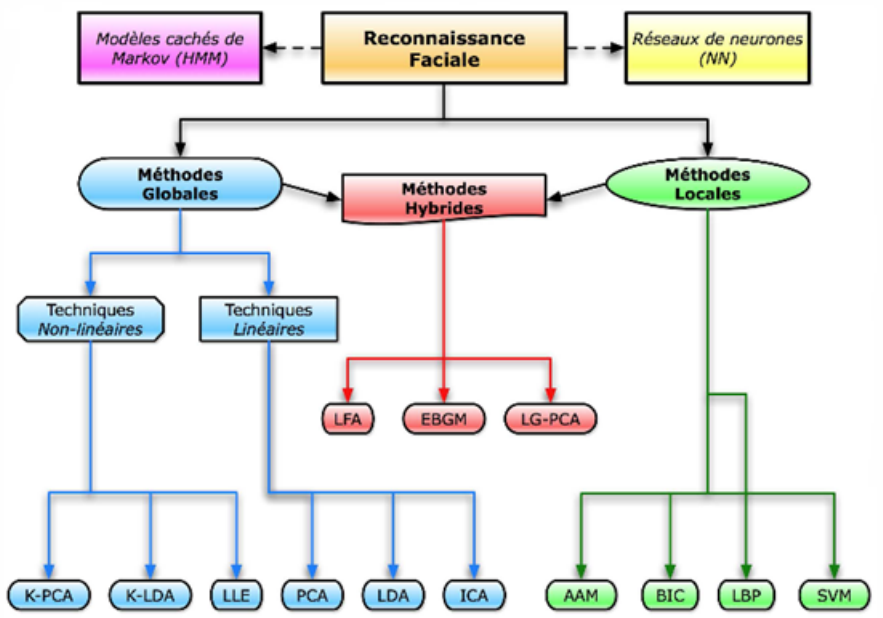

Figure 3.4: Classification des principaux algorithmes utilisés en reconnaissance faciale [6].

3.4.1 Méthodes globales

Les méthodes globales sont basées sur des techniques d'analyse statistique bien connues. Il n'est pas nécessaire de repérer certains points caractéristiques du visage (comme les centres des yeux, les narines, le centre de la bouche, etc.) à part pour normaliser les images. Dans ces méthodes, les images de visage (qui peuvent être vues comme des matrices de valeurs de pixels) sont traitées de manière globale et sont généralement transformées en vecteurs, plus faciles à manipuler.

L'avantage principal des méthodes globales est qu'elles sont relativement rapides à mettre en œuvre et que les calculs de base sont d'une complexité moyenne. En revanche, elles sont très sensibles aux variations d'éclairement, de pose et d'expression faciale.

Ces méthodes utilisent principalement une analyse de sous-espaces de visages. Cette expression repose sur un fait relativement simple : une classe de "formes" qui nous intéresse (dans notre cas, les visages) réside dans un sous-espace de l'espace de l'image d'entrée. Nous pouvons distinguer deux types de techniques parmi les méthodes globales : Les techniques linéaires et les techniques non linéaires.

Les techniques linéaires projettent linéairement les données d'un espace de grande dimension (par exemple, l'espace de l'image originale) sur un sous-espace de dimension inférieure. Malheureusement, ces techniques sont incapables de préserver les variations non convexes des variétés (géométriques donc au sens mathématique du terme) de visages afin de différencier

des individus.
La technique linéaire la plus connue et sans aucun doute l'Analyse en Composantes Principales (PCA) également appelée transformée de Karhunen-Loeve.
Il existe d'autres techniques également construites à partir de décompositions linéaires comme l'analyse discriminante linéaire (LDA) ou encore l'analyse en composantes indépendantes (ICA).
Tandis que le PCA construit un sous-espace pour représenter de manière "optimale" (mathématiquement parlant) seulement "l'objet" visage, le LDA construit un sous-espace discriminant pour distinguer de façon "optimale" les visages de différentes personnes.
Elle permet donc d'effectuer une véritable séparation de classes (une explication détaillée du LDA pourra être consultée en Annexe H).Des études comparatives montrent que les méthodes basées sur le LDA donnent généralement de meilleurs résultats que les méthodes basées sur le PCA [6].
L'algorithme ICA, quant à lui, est une généralisation de l'algorithme PCA avec lequel il coïncide dans le cas de données gaussiennes. L'algorithme ICA est basé sur le concept intuitif de contraste et permet d'éliminer la redondance statistique des données de départ.
Les méthodes globales linéaires basées sur l'apparence ne sont pas assez précises pour décrire les subtilités des variétés (géométriques) présentes dans l'espace de l'image originale. Afin de pouvoir traiter ce problème de non-linéarité en reconnaissance faciale, de telles méthodes linéaires ont été étendues à des techniques non linéaires basées sur la notion mathématique de noyau (≪kernel≫) comme le Kernel PCA et le Kernel LDA.Ici, une projection non linéaire (réduction de dimension) de l'espace de l'image sur l'espace de caractéristiques (≪feature space≫) est effectuée ; les variétés présentes dans l'espace de caractéristiques résultant deviennent simple, de même que le subtilités des variétés qui sont préservées.
Bien que les méthodes basées sur le noyau peuvent atteindre une bonne performance sur les données d'entraînement, il ne peut pas en être de même pour de nouvelles données en raison de leur plus grande flexibilité ; contrairement aux méthodes linéaires.

3.4.2 Méthodes locales

Les méthodes locales, basées sur des modèles, utilisent des connaissances a priori que l'on possède sur la morphologie du visage et s'appuient en général sur des points caractéristiques de celui-ci. Ces méthodes constituent une autre approche pour prendre en compte la non-linéarité en construisant un espace de caractéristiques local et en utilisant des filtres d'images appropriés, de manière à ce que les distributions des visages soient moins affectées par divers changements.
Les approches Bayesiennes, les machines à vecteurs de support, la méthode des modèles actifs d'apparence ou encore la méthode ≪local binary pattern≫ ont été utilisées dans ce but.

Toutes ces méthodes ont l'avantage de pouvoir modéliser plus facilement les variations de pose, d'éclairage et d'expression par rapport aux méthodes globales. Toutefois, elles sont plus lourdes à utiliser puisqu'il faut souvent placer manuellement un assez grand nombre de points sur le visage alors que les méthodes globales ne nécessitent de connaître que la position des yeux afin de normaliser les images, ce qui peut être fait automatiquement et de manière assez fiable par un algorithme de détection.

3.4.3 Méthodes hybrides

Les méthodes hybrides permettent d'associer les avantages des méthodes globales et locales en combinant la détection de caractéristiques géométriques (ou structurales) avec l'extraction de caractéristiques d'apparence locales. Elles permettent d'augmenter la stabilité de la performance de reconnaissance lors de changements de pose, d'éclairement et d'expressions faciales.
L'analyse de caractéristiques locales et les caractéristiques extraites par ondelettes de Gabor sont des algorithmes hybrides typiques.

3.5 Principales taches en identification et reconnaissance

Le principe de fonctionnement de base d'un système de détection et de reconnaissance faciale peut être résumé en par le schéma bloc suivant (fig. 3.5) : L'approche de l'identification

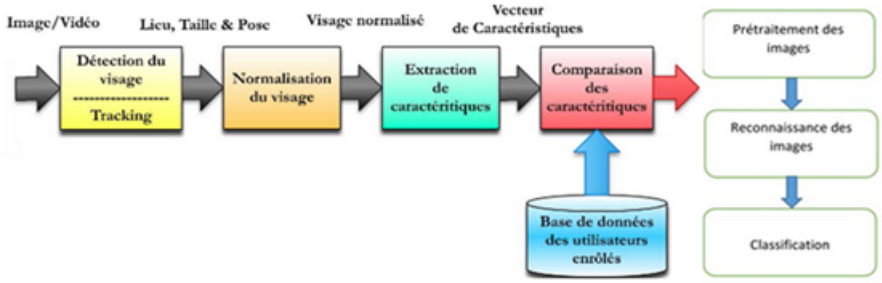

Figure 3.5: Principe de fonctionnement de base et principales tâches d'un système de reconnaissance faciale.

du visage comprend une variété de méthodes dans les quelles trois taches principale sont effectuer, voir figure 3.6.

- *Le prétraitement:* pour atténuer les influences non contrôlées liées à l'environnement, les images acquissent sont prétraités par rapport à l'éclairage et à l'orientation de la tête. De cette façon, les données d'entrée seront plus homogènes et on aura une reconnaissance plus précise.

- *La reconnaissance:* après le prétraitement les images seront normalisées ensuite seront introduits dans une base de projection basé sur l'approche *Fisher-faces*.

- *La classification:* les images de reconnaissance sont projetées dans un sous-espace, qui est calculée lors de la phase de formation. Le vecteur caractéristique est ensuite classifié avec des méthodes d'apprentissage tels que *(K-Nearest Neighbors)* ou *SVM (Support Vector Machines)*.

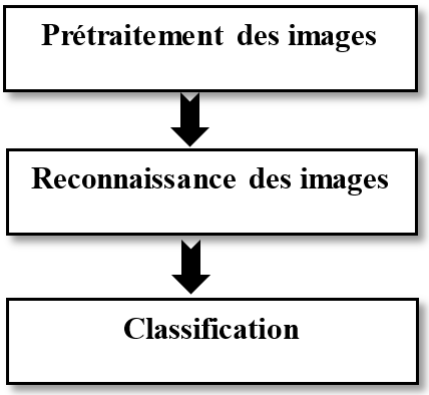

Figure 3.6: Organigramme des principales tâches en reconnaissance facial.

3.5.1 Prétraitement des images

3.5.1.1 Correction de l'illumination

pour compenser les effets et les variations de l'éclairage de l'environnement sur le visage humain, nous effectuant une normalisation en une forme standard afin d'atteindre des conditions d'éclairage uniforme, tout en conservant les traits du visage [48]. La première étape est d'effectuer une égalisation d'histogramme, ce qui permet d'obtenir une distribution uniforme des valeurs du gris à travers l'image. Cela devient visible, comme une augmentation du contraste local [29]. Une transformation logarithmique peut être appliquer afin d'étendre le faible niveau de gris et de réduire le haut niveau de gris [42] :

$$g(x, y) = ln(f(x, y) + 1) \tag{3.1}$$

L'image originale est notée $f(x, y)$ et de l'image logarithmique est $g(x, y)$. Une autre possibilité consiste à appliquer une correction gamma, ce qui évite le bruit dans les zones sombres de l'image qui peut se produire avec la transformation logarithmique [43] :

$$g(x, y) = f(x, y)^{\gamma} \tag{3.2}$$

Figure 3.7: (a) image RGB, (a) image niveau de gris (c) égalisation d'histogramme.

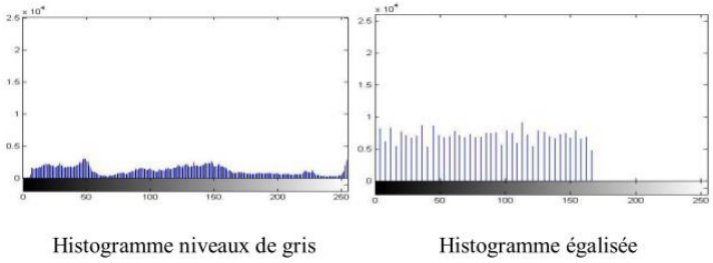

Figure 3.8: histogramme de l'image avant et après égalisation.

Cette normalisation permet de réduire l'écart entre les zones sombres et claires du visage par une transformation non linéaire des niveaux de gris. L'exposant γ est le paramètre que l'utilisateur peut choisir pour contrôler la force de la transformation.
Suite à la remise à l'échelle des valeurs de gris, l'image est transformée à partir d'une représentation de l'espace en une représentation en fréquence par l'intermédiaire d'une transformation en cosinus discrète (TCD).
À basse fréquence les coefficients sont mis à bas-échelle, car ils contiennent des informations d'éclairage [42]. Le nombre de mises à l'échelle des coefficients doit être équilibré afin de minimiser la variation d'éclairage, mais garder le pouvoir discriminant des traits du visage. L'illumination globale de l'image peut être contrôlée par le premier coefficient $C(0,0)$ de la DCT [29]. Par conséquent, il est réglé par rapport au niveau de gris moyen souhaité μ.

$$C(0,0) = \log(\mu) = \sqrt{MN} \tag{3.3}$$

Enfin, l'image est transformée à nouveau dans le domaine spatial via la DCT inverse. Grâce à ce prétraitement les désavantages des conditions d'éclairage peuvent être compensées.

3.5.1.2 Correction de la pose de la tête

Un inconvénient majeur des méthodes de reconnaissance basé sur l'apparence est la nécessité pour les deux données celle de formation et de test qu'elle doivent être autant que possible

aligné pour pouvoir les comparer. La déviation d'alignement précis de pixels conduit à une

Figure 3.9: exemple de variation de pose.

dégradation des performances de reconnaissance [1]. La correction automatique de la pose de la tête vise à aligner automatiquement les images de visage à partir de leur point de vue d'enregistrement d'origine à un point de vue virtuel normalisée d'enregistrement frontal. De cette façon, la dépendance de la variation des poses est réduite au minimum.
Pour établir la relation spatiale entre l'orientation du visage réel et la perspective normalisée, des points d'intérêt du visage tels que la position des yeux et le nez sont détectés dans l'image couleur en utilisant tout d'abord l'algorithme de détection 3 D du squelette grâce auquel on peut connaitre la position 3 D de la tête et donc réduire la zone de recherche du visage ensuite le détecteur *Viola-Jones* pour la détection faciale [46].
Ensuite, nous déterminons les coordonnées $3D$ des points d'intérêt du visage à partir de l'image de profondeur pour construire le système de coordonnées X_{face} les points de l'axe x sont de l'œil gauche jusqu'à l'œil droit, l'axe y est parallèle avec le nez et à l'axe z est choisi orthogonalement aux axes x et y dans la direction d'observation. Ensuite, nous calculons la matrice de transformation T_{cam} qui aligne le système de coordonnées X_{face} avec système de coordonnées du monde réel X_W qui est définit par le repère camera du capteur $RGB-D$. Enfin, la face $3D$ est déplacé en face de la caméra avec une autre transformation T_{norm} et projetée dans un plan virtuel d'image qui capture la face frontale. L'échelle est normalisée au cours de la projection en choisissant la matrice K de la caméra virtuelle de manière à ce que les yeux et le nez de la face $3D$ normalisé correspondent au modèle de visage fixe. Toute la transformation depuis le système de coordonnés X_{face} de la face $3D$ enregistré jusqu'aux coordonnées de l'image virtuel frontal u_{norm} est alors donnée par :

$$u_{norm} = KT_{norm}T_{cam}x_{face} \tag{3.4}$$

3.5.2 Reconnaissance

Le processus se déroule en deux étapes bien distinctes : La phase d'apprentissage Figure3.10 qui s'effectue en amont et dont les calculs sont parfois très longs et la phase de reconnaissance (figure 3.11) qui a lieu lorsqu'un individu se présente physiquement devant la caméra.

3.5.2.1 Phase apprentissage

Par analogie, la phase d'apprentissage correspondrait à un enrôlement réel de personnes qui seraient enregistrées dans une base de données. Cette phase consiste donc à récolter une grande quantité d'images de visage afin de se constituer une base de données de départ. Dans un premier temps, on construit une matrice Γ contenant M images de la base d'apprentissage ("Training Set" en anglais). Les images sont ensuite normalisées (normalisation géométrique, normalisation de l'histogramme, etc.) puis l'image moyenne est calculée. On réajuste ensuite les données par rapport à la moyenne, pour pouvoir suivre, de manière simple, le comportement des valeurs d'écart-type, de variance et de covariance (rappelons-nous que toutes ces formules font intervenir une soustraction par rapport à la moyenne). On applique alors un algorithme de reconnaissance globale à cette matrice réajustée. La chose à retenir est que ces algorithmes fournissent en sortie ce que l'on appelle une matrice de projection W qui va nous être très utile dans la seconde partie de la phase d'apprentissage. Elle consiste en la projection des images apprises (normalisées et réajustées par rapport à la moyenne) sur un espace vectoriel dont les vecteurs sont les éléments de notre matrice de projection W Toutes ces projections sont finalement stockées dans une grande base de données.

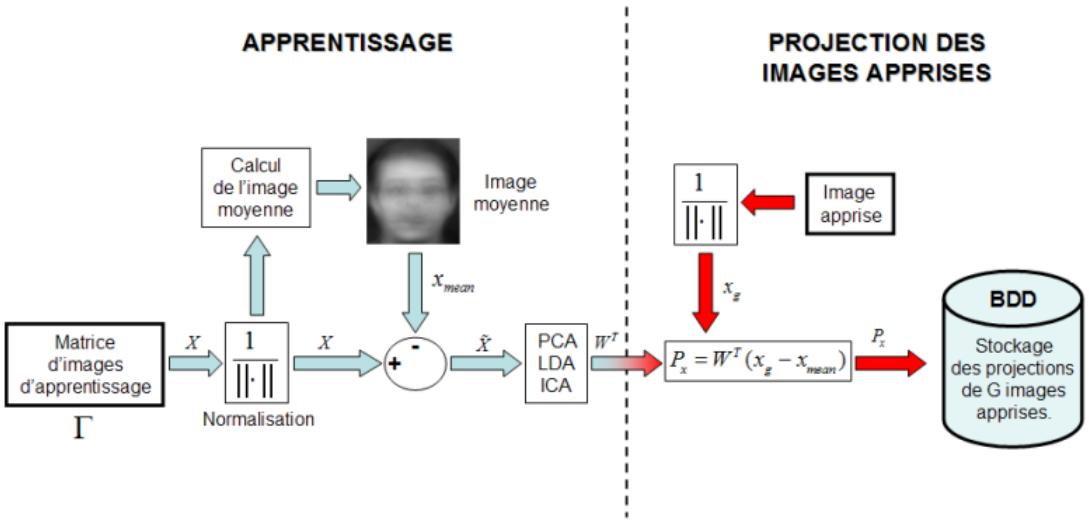

Figure 3.10: phase d'apprentissage dans un processus de reconnaissance faciale utilisant une méthode globale.

3.5.2.2 Phase reconnaissance

Passons maintenant à la phase de reconnaissance. Lorsqu'une nouvelle image de la base de Test ("Probe Set" en anglais) est mise devant le système, elle est normalisée et l'image moyenne de la base d'apprentissage lui est retirée. On la projette ensuite sur l'espace vectoriel relatif à la matrice de projection W afin de comparer avec toutes les projections issues de

la phase d'apprentissage et qui étaient stockées dans la base de données. Par le terme comparer, il faut comprendre qu'il faut effectuer un calcul de distance entre les projections vectorielles. Il semble logique que plus la distance entre deux projections est petite, plus ces deux projections se ressemblent. Ainsi, le résultat de la reconnaissance est l'image de la base d'apprentissage qui ressemble le plus à la nouvelle image présentée au système.

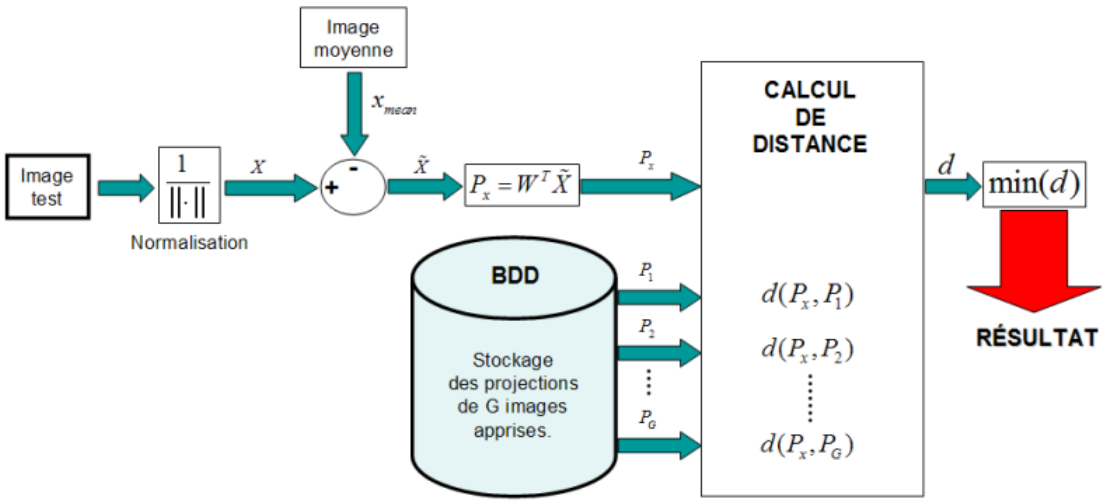

Figure 3.11: Phase de reconnaissance d'un système de reconnaissance faciale.

3.5.2.3 Analyse Discriminante Linéaire (Fisher-Faces)

L'algorithme *LDA* est né des travaux de *Belhumeur et al.* de *la Yale University (USA)*, en 1997. Il est aussi connu sous le nom de *Fisher-Faces*. L'algorithme *LDA* effectue une véritable séparation de classes (Figure3.12). Pour pouvoir l'utiliser, il faut donc au préalable organiser la base d'apprentissage d'images en plusieurs classes. Une classe par personne et plusieurs images par classe. Le *LDA* analyse les vecteurs propres de la matrice de dispersion des données, avec pour objectif de maximiser les variations inter-classes tout en minimisant les variations intra-classes.

L'algorithme *LDA* permet d'effectuer une véritable séparation de classes, selon un critère mathématique qui minimise les variations entre les images d'un même individu (variation intra-classe) tout en maximisant les variations entre les images d'individus différents (variation inter-classes), (voir annexe H).

3.5.3 Classification et identification des personnes

La projection W_{opt} est utilisé pour projeter une image arbitraire x à l'espace caractéristique via $y = W_{opt}$. x la classification est effectuée en trouvant sur quelles classes de projection le vecteur est le mieux décrit. Spécifiquement, on recherche un échantillon y_{ki} de la classe C_i

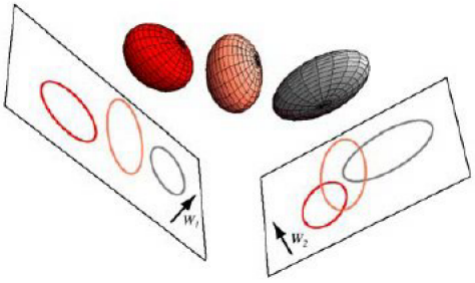

Figure 3.12: llustration du principe de séparation optimale des classes par le LDA. Trois distributions 3D sont projetées sur deux sous-espaces 2D décris par les vecteurs W_1 et W_2 . Puisque le LDA essaye de trouver la plus grande séparation parmi les classes, on voit bien que W_1 est ici le vecteur optimal.

qui minimise la distance dans l'espace de visage ($DIFS$) à la projection de l'image test y_{test} :

$$DIFS = \| y_{test} - y_{ik} \|^2 \tag{3.5}$$

Une autre alternative pour pouvoir distinguer et reconnaitre les personnes au lieu de calculer la distance euclidienne et trouve celle qui est la plus petite, le vecteur y_{ki} de la classe C_i peut également être classifiée par des méthodes d'apprentissage tels que *Support Vector Machines* (SVM) ou *K-plus proches voisins* (KNN). Le modèle de classification doit être formé à partir des données de formation (base de données acquise au préalable) avant que la classification puisse être effectuée.

3.6 Tracking de l'être humain

L'objectif du suivi d'une cible est de déterminer sa position de manière continue fiable tout au long (du flux vidéo). Le suivi (notamment de multiples cibles) en temps réel représente une étape primordiale pour des applications d'analyse d'activités et de compréhension d'évènements. Pour des applications de robots de services, le robot est censé fonctionner dans des environnements humains et partager l'espace de navigation avec des gens au comportement. Il doit généralement suivre de manière continue, rapide et fiable, tous les éléments impliqués dans la scène, et cela même lorsqu'ils sont partiellement occultés par d'autres objets ou interagissent avec. Dans certains cas, l'information spatiale en $3D$ d'une cible suivie peut être nécessaire, requérant ainsi l'utilisation de plusieurs caméras.
Diverses stratégies de suivi de cibles ont été développées. La Figure 3.13 présente la taxonomie de ces méthodes. On peut voir qu'on peut les regrouper en trois grandes familles :

- Les méthodes de suivi basées point, où les objets détectés dans des images consécutives sont représentés par des points, et l'association de ces points est basée sur l'état précédent de l'objet en terme de position et de mouvement. Cette approche nécessite l'addition d'un mécanisme de détection afin de détecter les objets dans chaque image.

- Celles basées noyau, où le terme ¡¡ noyau ¿¿ se réfère à la forme et à l'apparence de l'objet. Par exemple, le noyau peut être un motif rectangulaire ou une forme elliptique avec un histogramme associé. Les objets sont suivis en calculant le mouvement du noyau dans des images consécutives.

- Celles basées silhouette, qui effectuent le suivi d'un objet en estimant sa région dans chaque image puis utilisent les informations extraites de cette région pour le suivi. Les informations peuvent être sous forme de densité (l'apparence ou de modèle de forme qui sont généralement représentés par des cartes de contours. Pour un modèle d'objet, la silhouette est suivie soit par un appariement de forme ou par une évolution des contours.

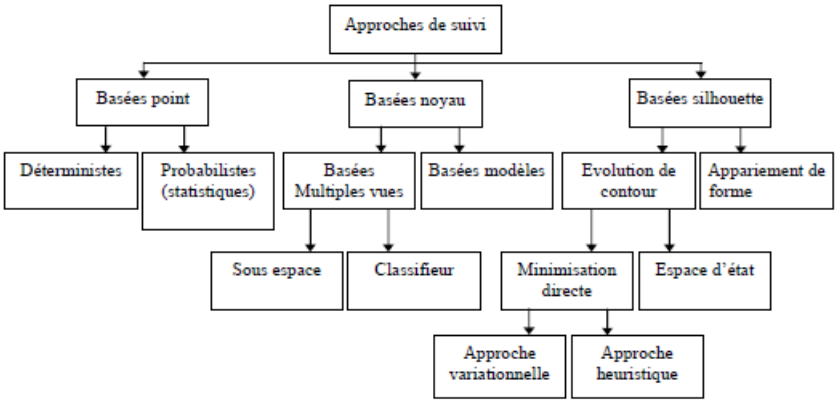

Figure 3.13: Taxonomie des approches de suivi d'objets [51].

Le suivi des personnes est devenu une compétence essentielle que les robots de service modernes doivent être fournis avec. Comme ils sont censés fonctionner dans des environnements humains et partager l'espace de navigation avec des gens normaux, les robots doivent être en mesure de détecter et suivre les personnes d'une manière rapide et fiable.
Tel est le cas pour les robots de service utilisés dans les expositions et les lieux publics pour divertir les visiteurs et leur fournir des informations utiles.
Comme on a constaté ci-dessous, Différente approches ont été utilisés pour la réalisation de cette tâche, avec l'utilisation de plusieurs filtres pour prédire la position et le mouvement de l'être humain, parmi ces filtres les plus utilisés est le filtre de *Kalman*.

3.6.1 Filtre de Kalman

3.6.1.1 Algorithme du filtre de Kalman discret

Le filtre de Kalman estime un processus en utilisant une forme de control de rétroaction, le filtre estime l'état du processus à un moment donné, puis obtient un feedback sous forme de mesures (bruitées). Par conséquent, les équations du filtre de Kalman se répartissent en deux groupes : les équations de mise à jour du temps et les équations de mise à jour de la mesure.

Les équations de mise à jour du temps sont responsables de la projection en amont (en temps) de l'état actuel et de la covariance de l'erreur pour obtenir une estimation a priori de l'état pour le pas du temps suivant. Les équations de la mise à jour de la mesure donne un feedback pour incorporer une nouvelle mesure dans l'estimation de l'état a priori afin d'obtenir une meilleur estimation a posteriori. Les équations de mise à jour du temps peuvent être vues comme des équations de prédiction, et les équations de mise à jour de la mesure peuvent être vues comme des équations de correction.

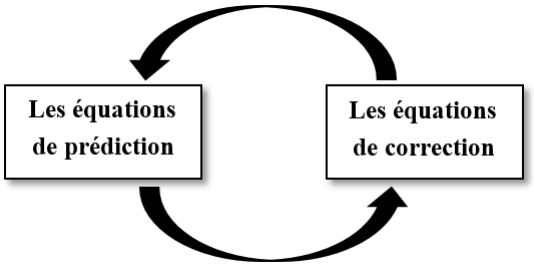

Figure 3.14: Le cycle du filtre de Kalman discret.

Les équations de prédiction

$$\begin{cases} \hat{x}_k = A\hat{x}_{k-1} + Bu_k \\ P_k^- = AP_{k-1}A^T + Q \end{cases} \tag{3.6}$$

Les équations de correction

$$\begin{cases} K_k = P_k^- C^T (CP_k^- C^T + R)^{-1} \\ \hat{x}_k = \hat{x}_k^- + K_k(y_k - H\hat{x}_k^-) \\ P_k = (I - K_k C)P_k^- \end{cases} \tag{3.7}$$

(Mise à jour de l'état ("Prédiction")
(1) (Prédiction de l'état)
$\hat{x}_k = \hat{x}_k^- + K(y_k - C\hat{x}_k^-)$
(2) (Prédiction de la covariance de l'erreur)
$P_k^- = AP_{k-1}A^T + Q$

Table 3.2: Etapes de prédiction

(Mise à jour de la mesure ("Correction"))
(1) (Calcul du gain de Klaman)
$K_k = P_k^- C^T (CP_k^- C^T + R)^{-1}$
(2) (Mise à jour de l'estimateur avec la mesure y_k)
$\hat{x}_k = \hat{x}_k^- + K_k(y_k - H\hat{x}_k^-)$
(3) (Mise à jour de la covariance de l'erreur)
$(P_k = (I - K_k C)P_k^-$

Table 3.3: Etape de correction

3.6.1.2 Application du filtre de Kalman pour le suivi des personnes

L'approche qu'on a utilisé, qu'elle soit basée sur un filtre de Kalman tridimensionnel ou bien bidimensionnel,est avec un modèle de mouvement uniforme à vitesse constante.
L'utilisation du filtre de *Kalman* dans ce travail a comme but de diminuer le temps de calculs et de renforcer les algorithmes de détection $2D$ et $3D$, en effet l'estimation de la position de l'être humain.
Le vecteur d'état X_k est défini comme :

$$X_k = [x(k), y(k), z(k), v_x(k), v_y(k), v_z(k)]$$

avec $(x(k), y(k), z(k))$ la position et $(v_x(k), v_y(k), v_z(k))$ la vitesse dans l'image k. Le vecteur d'état x_k et le vecteur d'observation y_k sont reliés par les équations suivantes :

$$\begin{cases} X_{k+1} = AX_k + w_k \\ y_k = CX_k + v_k \end{cases} \tag{3.8}$$

avec w_k et v_k les bruits du processus et de mesure, supposés bruits blancs gaussiens, A la matrice de transition de l'état et C la matrice d'observation, avec ΔT la période d'acquisition :

$$A = \begin{pmatrix} 1 & 0 & 0 & \Delta T & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta T & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta T \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Les trois composantes sont supposées indépendantes, ainsi les matrices de covariance sont diagonales.

3.6.2 Suivi 3D du squelette humain

Le suivi consiste à connaitre le positionnement des articulations à l'instant t constituant le squelette humain à partir de la connaissance des positions des articulations à l'instant $t-1$ (figure 3.15). Au chapitre 2, nous avons expliqué comment ce fait la détection des 31

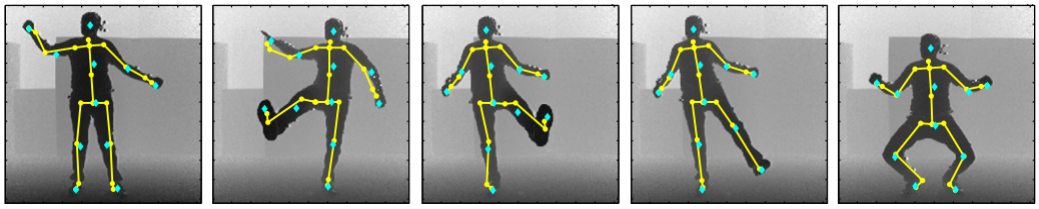

Figure 3.15: suivi du squelette humain.

articulations. En effet à $t = t_0$ une image en profondeur est acquise, puis chaque pixel est caractérisé par une fonction f_θ ensuite classifié par la méthode de « forêts de décision randomisées » et au final pour chaque pixel x_i une probabilité P_i^0. Enfin on utilise l'algorithme *Mean Shift* (voir Annexe L) pour générer les positions $3D$ des articulations et ce dernier doit être initialisé pour une valeur initiale qui sera choisie arbitrairement puisque c'est l'instant $t = t_0$. A la fin l'algorithme va converger vers une nouvelle valeur qui sera associer à une articulation. A l'instant $t = t_1$ commence le processus de suivi, la procédure précédente est reconduite, la seule différence réside dans le choix de la valeur initiale de l'algorithme d'optimisation *Mean Shift* qui sera choisie comme étant la valeur associer à l'articulation précédemment détecté (à instant $t = t_0$). Donc à l'instant $t = t_1$ une nouvelle distribution de probabilité de chaque partie du corps est générer (figure 3.16), l'algorithme *Mean*

Shift prend comme point de départ l'articulation précédente ($t = t_0$) cible bleu, une zone de recherche est choisie (cercle bleu et la future position de convergence est calculer qui est le centre de masse de la zone choisit (cible orange) après quelque itération (figure 3.17 et 3.18) l'algorithme *Mean shift* converge vers le centre de masse (cible orange (figure 3.19)) de toute la distribution et qui représente l'articulation de l'instant $t = t_1$.

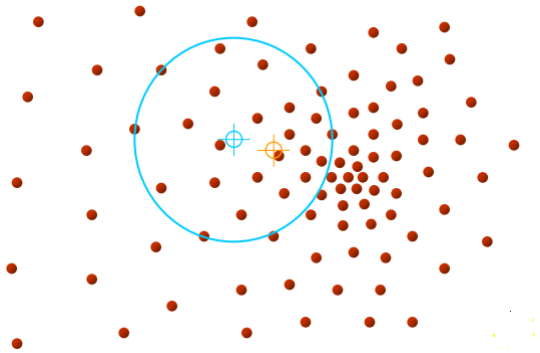

Figure 3.16: Distribution de probabilité d'une partie du cops à l'instant $t = t_1$.

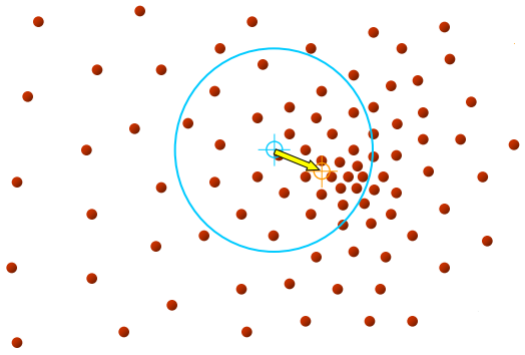

Figure 3.17: Première convergence vers le centre de masse de la zone choisit (cercle bleu).

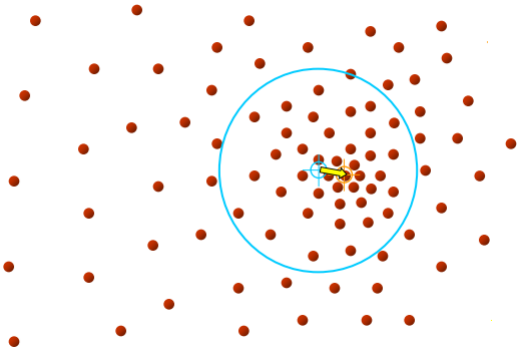

Figure 3.18: Deuxième convergence vers le centre de masse de la zone choisit (cercle bleu).

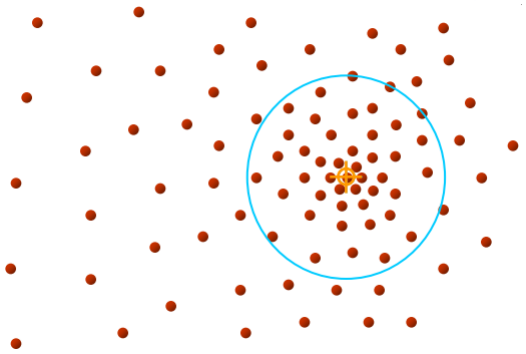

Figure 3.19: Convergence vers l'articulation de l'instant $t = t_1$.

3.7 Conclusion

Ce chapitre nous a permis de montrer et d'expliquer les différentes étape par les quelles passe notre processus de reconnaissance et suivi faciale, qui utilise la méthode de *Viola et Jones* pour la détection du visage présentée dans le chapitre 2 cela va d'un prétraitement qui consiste en une correction sur l'illumination et l'orientation de la pose ce qui permet d'avoir une méthode plus robuste et plus indépendante des conditions de l'environnement, ensuite une phase d'apprentissage dans laquelle une base de donnée est générée et sera utilisée comme référence de comparaison en reconnaissance, après vient la phase la plus importante est celle

de la reconnaissance dans laquelle l'image en temps réel est comparée avec les images de base de donnée pour classer est-ce que la personne est inconnu ou bien connu, la phase finale étant le suivi de la personne qui sera très important pour beaucoup d'applications en robotique de services.

Chapitre 4

Implémentation et Résultats

4.1 Introduction

Dans ce chapitre, nous présenterons les différentes étapes de l'implémentation de plusieurs algorithmes de détection et de suivi étudiés dans les chapitres précédents.
Nous commencerons par présenter la platforme expérimentale, le robot $B21r$ et la caméra *RGB-D Kinect*, ainsi que l'environnement logiciel *ROS* et le driver de la *Kinect*.
Les résultats expérimentaux seront détaillés et commentés, à travers différents scénarios, dans le paragraphe (1.5).

4.2 Présentation de la plateforme expérimentale B21r du CDTA

4.2.1 Présentation de l'environnement matériel

Le matériel disponible pour l'application développée dans le cadre de ce mémoire (voir fig. 4.1) est constitué de :

- Le robot mobile B21r du CDTA,
- Une caméras RGB-D (Kinect for windows) est placée sur le haut du robot.
- Une unité de traitement (Un PC portable avec une carte graphique),
- Une connexion est disponible entre le *B21r* et l'ordinateur via un câble USB.

4.2.2 Le Robot B21r

Le robot mobile *B21r* est une plateforme expérimentale construite par la société *iRobot* pouvant se déplacer sur un terrain non accidenté ayant comme type de traction, la traction

Figure 4.1: Plateforme du CDTA : Robot mobile B21r et caméras Kinect

synchrone. Il dispose quatre roues décentrées orientables tournant selon deux axes : une rotation selon l'axe y pour engendrer la translation, et une rotation selon l'axe verticale au sol pour engendrer une rotation sur lui-même. Ce robot mobile est muni de deux ceintures de capteurs à ultrasons, une ceinture de capteurs infrarouges, un laser.

4.2.3 Description de la Kinect

La Kinect est un périphérique de Microsoft constitué d'une barre horizontale reliée à une petite base avec un pivot motorisé, celui-ci permet à la caméra d'effectuer des petits mouvements vers le haut ou le bas, afin d'adapter la perception de la caméra,(figure 4.4).
Le dispositif comporte une caméra *RGB*, un capteur de profondeur «*3D depth sensor*» et une série de microphones multi-réseaux.

Figure 4.2: La Kinect de Microsoft

4.2.3.1 Les composants de la Kinect

La kinect est composée de trois parties essentielles;

- La camera *RGB* (Red Green Blue) ;
- Le capteur de profondeur « *3D depth sensor* » ;
- Le réseau *des microphones* ;

4.2.3.2 Caractéristiques de la Kinect

- Champ de vision :
  - Champ de vision horizontal : 57 degrés ;
  - Champ de vision vertical : 43 degrés ;
  - Portée du capteur :$1.2m$–$3.5m$.
- Flux de données :
  - 320×240 en couleur 16 bits à 30 images/sec ;
  - 640×480 en couleur 32 bits à 30 images/sec.

4.3 Présentation de l'environnement logiciel

4.3.1 ROS

Comme son nom l'indique *Robot Operating System*, souvent abrégé en *ROS*, est un système d'exploitation pour robot. Il a la particularité d'être open-source et donc soutenu par une communauté grandissante de chercheurs et de développeurs.
En toute rigueur, bien que la version disponible de *ROS* fournit des services proches d'un *OS* (abstraction du matériel, gestion de la concurrence et des processus, etc) tout en ayant les caractéristiques d'un *middleware* (un réseau d'échange d'informations entre différentes applications informatiques), il n'est pas un système d'exploitation dans le sens traditionnel du mot, il sera préférable de considérer *ROS* comme étant un environnement de programmation robotique. En effet, *ROS* s'éloigne de l'image que nous avons d'un système d'exploitation comme *Windows* ou *Linux*.
Il faut souligner que *ROS* a besoin d'un système d'exploitation traditionnel pour fonctionner les versions stables actuelles doivent être installées de mieux sur la distribution *Ubuntu* de Linux; il faut noter aussi qu'il existe une autre version de *ROS* qui est un système d'exploitation à part entière offrant les même services qu'un OS ordinaire cette version n'est pas disponible pour le grand publique [27], la figure 4.3 présente des exemples de robots compatibles avec ROS.

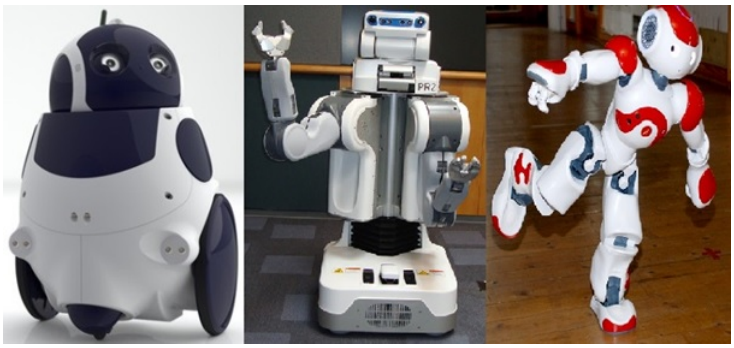

Figure 4.3: Exemples de robots compatibles avec *ROS*

4.3.1.1 Historique de ROS

ROS a été initialement développé en 2007 sous le nom *switchyard* par le *Stanford Artificial Intelligence Laboratory* dans le cadre du projet Stanford AI Robot STAIR (*STanford AI Robot*).
De 2008 à 2013, le développement a été effectué principalement par *Willow Garage*, un institut/incubateur de recherche en robotique. Pendant ce temps, des chercheurs de plus de vingt institutions ont collaboré avec les ingénieurs de *Willow Garage* dans un modèle de développement fédéré. En février 2013, le développement de *ROS* est poursuivi par *l'Open Source Foundation Robotics*.

4.3.1.2 Architecture de ROS

Niveau computationnel (informatique)
Ce sont les éléments de *ROS* qui facilitent plusieurs fonctionnalités [27].

- *Un nœud:* est une instance d'un exécutable. Il peut correspondre à un capteur, un moteur, un algorithme de traitement, etc. Pour faire simple, les nœuds sont les bouts de programmes simples que nous créons et qui, une fois lancés ensemble, réalisent une tâche complexe.

- *Un master:* est un service de déclaration et d'enregistrement des nœuds qui permet ainsi aux nœuds de se connaitre et d'échanger de l'information essentiellement via des topics ou des services.

- *Un topic:* est un système de transport de l'information basé sur le système *abonnement/publication (subscribe/publish)*; un ou plusieurs nœuds pourront publier de l'information sur un topic, tandis que d'autres pourront la lire. C'est donc un système de communication asynchrone *Many-to-Many*. Les topics ont la particularité d'être fortement typé ; l'information publiée (le message) est toujours structurée de la même manière.

- *Un message:* est une structure de donnée composite, composée de donnée de types primitifs (chaines de caractère, booléens, entiers, flottants, etc).

- *Un service:* permet une communication synchrone entre les nœuds. Ainsi, l'échange de l'information s'effectue soit de manière asynchrone via un topic ou de manière synchrone via un service.

Niveau du système de fichiers :
Ce sont des outils *ROS* pour gérer les codes sources, les instructions de compilation, et la définition d'un message, on peut citer :

- *Packages:* est l'unité principale d'organisation logicielle de *ROS*. Un package est un répertoire qui contient principalement plusieurs nœuds et des bibliothèques externes.

- *Manifest:* Manifest fournit des métadonnées sur un package " *Manifest.xml*'.

- *Stack:* une stack est définie simplement comme une collection de packages ce qui permet de regrouper les packages répondant à une même tâche principale de la robotique. Ainsi, on retrouve par exemple une stack dédiée à la navigation, à la localisation, etc.

- *Stack Manifests:* fournit des informations sur la stack (*stack.xml*).

- *Message types:* une description d'un message qui définit la structure de donné pour les messages envoyés en *ROS*.

- *Service types:* une description du service qui définit la structure de donné pour la requête et la réponse des services en *ROS*.

4.3.2 Outils logiciels pour la Kinect

En 2010, *PrimeSense*, l'entreprise ayant réalisé le capteur de profondeur utilisé par *la Kinect* a publié *OpenNI* qui est un *framework* open source permettant de développer des applications utilisant des interactions naturelles (voix, mouvements du corps, ..etc.). Ce *framework* est accompagné d'un driver open source pour la Kinect et d'un *middleware* propriétaire appelé *NITE* qui s'occupe à proprement parler du traitement des images de profondeur.
Le framework *OpenNI* est conçu pour être indépendant des capteurs utilisés et des algorithmes de traitement. En pratique, cependant, seules les caméras de profondeur basées sur la technologie de *PrimeSense* sont supportées et il n'existe toujours que le *middleware* propriétaire *NITE* pour utiliser les fonctions avancées de traitement.

4.4 Les différentes tâches effectuées au niveau du CDTA

Au cours de notre projet de fin d'étude qui s'est déroulé au niveau du *CDTA*, nous avons effectué de très nombreuse tâche qui nous ont permis d'acquérir certaines connaissances, de perfectionner d'autre et d'accomplir notre travail d'étude, tout d'abord nous avons commencé par effectuer certains travaux préliminaires indispensables à la réussite de notre projet, ensuite nous somme passés aux tâches principales.

4.4.1 Les tâches préliminaires

- Installation du système d'exploitation *Ubuntu* 14.04.03 en *dual boot* avec *Windows* sur nos machines (ordinateur portable).

- Installation du système d'exploitation pour robot *ROS Indigo* sous *Ubuntu* et se familiariser avec son environnement, ces notions de base et ces outils.

- L'apprentissage de quelque langages et notions de programmation (C++, python, l'orienté objet, etc) indispensables pour l'utilisation du *ROS* et l'interaction avec le robot.

- Étude du matériel utilisé (partie hardware) : fonctionnement de *la Kinect* et du robot *b21r* (les différents composants du robot, type de communication, etc).

- Installation des différents pilotes de *la Kinect* qui prennent en charge l'acquisition de donnée depuis la caméra, la communication et l'interaction avec le système *ROS*.

- L'acquisition de donnée à partir de la Kinect : image de profondeur, image RGB (rouge, vert, bleu), image monochrome.

4.4.2 Les tâches principales

4.4.2.1 L'architecture prototype

L'architecture réalisée lors de notre travail (figure 4.4), s'inscrit dans un processus de détection, suivi et reconnaissance. Un cycle complet de l'application commence par la capture de l'image/vidéo par la camera (Kinect) du robot B21r et se poursuit par la transmission de l'image à l'ordinateur via le driver de la Kinect. Ensuite les étapes de traitement et d'implémentation d'algorithme sont effectuées, système d'exploitation *ROS*, afin de réaliser les différentes opérations de détection, suivi et reconnaissance.

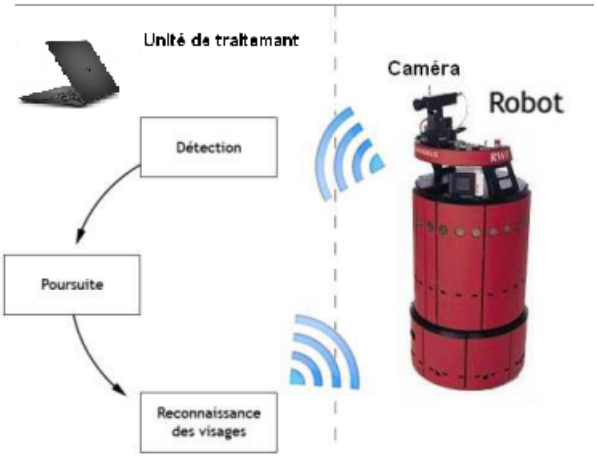

Figure 4.4: Architecture prototype.

4.4.2.2 Implémentation des algorithmes

Les différents algorithmes implémentés et utilisés dans notre processus de détection, suivi et reconnaissance sont les suivant :

- Détection $2D$ des personnes par la méthode du HOG (Histogramme des Gradients Orientés).
- Détection faciale $2D$ par la méthode Viola et Jones.
- Détection $2D$ des personnes et de leur visages (simultanément) en combinant les deux méthodes de détection $2D$ précédentes.
- Détection et suivi $3D$ du squelette humain.
- Détection, suivi 3D, reconnaissance facial des personnes.

4.4.2.3 Evaluation par des tests et des scénarios

Les algorithmes implémentés sont évalués en temps réel par des tests et scénarios pour juger leur performance en termes de robustesse et de précision à partir des séquences vidéos et images acquissent depuis le capteur Kinect. Les algorithmes seront jugés par des résultats donnés par ces derniers face à des contraintes liées à l'environnement : éclairage de la scène, réflexion de la lumière et des contraintes liées au personnes : orientations et mouvement du corps, nombre de personnes présentes dans la scène, variations de pose et expression du visage, présence d'accessoires sur la personne (lunettes et casquette).

4.5 Détection du corps humain en 2D

4.5.1 Cas d'une seule personne

4.5.1.1 Scénario 1

Le robot b21r observe la scène devant lui en utilisant les données fournies par la Kinect, une personne est en face du robot dans deux conditions différentes de luminosité, voir figure 4.5.

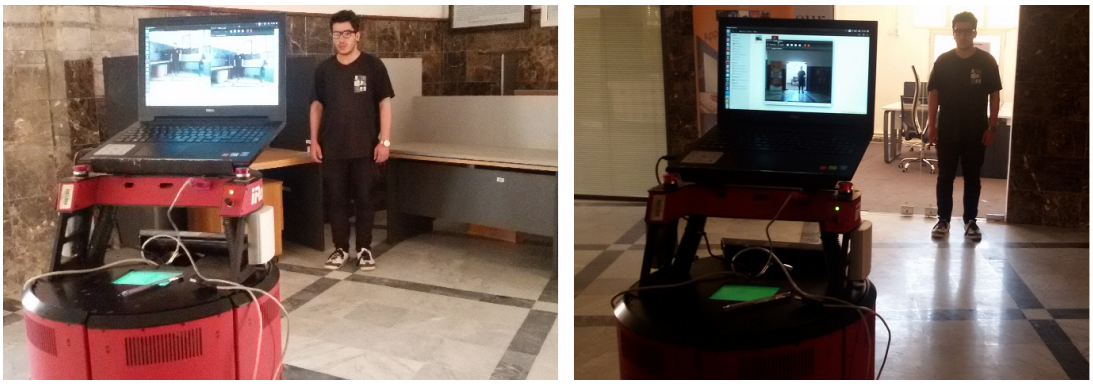

Figure 4.5: Scénario 1: scènes a et b.

Figure 4.6: Scénario 1: la scène du point de vue robot.

Commentaire:
Dans ce scénario, on a deux scènes (Figure 4.5 *a* et *b*) dans des conditions de luminosité différentes la fig 4.6 représente la scène *a* (resp. b) du point de vue robot, c'est-à-dire comment le robot perçoit la scène. On remarque bien que la personne est entièrement détectée en $2D$ (encadrement par un rectangle bleu du corps tout entiers), en plus l'algorithme a donné d'excellents résultats en dépit des variations de l'illumination sur le corps (la première scène est plus éclairée par rapport à la deuxième).

4.5.1.2 Scénario 2

Le robot b21r observe la scène devant lui en utilisant les données fournies par la Kinect, une personne assise est en face du robot.

Figure 4.7: A gauche la scène réel. A droit la scène du point de vue robot.

Commentaire:
On voit bien dans la figure 4.7, que la personne assise est entièrement détectée (encadrée par un rectangle bleu), malgré le changement de pose du corps, ce qui montre la robustesse de la méthode vis à vis les variations de pose du corps humain.

4.5.1.3 Scénario 3

Le robot b21r observe la scène devant lui en utilisant les données fournies par la Kinect, une personne est en face du robot change sa pose, de face, de côté, de dos. voir figure 4.8.

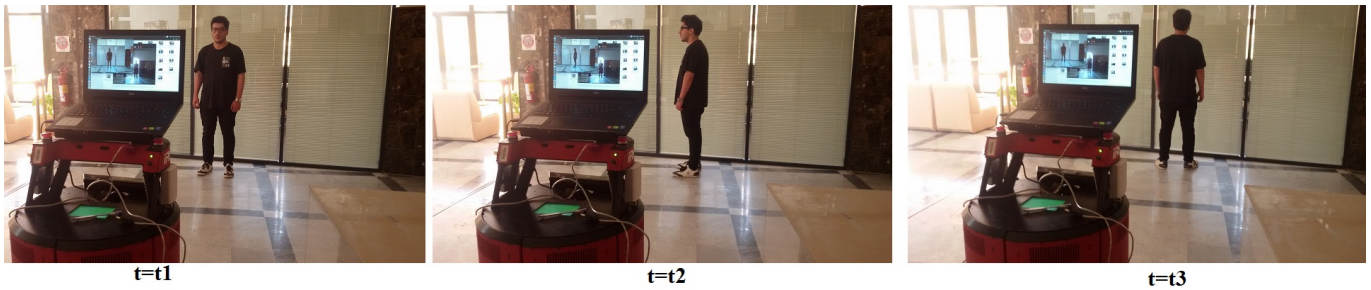

Figure 4.8: la scène réelle.

Commentaire:
Cette fois ci l'algorithme de détection $2D$ sera confronté à un autre défi, on voit dans la figure 4.8 la scène réelle dans différentes cas de l'orientation de la personne, à t_1 la personne est face au robot, à t_2 elle est de coté et à t_3 elle est de dos. On remarque dans la figure 4.9

Figure 4.9: la scène du point de vue robot.

que malgré ces changements, la personne est entièrement détectée (encadrée par un rectangle bleu), ce qui prouve l'efficacité de l'algorithme dans le cas des variations de l'orientation du corps humain.

4.5.1.4 Scénario 4

Le robot b21r observe une personne qui se déplace dans la scène d'un point de départ correspondant à un point d'arrivée.

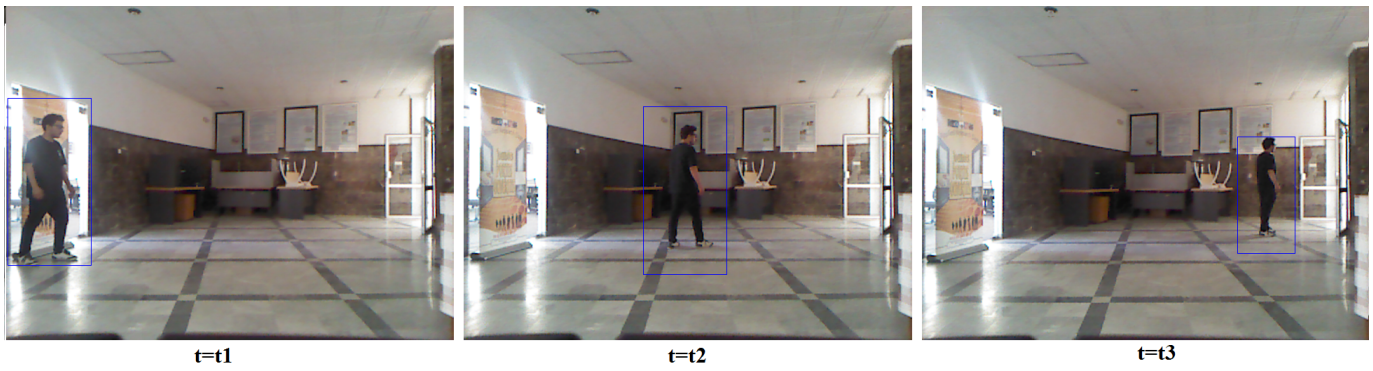

Figure 4.10: la scène du point de vue robot.

Commentaire:
La personne se déplace d'un point correspondant à un instant t_1 vers un autre point correspondant à l'instant t_3 (figure 4.10), au cours de ce déplacement, la personne est entièrement détectée en $2D$ (encadrement par un rectangle du corps tout entier). Ainsi l'algorithme donne d'excellents résultats en dépit des variations de la vitesse du corps au cours du déplacement.

4.5.2 Cas de plusieurs personnes

4.5.2.1 Scénario 1

Le robot b21r observe la scène devant lui en utilisant les données fournies par la Kinect, trois personnes sont en face du robot (à gauche de la figure 4.11).

Figure 4.11: A gauche la scène réel. A droit la scène du point de vue robot.

Commentaire:
Dans ce scénario La méthode de détection utilisée va être confronté à un autre enjeu qui est la présence de plusieurs personnes dans la scène. On remarque bien, à travers la figure 4.11, que les trois personnes sont parfaitement détectées (encadrement par un rectangle du corps tout entier), et l'algorithme s'est montré assez robuste vis à vis, du nombre de personnes, de la courte distance et du rapprochement entre eux.

4.6 Détection du visage en 2D

4.6.0.2 Scénario 1

Le robot b21r observe la scène devant lui en utilisant les données fournies par la Kinect, une personne assise sur un fauteuil devant le robot (voir figure 4.12) change à chaque fois, l'orientation de sa tète, les expressions de son visage et ses accessoires. et enfin un test de changement de la luminosité.

Figure 4.12: Scénario 1: la scène a.

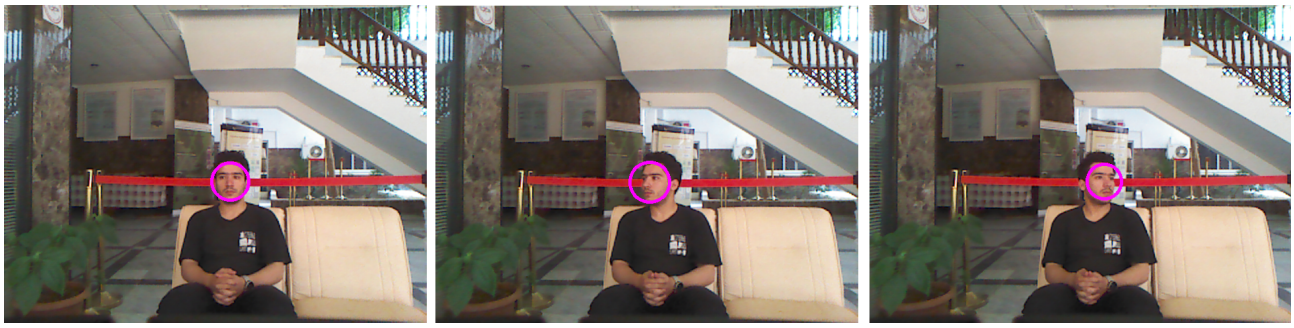

Figure 4.13: Changement de l'orientations de la tête.

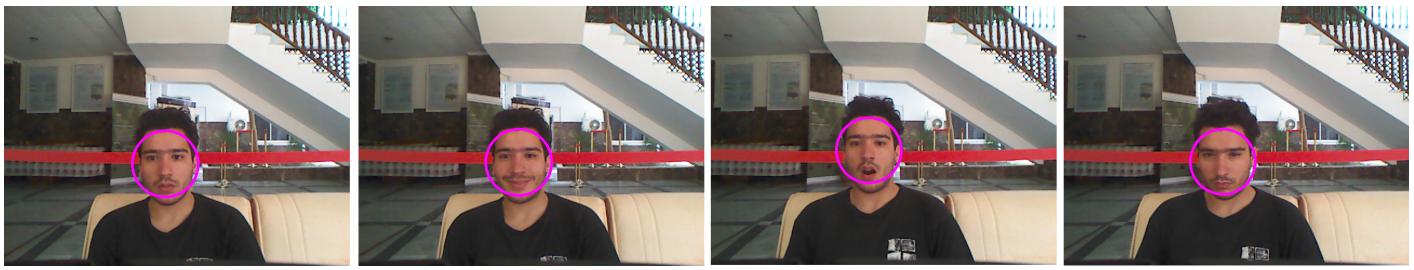

Figure 4.14: Changement d'expressions du visage.

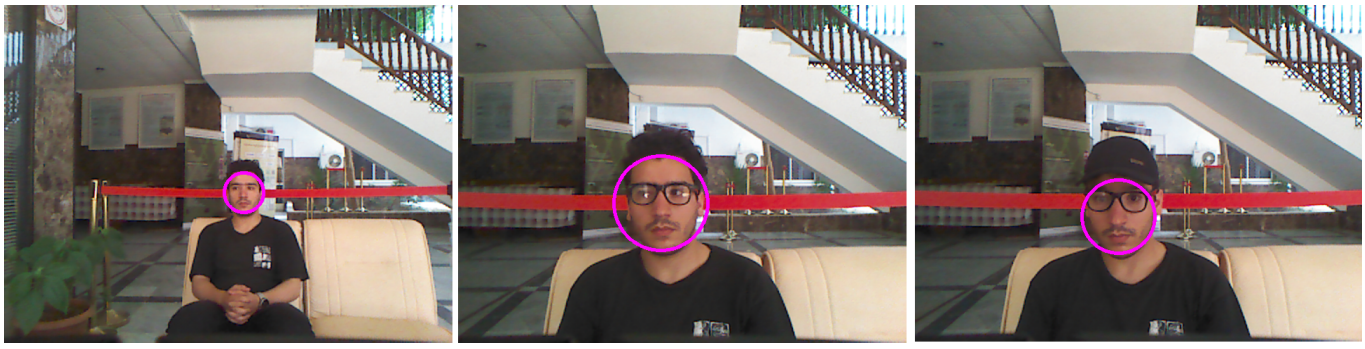

Figure 4.15: Changement d'accessoires.

Figure 4.16: Changement de la luminosité.

Commentaire:

La figure 4.13 montre que lors du déplacement de la tête de la personne dans différentes orientations le visage a été parfaitement détecté (cercle sur le visage). En regardant la figure 4.14 et la figure 4.15 on constate qu'en dépit des variations des expressions du visage, et le présence d'un ou plusieurs d'accessoires (les lunettes et la casquette), la méthode de utilisée reste robuste et le visage est bien détecté, et cela même dans le cas du changement de luminosité (voir figure 4.16).

4.7 Détection du corps et du visage humain simultanément

4.7.0.3 Scénario 1

Le robot b21r observe la scène devant lui en utilisant les données fournies par la Kinect, , une personne se rapproche du robot et se déplace d'un point de départ vers un autre point d'arrivée, (figure 4.17).

Figure 4.17: Scénario 1: La scène a.

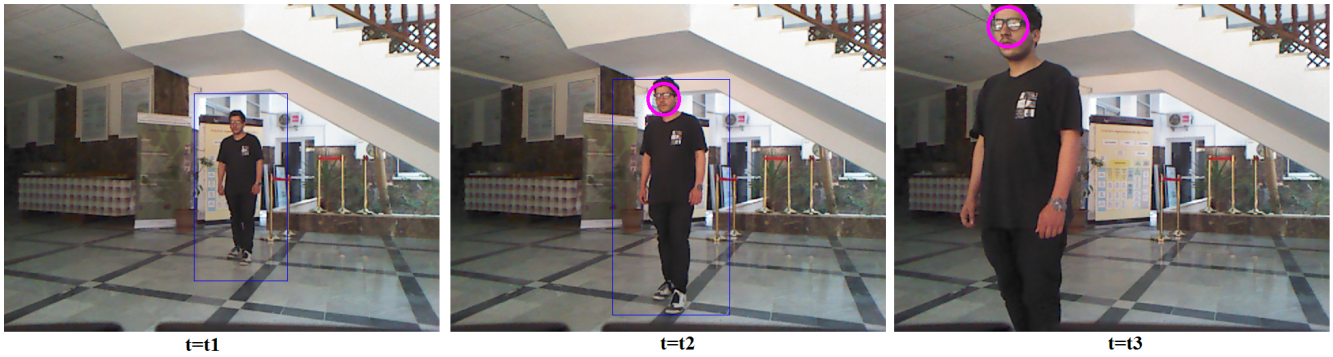

Figure 4.18: Scénario 1: la scène du point de vue robot.

Commentaire:

Dans ce scénario on a essayé de faire fonctionner les deux algorithmes simultanément, afin de faire une comparaison entre les deux. On remarque à travers la figure 4.18, à l'instant t_1 où la personne est loin du robot, que seulement la détection du corps c.-à-d. *HOG* fonctionne, et à l'instant t_2 les deux méthodes fonctionnent (détection du corps entier et détection du visage), à l'instant t3 la personne est plus proche au robot, seulement la détection du visage fonctionne, le corps n'est pas détecté.

4.8 Détection et suivi 3D du squelette humain

La figure 4.19 représente le modèle des 15 articulations utilisées dans les tests.

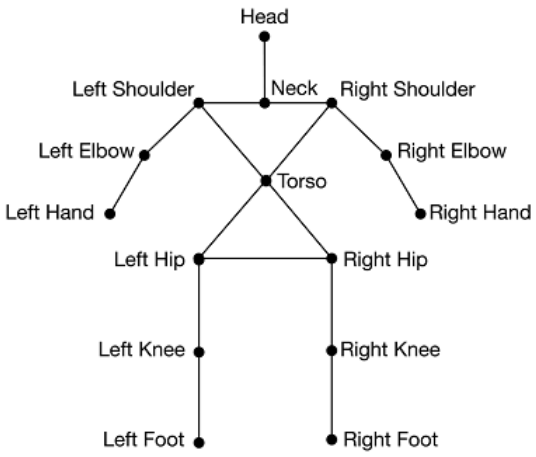

Figure 4.19: Modèle des 15 articulations utilisées dans les tests.

4.8.1 Cas d'une seule personne

4.8.1.1 Scénario 1

La Kinect observe la scène, une personne se déplace d'un point de départ (milieu de la scène (figure 4.20)) vers un autre point d'arrivée (côté droit de la scène (figure 4.21)).

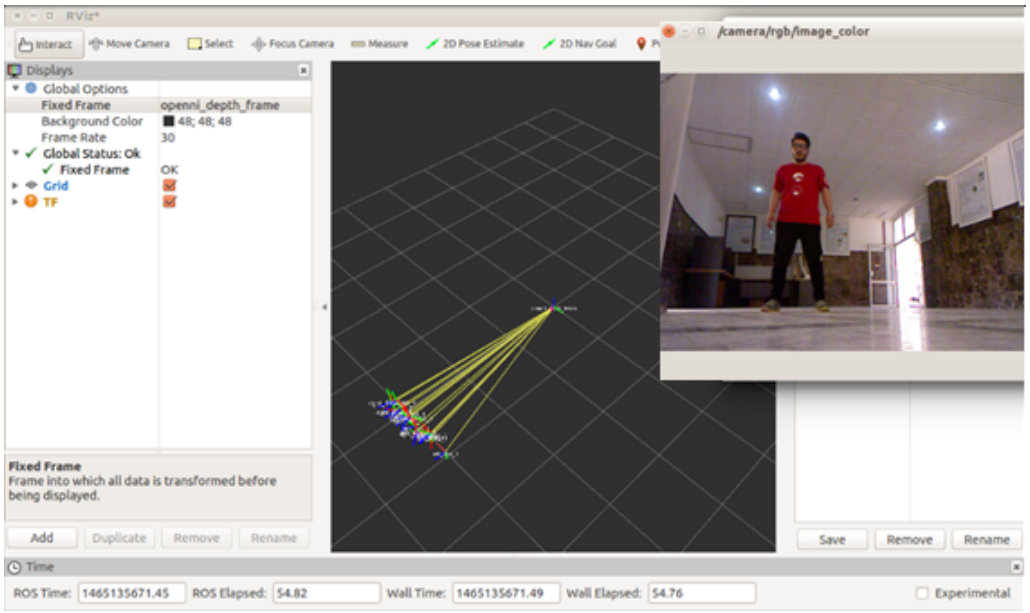

Figure 4.20: Représentation 3D des articulations dans Rviz en temps réel à l'instant t_1.

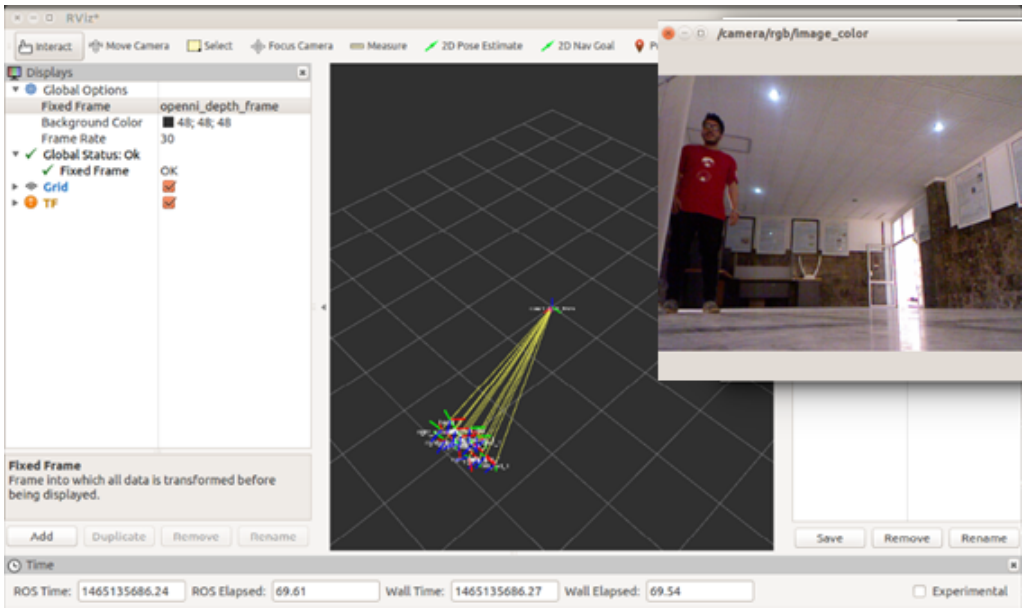

Figure 4.21: Représentation 3D des articulations dans Rviz en temps réel à l'instant t_2.

Commentaire:

Rviz nous a permit de visualiser les résultas de la detection et du suivi $3D$ des 15 articulation d'une personne (Modèle du squelette (figure 4.19)) qui se déplace d'un point de départ correspondant à l'instant t_1 (milieu de la scène (figure 4.20)) vers un autre point correspondant à l'instant t_2 (côté droit de la scène (figure 4.21)), le repère de la Kinect se trouve au milieu de la carte de Rviz (figure 4.20) et pour chaque articulation un repère va être affecté pour pouvoir connaitre l'orientation des articulation, les points des articulations sont reliés à l'origine du repère de la Kinect par des lignes jaunes dans la carte Rviz.

4.8.1.2 Scénario 2

La Kinect à une hauteur d'environ 1.5 *mètre observe la scène, une personne est en mouvement dans la scène (figure 4.23), la position 3D de la tête de la personne est acquise par rapport au repère de la Kinect (figure 4.22).* **Commentaire:**

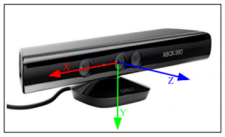

Figure 4.22: Repère de la Kinect.

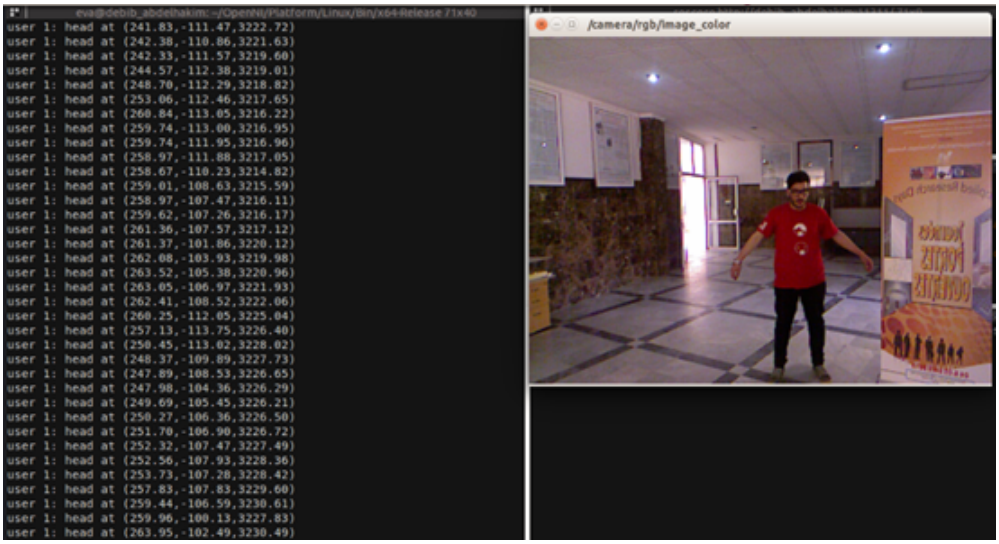

Figure 4.23: Position 3D de la tête en temps réel de la personne.

Dans ce test la Kinect se trouve à une hauteur de 1.5 m du sol, l'algorithme de détection et suivi du squelette permet de faire l'acquisition des positions des 15 points d'articulations en $3D$ par rapport au repère de la Kinect (figure 4.22), ce test a pour but de donner un exemple sur le positionnement d'une articulation, dans notre cas on prend celle de la tête. La personne est en mouvement dans la scène à un instant bien déterminé, les résultats de

positionnement de la tête sont acquise (figure 4.23) on remarque que la variation varie faiblement (variation de l'ordre du millimètre) ceci est dû au fait que la vitesse d'acquisition des positions est très supérieur à la vitesse de déplacement de la personne , au dernier instant, l'articulation se trouve à une distance selon l'axe z d'environ 3230.49 mm, une distance selon l'axe x d'environ 263.95 mm et une distance de -102.49 mm.

4.8.1.3 Scénario 3

Le robot b21r observe la scène devant lui en utilisant les données fournies par la Kinect, une personne est en mouvement dans la scène d'un point de départ correspondant à l'instant t_1 *(acquisition de l'image de profondeur(figure 4.24)) vers d'autre points successif correspondant à l'instant* t_2 *(silhouette de la personne (figure 4.25)), l'instant* t_3 *(construction du squelette de la personne (figure 4.26)), l'instant* t_4 *(personne du côté gauche de la scène (figure 4.27)), l'instant* t_5 *(personne du côté droit de la scène (figure 4.28)) et enfin l'instant* t_6 *(personne dos à la Kinect (figure 4.29)).*

Figure 4.24: A gauche image couleur (RGB), à droite image profondeur (carte de profondeur), à un instant t_1.

Figure 4.25: A gauche image couleur (RGB), à droite silhouette de la personne dans l'image de profondeur (carte de profondeur), instant t_2.

Figure 4.26: A gauche image couleur (RGB), à droite silhouette et squelette de la personne dans l'image de profondeur, à un instant t_3.

Figure 4.27: A gauche image couleur (RGB), à droite silhouette et squelette de la personne dans l'image de profondeur à un instant t_4.

Figure 4.28: A gauche image couleur (RGB), à droite silhouette et squelette de la personne dans l'image de profondeur à un instant t_5.

Figure 4.29: A gauche image couleur (RGB), à droite silhouette et squelette de la personne dans l'image de profondeur à un instant t_6.

Commentaire:

à l'instant t_1 (figure 4.24) l'image de profondeur de la scène est acquise par la Kinect, on distingue clairement grâce à la différence de profondeur, la silhouette de la personne se tenant au milieu de la scène, à l'instant t_2 la silhouette de la personne est détectée (silhouette en bleu figure 4.25) et donc séparée du reste de la scène (séparation de l'utilisateur de arrière-plan), à l'instant t_3 (figure 4.26) les articulations sont détecter et le squelette est constitué(on utilise seulement un modèle de 15 articulation (figure 4.19)) , on notera que l'opération depuis l'instant t_1 en passant par l'instant t_2 et enfin l'instant t_3 ne dépasse pas généralement les 4 seconds. Après la détection le processus du suivi commence du point correspondant à l'instant t_3 (ou la personne est au milieu de la scène (figure 4.26)) vers le point correspondant à l'instant t_4 (ou la personne est à gauche de la scène (figure 4.27)) ensuite au point correspondant à l'instant t_5 (ou la personne est à droite de la scène (figure 4.28)) et enfin le point correspondant à l'instant t_6 (ou la personne est à droite de la scène dos à la camera (figure 4.29)), au cours de cet déplacement on a pu non seulement faire le suivie de la personne (le corps tout entier mais aussi) mais aussi les 15 articulation choisit, on notera tout de même certains remarques suivantes :

La qualité de la détection et du suivi du squelette se dégrade au niveau des articulation qui sont soit dans les extrémités du champ de vision de la Kinect ou bien en dehors, et ceci influe même sur les autres articulations adjacentes, par exemple au niveau des figures 4.26 et 4.27, l'articulation de la tête de la personne n'est pas détecté (la tête est partialement en dehors du champ de vision de la Kinect), il faut attendre jusqu'à l'instant t_4 pour que la tète soit entièrement visible et que l'articulation soit détecter et suivi (figure 4.27), de plus on constate au niveau des même figure (4.26 et 4.27) des erreurs de construction du squelette au niveau des jambes et ceci est dû au fait que les articulation du bas des jambes sont soit en dehors ou bien dans les extrémité du champ de vision de la camera.

Une autre contrainte qui cause la dégradation du processus de détection et de suivi est la pose que le corps humain prend lors du mouvement, on remarque clairement ceci en comparant

la figure 4.28 ou la personne est face à la caméra, les jambes et les bras un peu écarter du reste du corps (dans ce cas les résultats son excellent) et la figure 4.29 ou la personne se tiens dos à la camera et les bras sont collées au reste du corps (dans ce cas les résultats sont assez bonne).
On constate que malgré les contraintes et les erreurs cité précédemment, la méthode de détection et du suivi du squelette humain donne d'excellent résultat si on se place dans des condition favorable (la personne est entièrement visible).

4.8.2 Cas de plusieurs personnes

4.8.2.1 Scénario 1

Le robot b21r observe la scène devant lui en utilisant les données fournies par la Kinect, deux personnes sont en mouvement dans la scène d'un point de départ (image de profondeur (figure 4.30)) vers d'autre points successifs.

Figure 4.30: A gauche image couleur (RGB), à droite image profondeur (carte de profondeur), à un instant t_1.

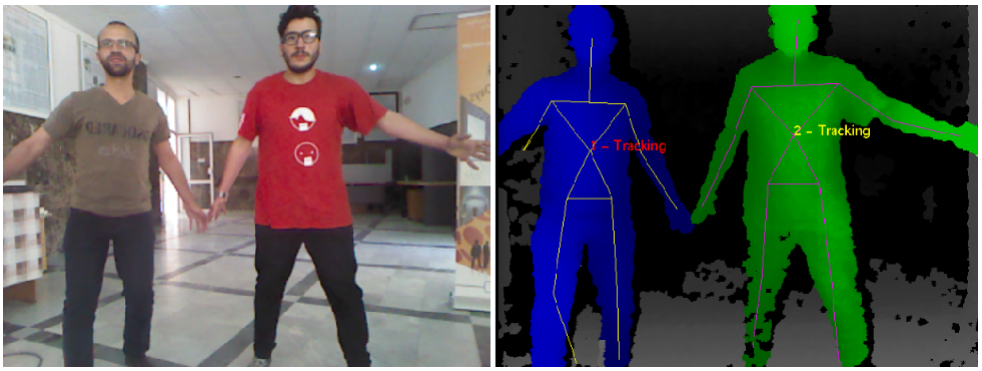

Figure 4.31: A gauche image couleur (RGB), à droite silhouette et squelette de plusieurs personnes dans l'image de profondeur, à un instant t_2.

Commentaire:
Cette fois ci deux personne seront simultanément présent et en mouvement dans la scène,

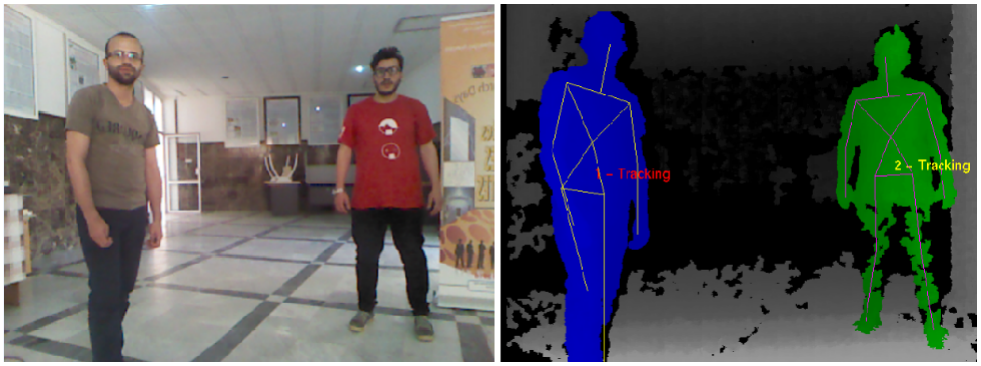

Figure 4.32: A gauche image couleur (RGB), à droite silhouette et squelette de plusieurs personnes dans l'image de profondeur, à un instant t_3.

Figure 4.33: A gauche image couleur (RGB), à droite silhouette et squelette de plusieurs personnes dans l'image de profondeur, à un instant t_4.

après l'acquisition de l'image en profondeur par la Kinect à l'instant t_1 (figure 4.30) la silhouette des deux personnes est détectée ensuite à l'instant à t_2 les 15 articulation de chaque personne seront détectée et leur modèle de squelette sera constituer (figure 4.31), on notera qu'en raison des différences de taille (la forme des deux personne n'est pas la même), de la distance (par rapport à la Kinect) et de l'influence de l'environnement sur chacun, la détection de la silhouette et des articulation ne se fait pas en même temps pour les deux personne (petit décalage dans le temps dans le processus de détection de une personne par rapport à l'autre). Les deux personnes se déplacent chacun depuis un point correspondant à l'instant t_2 (figure 4.31) vers un point correspondant à l'instant t_3 (chacun dans un coté da la scène (figure 4.32)), hormis les erreurs et contraintes constater dans le test précédant (cas d'une seule personne, section précédant), l'algorithme de détection et suivi lors du déplacement des deux personnes a donnée de bons résultats malgré le rapprochement entre les deux personnes, la variation de illumination et du mouvement de chaque personne. On remarque que à l'instant t_4 au dépit que la première personne dissimule partialement l'autre, les squelettes des deux personnes ont été parfaitement suivi et ceci est dû à la prédiction et à l'estimation des positions actuel des articulations dissimuler de la deuxième personne à partir de leur position précédente (figure 4.33).

4.9 Détection, suivi 3D et reconnaissance faciale des personnes

4.9.1 Scénario 1

Le robot b21r observe une personne qui se déplace dans la scène, au départ la personne est identifiée comme étant un inconnu (elle n'est pas dans la base de données (figure 4.34)), ensuite après avoir été ajouter à la base de données la personne sera identifier avec son prénom (figure 4.35), et enfin le visage de la personne sera entièrement dissimulé (figure 4.36).

Commentaire:

au dépût du test (figure 4.34) le visage de la personne est parfaitement détecter (engadrement par un carré, après une comparaison avec la base d'apprentisage (base de donnés) , aucune identité n'est trouvée dans la base et la personne sera classer comme un inconnu (figure 4.34), aprrés on prend un échantillon ou bien plusieurs pour amméliorer la précision de la reconnaissance de la personne (des image du visage de la personne) on les ajoute à notre base de donnée, on refait le test précedant , le opération de comparaison s'effectuer et enfin la personne sera identifei par son nom sous le cadre de détection (figure 4.35).

En plus de l'opération de la reconnaissance et dedection l'algorithme parmet de faire le suivi, en faisant la prédiction de la position et de l'emplacement du visage en utilisant les états

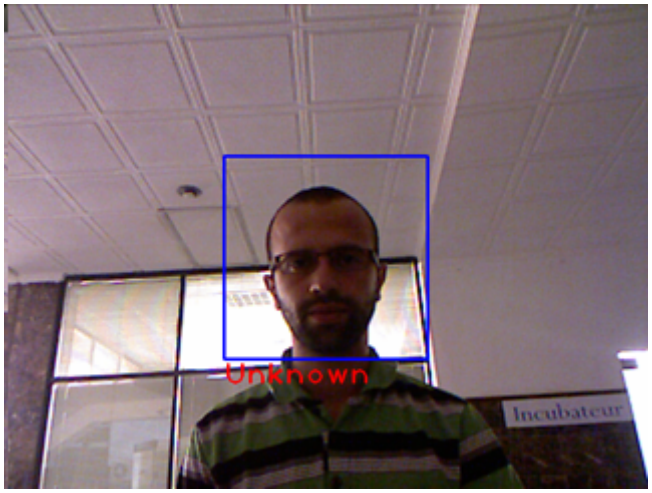

Figure 4.34: Identification d'une personne comme étant inconnue.

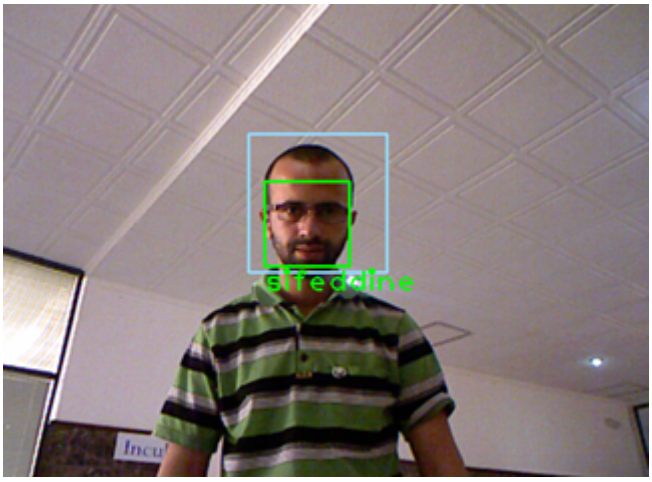

Figure 4.35: Identification d'une personne comme étant connue (identifiée par son prénom).

Figure 4.36: Dissimulation du visage de la personne par un objet.

précedante, cela est visible dans le test de la figure 4.36 dans lequel on dissimule entièrement le visage de la personne, on constate que la personne est toujours reconnue par son identité malgré l'absence du visage sur l'image acquise par la Kinect.

4.10 Conclusion

Au cours de chapitre, après avoir fait une brève présentation du matériel (hardware) et les logiciels (software) utilisés, nous avons fait l'implémentation des algorithmes de détection, suivi et reconnaissance, et afin juger leurs performances en termes de précision et de robustesse nous avons effectué des tests qui confrontent ces algorithmes face aux contraintes liées à l'environnement dans laquelle évolue le robot (l'illumination de la scène et la réflexion de la lumière) et les contraintes liées aux personnes (variation d'orientation du corps, changement de pose et d'expression , mouvement de la personne, nombre de personne présents dans la scène, présence ou absence d'accessoires sur le visage). Nous avons commencé par deux algorithmes de détection 2D l'un pour le visage (entourer le visage par un cercle) et l'autre pour tout le corps (encadrement corps tout entier par un rectangle bleu) ensuite nous avons effectué une combinaison entre les deux pour effectuer un autre test qui fait la détection simultanée du visage et du corps, ensuite on a implémenté l'algorithme de détection et de suivi 3D du squelette humain et enfin un autre algorithme qui permet de faire la reconnaissance (l'identification de la personne par son nom),et le suivi 3D visuel.
On notera que les différents tests et scénarios étaient déterminants pour pouvoir porter un jugement sur l'efficacité de chaque algorithme et la sensibilité de ces derniers face à la dégradation des conditions liées à l'environnement et à la personne.

Conclusion générale et perspectives

La robotique mobile de service est devenue depuis quelques années un enjeu fondamental et stratégique pour de nombreux pays industriels, parmi les machines comptabilisées actuellement comme robots de service, la plupart sont conçues pour des tâches spécialisées et ont une autonomie limitée dans un contexte restreint d'utilisation. Pour l'avenir, un nombre croissant de recherches sont menées pour insérer dans des milieux ouverts des robots dotés d'une autonomie plus importante, de capacités d'interaction avec les humains plus élaborées, de capacités d'apprentissage accrues, ce qui ouvre à des nouveaux perspectives d'usages.
Ce mémoire s'inscrit dans le cadre du projet du Centre de Développement des Technologies Avancées (CDTA) visant à doter la plateforme robotique expérimentale "le robot B21r" de moyens lui permettant une navigation sure et autonome dans un milieu urbain encombré par des personnes.
Dans le cadre de ce mémoire, nous avons traité le problème de la détection, suivi et reconnaissance de l'être humain pour un robot de service (un robot guide) en utilisant l'information visuelle acquise à partir du capteur Kinect et prévenante de l'environnement qui l'entoure, afin de lui permettre de mieux comprendre la scène dans laquelle il se déplace.
Pour ce faire, nous avons débuté notre mémoire par un bref rappel sur la robotique mobile, ensuite nous avons traité l'aspect théorique et méthodologique des différents algorithmes utilisés, après cela commence le travail d'implémentation des algorithmes de détection, suivi et reconnaissance, nous avons tous d'abord fait l'implémentation de deux méthodes de détection $2D$ celle du corps (Histogramme des Gradients Orientés), et celle du visage (Viola et Jones), après nous les avons testé simultanément afin de réaliser en même temps la détection du visage et du corps, ensuite nous avons implémenté l'algorithme de détection et suivi 3D du squelette humain, et enfin on a fait une implémentation d'un algorithme qui permet de faire la reconnaissance faciale (identification) et le suivi grâce au filtre de Kalman et le suivi du squelette.
Afin de juger les performances des algorithmes implémentés en termes de précision et de robustesse nous avons effectués des tests qui confrontent ces algorithmes face au contraintes liées à l'environnement dans lequel évolue le robot (l'illumination de la scène, réflexion de la lumière) et les contraintes liées aux personnes (variation d'orientation du corps, changement de pose et d'expression , mouvement de la personne, nombre de personnes présentes dans la

scène, présence ou absence d'accessoires sur le visage).
D'après cette expérience acquise à travers ce projet de fin d'étude, et à travers les résultats des tests effectués, il nous semble intéressant de continuer ce travail en traitant par la suite les thèmes suivants :

- L'utilisation d'une technique de détection beaucoup plus robuste basée sur la profondeur qui est *HOD " Histogram of Oriented Depth"* et la combiner avec la méthode *HOG "Histogram of Oriented Gradient"*.

- L'utilisation de la méthode de suivi et reconnaissance, qui permet non seulement d'identifier la personne mais aussi de déterminer sa position $3D$ à chaque instant, dans un asservissement visuel ce qui permettra au robot mobile d'interagir (suivi, guidage) avec la personne en fonction de son identité.

- Utiliser les techniques de détection-suivi-reconnaissance implémentées dans notre projet comme point de départ pour le développement d'une méthode qui prend en considération les groupe de gens et ne considère pas les personnes comme étant des individus complétement indépendants, cette dernière est très importantes pour la navigation sociale.

Bibliographie

[1] A. Ganesh Z. Zhou H. Mobahi A. Wagner, J. Wright and Y. Ma. Toward a practical face recognition system: Robust alignment and illumination by sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[2] Gul A.B. Holostic face recognition by dimension reduction. Technical report, School of Natural and Applied Sciences of the Middle East Technical University, 2003.

[3] Dora Luz Almanza-Ojedar. *Détection et suivi d'objets mobiles perçus depuis un capteur visuel embarqué*. PhD thesis, Université Paul Sabatier - Toulouse III, 2011.

[4] Y. Amit and D. Geman. Shape quantization and recognition with randomized trees. *Neural Computation*, 1997.

[5] A. L. C. Barzak. A. l. c. barzak, to ward and efficient implementation of a rotation invariant detection using haar-like features. *international conference on Intelligent robotsand systems*, 2005.

[6] Serge KOMANDA BASEMA. Identification des personnes par reconnaissance de visage pour la sécurité d'une institution bancaire. Technical report, Institut supérieur pédagogique de Bukavu, 2010.

[7] Kriegman D Belhumeur P., Hespanha J. Eigenfaces vs. fisherfaces : Recognition using class specific linear projection. *the Fourth European Conference on Computer Vision*, 1996.

[8] Valstar Bihan Jiang, Michel F and Maja Pantic. Facial action detection using block-based pyramid appearance descriptors. Technical report, Chalmers University of Technology, 2005.

[9] P. BONNET. Cours de traitement d'image. Technical report, USTL, 2013.

[10] Khefif Bouchra. Mise au p oint d'une application de reconnaissance faciale. Technical report, Université Ab ou Bakr Belkaid – Tlemcen, 2013.

[11] Prof Dj. Boukhetala. élément de cous identification des processus industrielles. Technical report, Ecole Nationale Polytechnique, 2015.

[12] L. Breiman. Random forests. Technical report, University of California, Berkeley, 2001.

[13] A. Schulz C. Wojek, G. Dorko and B. Schiele. Sliding-windows for rapid object class localization: a parallel technique. *Pattern Recognition (DAGM)*, 2008.

[14] D. Comaniciu and P. Meer. Meer. mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002.

[15] Navneet DALAL. Histograms oforiented gradients for human detection. *Conference on Computer Vision and Pattern Recognition*, 2005.

[16] Navneet DALAL. *Finding People in Images and Videos*. PhD thesis, L'INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE, 2006.

[17] D.Alazard. Introduction au filtre de kalman. Technical report, SUPAERO, 2006.

[18] Grgic S Delac K., Grgic M. Independent comparative study of pca, ica, and lda on the feret data set. Technical report, University of Zagreb, 2004.

[19] K. Fukunaga et L. D. Hostetler. Mean shift analysis and applications. *Computer Vision and Pattern Recognition*, 1975.

[20] M. Kolsch et M. Turk. Analysis of rotational robustness of hand detection with a viola-jones detector. *ICPR*, 2004.

[21] Paul Viola et Michael Jones. Rapid object detection using a boosted cascade of simple features. *IEEE CVPR*, 2001.

[22] D. Comaniciu et P. Meer. Object class recognition by unsupervised scale-invariant learning. *IEEE International Conference on Computer Vision (ICCV'99)*, 1999.

[23] D. Comaniciu et P. Meer. A robust approach toward feature space analysis. *EEE Transaction on Pattern Analysis and Machine Intelligence*, 2002.

[24] M. Jones et P. Viola. Fast multi-view face detection. *IEEE CVPR*, 2003.

[25] Sadek Amrouche et Said OUALA. Détection des objets mobiles et prédiction de leur mouvement dans un environnement dynamique. Technical report, Ecole Nationale Polytechnique, 2015.

[26] B. Triggs F. Moosmann and F. Jurie. Fast discriminative visual codebooks using randomized clustering forests. *Neural Information Processing Systems*, 2006.

[27] ROS foundation. What is ros?, 2016.

[28] Golub G.H. The generalized eigenvalue problem. Technical report, Istituto Guido Castelnuovo, 2004.

[29] J. Miao H. Liu, W. Gao and J. Li. A novel method to compensate variety of illumination in face detection. *Joint Conference on Information Sciences*, 2002.

[30] M. Johnson J. Shotton and R. Cipolla. Semantic texton forests for image categorization and segmentation. *Computer Vision and Pattern Recognition*, 2008.

[31] Luc JAULIN. Robotique mobile. Technical report, ENSI, 2014.

[32] A. Johnson Y. Cheng L. Matthies, M. Mainmone. Computer vision on mars. *International Journal of computer Vision*, march 2007.

[33] Peter S. Maybeck. Stochastic models, estimation, and control. Technical report, Academic Press, 1979.

[34] Nicolas MORETTE. *Contribution à la Navigation de robots mobiles : approche par modèle direct et commande prédictive*. PhD thesis, Université d'Orléans, 2009.

[35] Slimane NOUREDDINE. *Système de Localisation pour Robots Mobiles*. PhD thesis, Université de Batna, 2005.

[36] B. Qadeer. *Multisensor Data Fusion for Detection and Tracking of Moving Objects From a Dynamic Autonomous Vehicle*. PhD thesis, University of Grenoble, 2012.

[37] J. R. Quinlan. Induction of decision trees. *International Joint Conference on Artificial Intelligence*, 1983.

[38] P. Perona R. Fergus and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. *Computer Vision and Pattern Recognition*, 2003.

[39] Patrick RIVES. Navigation autonome des robots mobiles : des problèmes de modélisation, perception et contrôle. Technical report, INRIA, 2004.

[40] T. Sharp. Implementing decision trees and forests on a gpu. *European Conference on Computer Vision*, 2008.

[41] B. Shepherd. An appraisal of a decision tree approach to image classification. *International Joint Conference on Artificial Intelligence*, 1983.

[42] V. Nehra T. Goel and V. P. Vishwakarma. Comparative analysis of various illumination normalization techniques for face recognition. *International Journal of Computer Applications*, 2001.

[43] X. Tan and B. Triggs. Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE Transactions on Image Processing*, 2010.

[44] P. Lagger V. Lepetit and P. Fua. Randomized trees for real-time keypoint recognition. *Computer Vision and Pattern Recognition*, 2005.

[45] Vladimir Vapnik. The nature of statistical learning theory. *Springer*, 1995.

[46] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[47] Paul Viola and Michael Jones. Robust real-time object detection. *international work shop on statistical and computation atheories of vision, Vancouver, Canada*, July 13 2001.

[48] M. J. Er W. L. Chen and S. Q. Wu. Illumination compensation and normalization for robust face recognition using discrete cosine transform in logarithm domain. *IEEE Transactions*, 2002.

[49] A. Rosenfeld W. Zhao, R. Chellappa and P. Phillips. Face recognition : a literature survey. Technical report, University of Maryland, USA, October 2000.

[50] J. Winn and J. Shotton. The layout consistent random field for recognizing and segmenting partially occluded objects. *Computer Vision and Pattern Recognition*, 2006.

[51] Alper Yilmaz. Object tracking. *ACM Computing Surveys*, 2006.

[52] YIXIAO YUN. Analysis and classification of ob ject poses: Using visual / infrared images and feature fusion. Technical report, Department of Computing, Imperial College London, UK, Sweden 2011.

Annexe A

Le processus de la perception

Afin de bien expliquer le processus de la perception. Nous représentons en figure A.1 , les relations existantes entre plusieurs fonctions, regroupées dans cinq modules (dans les boîtes rectangulaires), et les données prises en entrée ou générées en sortie de ces modules (dans les formes elliptiques) [3] :

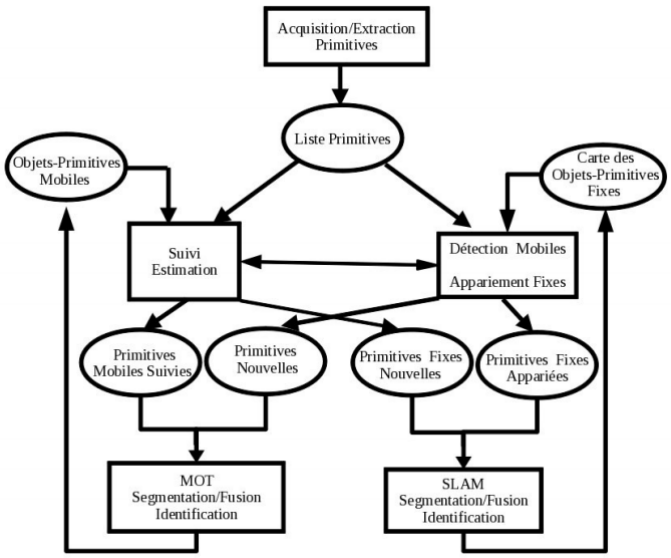

Figure A.1: Le processus de la perception d'après [3]

- *Acquisition-Extraction de primitives :* analyse des données sensorielles courantes Z_t acquises à l'instant t, et extraction de primitives. Cette fonction d'extraction peut être active : elle est alors guidée par les modèles courants et intégrée avec les procédures de suivi et de détection.

- *Suivi-Estimation des Objets Mobiles :* parmi les primitives extraites dans les données courantes Z_t, il s'agit de rechercher le sous-ensemble M_t des éléments qui correspondent ou appartiennent aux objets et primitives déjà perçus lors des acquisitions précédentes,

sur des composantes mobiles de l'environnement. D'une maniéré plus simple ce module reconnaît si une primitive nouvelle détectée aux instants précédents, est fixe ou mobile. Il génère donc un ensemble de primitives fixes nouvelles qui seront fusionnées à la carte des objets et primitives stationnaires de l'environnement. Parmi ces primitives nouvellement perçues, celles considérées comme mobiles seront intégrées dans M_t, la liste des primitives mobiles de l'environnement.

- *Détection des Objets Mobiles/Appariement des Primitives Fixes :* parmi les primitives extraites dans les données courantes Z_t, ce module recherche le sous-ensemble de primitives qui correspondent ou appartiennent aux objets et primitives déjà perçus lors des acquisitions précédentes, sur des composantes fixes de l'environnement. En échangeant des informations avec le module de Suivi, il en déduit une liste de primitives nouvelles qui seront analysées dans les itérations suivantes pour les classer comme Mobile ou Stationnaire.

- *Segmentation-Fusion-Identification pour la localisation simultanée (SLAM) :* à partir des listes de primitives fixes appariées et fixes nouvelles, qui va permettre d'obtenir la carte de l'environnement ou évolue le robot ainsi que la position, à chaque instant, de celui-ci dans cette carte.

- *Segmentation-Fusion-Identification pour le suivi d'objets mobiles (MOT) :* même traitement que le module précédent, mais pour les composantes mobiles de la scène. Les primitives mobiles sont stockées dans des listes ; le regroupement en objets exploite aussi la cohérence des mouvements des primitives appartenant au même Objet.

Finalement, on peut dire que l'importance de la perception peut être vue à partir du fait qu'elle est le premier élément dans l'architecture d'un système robotique qui regroupe également la décision et l'action. On constate d'emblée que les actions du robot dépendent des décisions qu'il a prise, qui elles, dépendent directement du modèle de l'environnement produit par la perception.

Annexe B

La méthode de la fenêtre glissante

B.1 La méthode de la fenêtre glissante

La méthode de la détection par fenêtre glissante se base sur la méthodologie qui consiste à scanner une image sur toutes les positions pertinentes et sur toutes les échelles pour détecter une présence humaine, voir figure B.1. Comme généralement de nombreuses positions et échelles sont scannées cette technique est intrinsèquement coûteuse et gourmande en calcul. Heureusement, grâce aux progrès récents dans les GPU (les cartes graphiques), la détection des gens en temps réel est devenue possible et assez facile à implémenter, par exemple démontré par [13]. Pour pouvoir appliquer cette méthode on aura besoin de deux composant

Figure B.1: détection par la méthode de la fenêtre glissante, les fenêtres en jaune contiennent des personne par contre celle en rouge ne contiens pas d'être humain.

essentiels qui sont :

- *La fonction composant :* cette fonction a pour objectif de coder et d'enregistrer l'aspect visuel de la personne.
- *Le classificateur (le classifier) :* ce dernier a pour fonction principale la détermination pour chaque fenêtre glissante, celle qui contient ou bien ne contient pas la personne.

Cette méthode consiste en la construction d'une fenêtre de taille fixe (par exemple $n*m$) qui va parcourir graduellement l'image de test, au fur et à mesure que cette fenêtre se déplace il aura lieu une comparaison avec des donnés sur la personne ou bien les personnes recherchés

(la fonction composant), si la fenêtre contient une présence humain elle sera marquée positif sinon elle sera marquée négatif à la fin on aura un classificateur avec une base de donnés étiquetées soit positive ou bien négative.

B.2 Algorithme de détection par fenêtre glissante

- *Former un classificateur sur n * m fenêtres d'image. Les exemples positifs contiennent l'objet et les exemples négatifs ne le contiennent pas.*
- *Choisissez un seuil t et les étapes Δx et Δy dans les directions x et y*
- *Construire une pyramide d'images.*
- *Pour chaque niveau de la pyramide appliquer le classificateur à chaque $n * m$ fenêtres, pas à pas par δx et δy dans ce niveau pour obtenir une force de la réponse c.*
- *Si $c > t$*
- *Insérez un pointeur vers la fenêtre dans une liste classée L, selon leur indice de c.*
- *Pour chaque fenêtre W en L, en commençant par la plus forte réponse*
- *Retirez toutes les fenêtres $U \neq W$ qui se chevauchent.*
- *W, le chevauchement est calculée dans l'image d'origine en augmentant les fenêtres dans les échelles grossières.*
- *L est maintenant la liste des objets détectés.*

Annexe C

Le gradient d'une image

L'analyse des images a souvent pour base l'étude des variations locales du niveau de gris. La dérivation de la fonction image est l'outil le plus utilisé pour mettre en évidence ces variations locales, d'ou l'importance du gradient.

Définition 5. *On définit le vecteur gradient* $\overrightarrow{\nabla I}$ *de la fonction* $I(x, y)$ *par :*

$$\overrightarrow{\nabla I} \begin{pmatrix} \frac{\delta I(x,y)}{\delta x} \\ \frac{\delta I(x,y)}{\delta y} \end{pmatrix} \tag{C.1}$$

le vecteur gradient est caractérisé par son module et sa direction. les expressions usuelles des grandeurs en norme euclidienne sont:

$$| \overrightarrow{\nabla I} |= [(\frac{\delta I(x,y)}{\delta x})^2 + (\frac{\delta I(x,y)}{\delta y})^2]^{1/2} \tag{C.2}$$

$$\theta = Arg(\overrightarrow{\nabla I}) = \tan^{-1}(\frac{\delta I(x,y)}{\delta x} / \frac{\delta I(x,y)}{\delta y}) \tag{C.3}$$

Le vecteur gradient peut être calculé pour chaque pixel[1] d'une image. Il est simplement une mesure de la variation des valeurs de pixel le long de la direction x et la direction y autour de chaque pixel [9]. Regardons un exemple simple ; disons que nous voulons calculer le vecteur gradient au niveau du pixel en rouge, voir figureC.1.

Dans cette image en niveaux de gris, les valeurs de pixels vont juste de 0 à 255 (0 correspond au noir, 255 correspond au blanc). Les valeurs de pixels à gauche et à droite du pixel en question sont marqués dans l'image: 56 et 94. Nous prenons juste la valeur diminuée de la valeur de gauche et dire que le taux de changement dans la direction x est 38 ($94 - 56 = 38$).

[1]Pixel: (souvent abrégé px) est l'unité de base permettant de mesurer la définition (la taille) matricielle d'une image numérique. Son nom provient de la locution anglaise "picture element", qui signifie « élément d'image ».

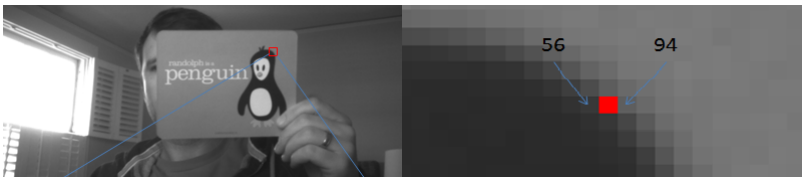

Figure C.1: Exemple de calcul du gradient

Le calcul du gradient peut se faire en soustrayant de gauche à droite ou de droite à gauche, il faut juste être cohérent sur l'image, voir figure C.2
Nous pouvons faire la même chose pour les pixels au-dessus et au-dessous pour obtenir le changement dans la direction y, $93 - 55 = 38$ dans la direction de y.

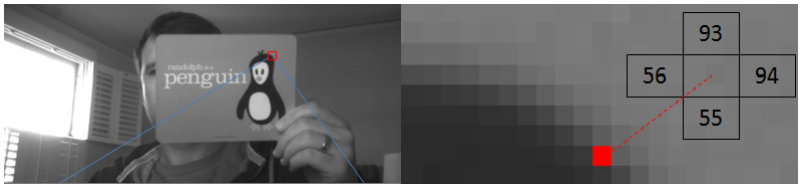

Figure C.2: Exemple de calcul du gradient

Mettons ces deux valeurs ensemble nous avons maintenant notre vecteur.

$$[38 \quad 38]^T$$

Nous pouvons également utiliser les équations du module et de l'argument d'un vecteur pour calculer les valeurs suivantes:

$$|\overrightarrow{\nabla I}| = \sqrt{38^2 + 38^2} = 53.74$$

$$\theta = \arctan \frac{38}{38} = 45$$

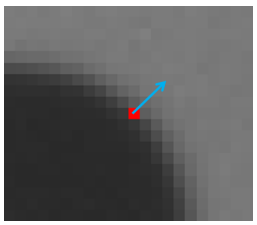

Figure C.3: représentation du vecteur gradient calculé

Nous pouvons maintenant tirer le vecteur de gradient comme une flèche sur l'image, voir figure C.3. On peut bien remarquer comment la direction du vecteur de gradient est perpendiculaire au bord de la tête ce qui est une propriété importante de ce vecteur.

C.1 Les applications de l'image gradient

La première et la plus évidente application du vecteur gradient est la détection de contour[1]. La figure C.4 illustre comment les grandes valeurs de gradient correspondent aux contours forts dans l'image. L'autre application moins évidente est l'extraction des caractéristiques des

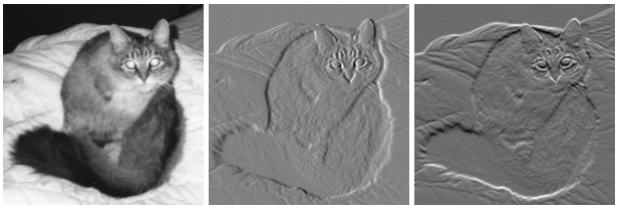

Figure C.4: A gauche, une image d'intensité d'un chat. Au centre, une image gradient dans la direction x. A droite, une image gradient dans la direction y.

objets. Regardons ce qui ce passe au vecteur de gradient lorsque on augmente la luminosité de l'image en ajoutant 50 à l'ensemble des valeurs de pixels.

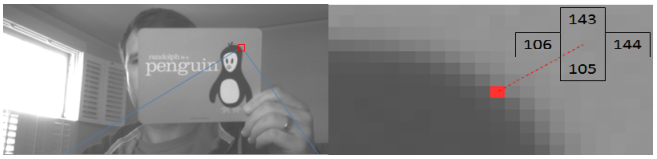

Figure C.5: Changement de luminosité de l'image.

Dans cette image plus lumineuse, voir figureC.5, le taux de changement dans la direction x est encore $144 - 106 = 38$, et le taux de changement dans la direction y est encore $143 - 105 = 38$, le même que dans notre image originale. Ainsi, même si les valeurs de pixels sont complètement différentes, nous obtenons toujours le même vecteur de gradient à ce pixel.

Lorsque nous basons nos descripteurs sur des vecteurs de gradient au lieu de prendre simplement les valeurs brutes de pixels, nous gagnons quelques "invariance par rapport au changement de luminosité",donc de la robustesse. Donc nous calculons le même descripteur (ou au moins proche du même descripteur) pour un objet dans différentes conditions d'éclairage, ce qui rend plus facile de reconnaître l'objet en dépit des changements dans l'éclairage.

[1]Contour ou pixels de contour, ce sont les Pixels situés dans des zones locales de forte discontinuité (une forte variation locale) d'intensité, de couleur ou de texture.

Annexe D

Histogramme d'intensité

L'histogramme d'intensité est la mesure du nombre de pixels dans une image pour chaque valeur d'intensité, voir figure D.1. Mathématiquement, l'histogramme pour une image au niveau de gris est une fonction discrète $h(r_k) = n_k$, où rk est le niveau de gris k_{eme} et n_k est le nombre de pixels de l'image qui ont ce niveau de gris r_k.
Les Histogrammes peuvent également être appliqués aux images couleur. Pour cela il 'y a deux représentation possible soit un histogramme individuel pour chaque couleur (bleu, rouge et vert), ou bien un histogramme 3D qui peut être produit avec trois axes représentant les canaux RGB, et la valeur à chaque point qui représente le nombre de pixels.
Le traitement de l'histogramme est très simple. Les intensités de l'image sont considérés comme des variables aléatoires avec une fonction de densité de probabilité *(PDF)* pour *(probability density function)* , cette fonction de densité peuvent être estimés à partir des données empiriques fournies par l'image elle-même. Par conséquent, chaque pixel de l'image est numérisé, et un comptage du nombre de pixels trouvés à chaque valeur d'intensité est maintenu. Autrement dit, la distribution de la fréquence des intensités de l'image est enregistrée. Elle est ensuite utilisée pour construire un histogramme d'intensités de l'image, comme le montre la figure suivante.

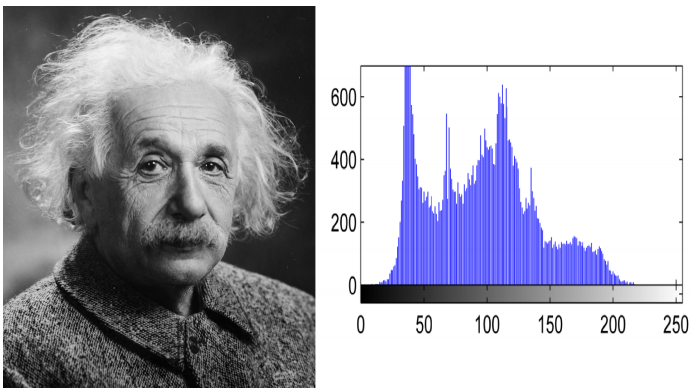

Figure D.1: A gauche : l'image simple. A droite : histogramme d'intensité correspondant.

L'histogramme doit être normalisé en divisant chaque bin (case) par le nombre total de pixels de l'image, afin de donner une estimation pour le PDF. Ensuite, chaque élément du tableau indique la probabilité que l'intensité se produisant à un pixel sélectionné de façon aléatoire. Donc, l'histogramme des intensités ne contient que des informations globales de l'image entière, qui peut être utilisé comme un descripteur de fonctionnalité très compacte pour la classification d'objets.
Cependant, le principal inconvénient de cette approche est que chaque image aura un seul histogramme, et toute l'information spatiale de l'image est éliminée. Une solution possible pourrait être d'intégrer l'histogramme des intensités avec d'autres types d'entités pour produire conjointement des représentations robustes des caractéristiques de l'image [52].

Annexe E

Machine à vecteurs de support SVM

Les machines à vecteurs de support ou séparateurs à vaste marge (en anglais Support Vector Machine, SVM) sont un ensemble de techniques d'apprentissage supervisé destinées à résoudre des problèmes de discrimination et de régression. Les SVM sont une généralisation des *classifieurs linéaires*. Ils ont été développés dans les années 1990 à partir des considérations théoriques de *Vladimir Vapnik* [45] sur le développement d'une théorie statistique de l'apprentissage.

E.1 Principe général

Les *SVM* peuvent être utilisés pour résoudre des problèmes de *discrimination*, c'est-à-dire décider à quelle classe appartient un échantillon, ou de *régression*, c'est-à-dire prédire la valeur numérique d'une variable. La résolution de ces deux problèmes passe par la construction d'une fonction h qui, à un vecteur d'entrée x, fait correspondre une sortie y :

$$y = h(x) \tag{E.1}$$

On se limite pour notre cas à un problème de discrimination à deux classes (discrimination binaire), c'est-à-dire $y\epsilon\{-1, 1\}$, le vecteur d'entrée x étant dans un espace X muni d'un produit scalaire. On peut prendre par exemple $X = R^N$.

E.2 Discrimination linéaire et hyperplan séparateur

Le cas simple est le cas d'une fonction discriminante linéaire, obtenue par combinaison linéaire du vecteur d'entrée $x = X_1, ..., X_N{}^T$, avec un vecteur de poids $w = W_1, ..., W_N{}^T$:

$$h(x) = w^T.x + w_0 \tag{E.2}$$

Il est alors décidé que x est de classe 1 si $h(x) \geq 0$ et de classe -1 sinon. C'est un classifieur linéaire. La frontière de décision $h(x) = 0$ est un hyperplan, appelé hyperplan séparateur, ou séparatrice. Le but d'un algorithme d'apprentissage supervisé est d'apprendre la fonction $h(x)$ par le biais d'un ensemble d'apprentissage :

$$\{(x_1, l_1), (x_2, l_2), ..., (x_k, l_k), ..., (x_p, l_p)\} \subset R^N * \{-1, 1\} \tag{E.3}$$

où les l_k sont les labels, p est la taille de l'ensemble d'apprentissage, N la dimension des vecteurs d'entrée. Si le problème est linéairement séparable, on doit alors avoir :

$$l_k.h(x_k) \geq 0 \ \ \& \ \ 1 \leq k \leq p \tag{E.4}$$

autrement dit:

$$l_k.(w^T.x_k + w_0) \geq 0 \ \ \& \ \ 1 \leq k \leq p \tag{E.5}$$

E.3 Marge maximale

On se place désormais dans le cas où le problème est linéairement séparable. Même dans ce cas simple, le choix de l'hyperplan séparateur n'est pas évident. Il existe en effet une infinité d'hyperplans séparateurs, dont les performances en apprentissage sont identiques, mais dont les performances en généralisation peuvent être très différentes. Pour résoudre ce problème, il a été montré [45], qu'il existe un unique hyperplan optimal, défini comme l'hyperplan qui maximise la marge entre les échantillons et l'hyperplan séparateur, ce qui est représenté à droite de la figure E.1.
Il existe des raisons théoriques à ce choix. *Vapnik* a montré que la capacité des classes d'hyperplans séparateurs diminue lorsque leur marge augmente.

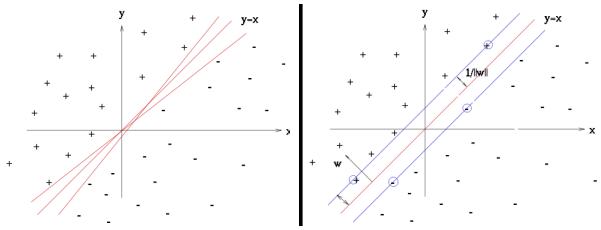

Figure E.1: A gauche : une infinité d'hyperplans séparateurs. A droite :L'hyperplan optimal (en rouge) avec la marge maximale.

La marge est la distance entre l'hyperplan et les échantillons les plus proches. Ces derniers sont appelés vecteurs supports. L'hyperplan qui maximise la marge est donné par :

$$min_k\{\| \ x - x_k \ \| : x \epsilon R^N, w^T.x + w_0 = 0\} \tag{E.6}$$

Il s'agit donc de trouver w et w_0 remplissant ces conditions, afin de déterminer l'équation de l'hyperplan séparateur :

$$h(x) = w^T.x + w_0 = 0 \tag{E.7}$$

E.4 Recherche de l'hyperplan optimal

La marge est la plus petite distance entre les échantillons d'apprentissage et l'hyperplan séparateur qui satisfasse la condition de séparabilité (à savoir $l_k(w^T.x_k + w_0) \geq 0$ comme expliqué précédemment). La distance d'un échantillon x_k à l'hyperplan est donnée par sa projection orthogonale sur l'hyperplan:

$$\frac{l_k(w^T x_k + w_0)}{||w||} \tag{E.8}$$

L'hyperplan séparateur $(w, w0)$ de marge maximale est donc donné par :

$$\left\{ \min_k \left[\frac{1}{||w||} l_k(w^T x_k + w_0) \right] \right\} \tag{E.9}$$

Afin de faciliter l'optimisation, on choisit de normaliser w et w_0, de telle manière que les échantillons à la marge (x^+_{marge} pour les vecteurs supports sur la frontière positive, et x^-_{marge} pour ceux situés sur la frontière opposée) satisfassent :

$$w^T.x^+_{marge} + w_0 = 1 \tag{E.10}$$

$$w^T.x^-_{marge} + w_0 = -1 \tag{E.11}$$

D'où pour tous les échantillons, $k = 1, \ldots, p$

$$l_k(w^T x_k + w_0) \geq 1 \tag{E.12}$$

Cette normalisation est parfois appelée la forme canonique de l'hyperplan, ou hyperplan canonique.

Avec cette mise à l'échelle, la marge vaut désormais $\frac{1}{||w||}$, il s'agit donc de maximiser $||w||^{-1}$. La formulation dite primale des SVM s'exprime alors sous la forme suivante:

$$Minimiser \quad \frac{1}{2}||w||^2 \quad sous\ les\ contraintes \quad l_k(w^T x_k + w_0) \geq 1$$

Ceci peut se résoudre par la méthode classique des multiplicateurs de Lagrange, où le

lagrangien est donné par :

$$L(w, w_0, \alpha) = \frac{1}{2}||w||^2 - \sum_{k=1}^{p} \alpha_k \left\{ l_k(w^T x_k + w_0) - 1 \right\} \tag{E.13}$$

Le lagrangien doit être minimisé par rapport à w et w_0, et maximisé par rapport à α Afin d'obtenir l'hyperplan solution, on remplace w par sa valeur optimale w^*, dans l'équation de l'hyperplan $h(x)$, ce qui donne :

$$h(x) = \sum_{k=1}^{p} \alpha_k^* l_k(x \cdot x_k) + w_0 \tag{E.14}$$

.

Annexe F

image intégrale

Pour calculer rapidement et efficacement ces caractéristiques sur une image, les auteurs proposent également une nouvelle méthode, qu'ils appellent « image intégrale ». C'est une représentation sous la forme d'une image, de même taille que l'image d'origine, qui en chacun de ses points contient la somme des luminances des pixels situés au-dessus de lui et à sa gauche. Plus formellement, l'image intégrale ii au point (x, y) est définie à partir de l'image i, voir figure F.1 Le calcul de la somme des valeurs des pixels appartenant à une zone rectangulaire s'effectue donc en accédant seulement à quatre pixel de l'image intégrale : soit un rectangle $ABCD$ dont les sommets sont nommés dans le sens des aiguilles d'une montre en commençant par le sommet supérieur gauche et soit x la valeur sous la représentation intégrale d'un sommet X du rectangle ($X\epsilon\{A, B, C, D\}$). La somme des valeurs des pixels appartement à $ABCD$ est, quelle que soit sa taille, donnée par $c-b-d+a$. Une caractéristique de *Haar* étant une combinaison linéaire de tels rectangles $ABCD$, son calcul se fait alors en un temps indépendant de sa taille. Illustrons le calcul de la somme de

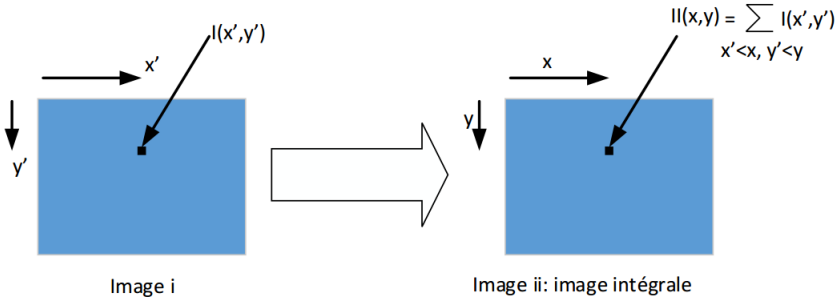

Figure F.1: Calcul de l'image intégral

la luminance des pixels dans la zone hachurée ABCD dans la figure F.2. Seulement 4 accès à l'image intégrale nous permet de calculer la somme de la luminance de la zone $ABCD$ ($D + A - B - C$) Une caractéristique *pseudo-Haar* formée de deux zones rectangulaires peut être calculée en seulement 6 accès à l'image intégrale, et donc en un temps constant quelle que soit la taille de la caractéristique.

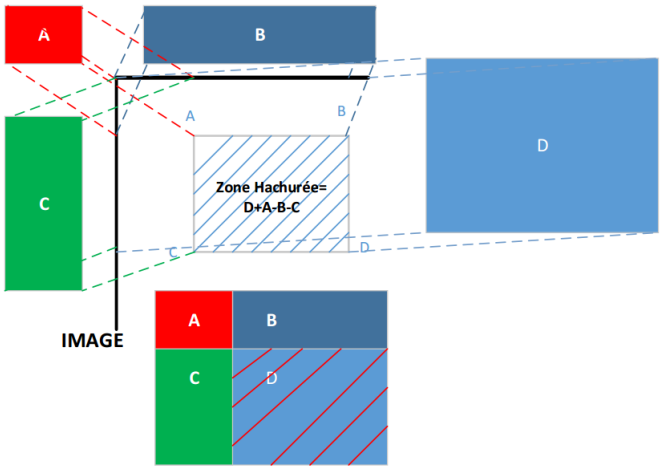

Figure F.2: Illustration du calcul de la somme de la luminance des pixels

Annexe G

Algorithme d'apprentissage basé sur Adaboost

L'algorithme de boosting utilisé est en pratique une version modifiée d'AdaBoost, qui est utilisée à la fois pour la sélection et pour l'apprentissage d'un classifieur fort. Les classifieurs faibles utilisés sont souvent des arbres de décision. Un cas remarquable, fréquemment rencontré, est celui de l'arbre de profondeur, qui réduit l'opération de classification à un simple seuillage.
L'algorithme est de type itératif, à nombre d'itérations déterminé. À chaque itération, l'algorithme sélectionne une caractéristique, qui sera ajoutée à la liste des caractéristiques sélectionnées aux itérations précédentes, et le tout va contribuer à la construction du classifieur fort final. Cette sélection se fait en entraînant un classifieur faible pour toutes les caractéristiques et en sélectionnant celui avec l'erreur la plus faible sur l'ensemble d'apprentissage.
L'algorithme tient également à jour une distribution de probabilité sur l'ensemble d'apprentissage, réévaluée à chaque itération en fonction des résultats de classification. En particulier, plus de poids est attribué aux exemples difficiles à classer, c'est-à-dire ceux dont l'erreur est élevée.
Le classifieur fort final construit par AdaBoost est composé de la somme pondérée des classifieurs sélectionnés.
Plus formellement, on considère un ensemble de n images $(x_1, ..., x_n)$ et leurs étiquettes associées $(y_1, ..., y_n)$, qui sont telles que $y_i = 0$ si l'image x_i est un exemple négatif et $y_i = 1$ si x_i est un exemple de l'objet à détecter (une personne). L'algorithme de boosting est constitué d'un nombre T d'itérations, et pour chaque itération t et chaque caractéristique j, on construit un classifieur faible h_j. Idéalement, le but est d'obtenir un classifieur h qui prédise exactement les étiquettes pour chaque échantillon, c'est-à-dire $y_i = h(x_i)\ \forall\ i\epsilon\{1, ..., n\}$. En pratique, le classifieur n'est pas parfait et l'erreur engendrée par ce classifieur est donnée par :

$$\epsilon_j = \sum_{i=1}^{n} \omega_i |h_j(x_i) - y_i| \tag{G.1}$$

Les ω_i étant les poids associés à chaque exemple et mis à jour à chaque itération en fonction de l'erreur obtenue à l'itération précédente. On sélectionne alors à l'itération t le classifieur h_t présentant l'erreur la plus faible : $\epsilon_t = min(\epsilon_j)$. Le classifieur fort final $h(x)$ est construit par seuillage de la somme pondérée des classifieurs faibles sélectionnés :

$$h(x) = \begin{cases} 1 \; si \; \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 \; sinon \end{cases} \tag{G.2}$$

Les α_t sont des coefficients calculés à partir de l'erreur ϵ_t:

$$\alpha_t = \frac{1}{2} \ln \frac{1 - \epsilon_t}{\epsilon_t} \tag{G.3}$$

Une des limitations de la méthode de *Viola et Jones* est son manque de robustesse à la rotation, et sa difficulté à apprendre plusieurs vues d'un même objet. En particulier, il est difficile d'obtenir un classifieur capable de détecter à la fois des visages de face et de profil. *Viola et Jones* ont proposé une amélioration qui permet de corriger ce défaut [24], qui consiste à apprendre une cascade dédiée à chaque orientation ou vue, et à utiliser lors de la détection un arbre de décision pour sélectionner la bonne cascade à appliquer. Plusieurs autres améliorations ont été proposées par la suite pour apporter une solution à ce problème [5, 20].

Annexe H

Analyse Discriminante Linéaire (Fisher-Faces)

L'approche consiste à rassembler les images de la base d'apprentissage dans une grande matrice d'image Γ ou chaque colonne représente une image Γ_i . Puis on calcule l'image moyenne Ψ

$$\Downarrow$$

$$\mathbf{I_1} = \begin{pmatrix} a_{1,1} & \dots & a_{1,m} \\ \vdots & \ddots & \vdots \\ a_{n,1} & \dots & a_{n,m} \end{pmatrix} \quad \Rightarrow \quad \mathbf{\Gamma_1} = \begin{pmatrix} a_{1,1} \\ \vdots \\ a_{n,1} \\ \vdots \\ a_{1,m} \\ \vdots \\ a_{n,m} \end{pmatrix}$$

Figure H.1: Passage d'une image vers un vecteur dans un espace vectoriel de grande dimension. Les coefficients a_{ij} représentent les valeurs des pixels en niveau de gris, codés de 0 à 255.

Ensuite, pour chaque classe C_i , on calcule l'image moyenne ψ_{Ci} :

$$\psi_{Ci} = \frac{1}{q_i} \sum_{k=1}^{q_i} \Gamma_k \tag{H.1}$$

Avec q_i , le nombre d'images dans la classe C_i. Chaque image Γ_i de chaque classe C_i est

ensuite recentrée par rapport à la moyenne. On obtient alors une nouvelle image ϕ_i :

$$\phi_i = \Gamma_i - \psi_{Ci} \tag{H.2}$$

Vient ensuite le calcul de nos différentes matrices de dispersion. On notera c le nombre total de classes (i.e. le nombre d'individus), q_i le nombre d'images dans la classe C_i et M le nombre total d'images.

- La Matrice de Dispersion Intra-Classe (S_ω)

$$S_\omega = \sum_{i=1}^{c} \sum_{\Gamma_k \epsilon C_i} (\Gamma_k - \psi_{Ci})(\Gamma_k - \psi_{Ci})^T$$

- La Matrice de Dispersion Inter-Classe (S_b)

$$S_b = \sum_{i=1}^{c} q_i(\psi_{Ci} - \psi)(\psi_{Ci} - \psi)^T$$

- La Matrice de Dispersion Totale (S_T)

$$S_T = \sum_{i=1}^{M} (\Gamma_i - \psi)(\Gamma_i - \psi)^T$$

Une fois ces matrices calculées, nous devons trouver une projection optimale W qui maximise la dispersion intra-classe, relative à la matrice S_ω , tout en minimisant la dispersion interclasse, relative à la matrice S_b. En d'autres termes, nous devons trouver W qui maximise deux critère $J(T)$ le critère d'optimisation de Fisher et celui de l'approche l'Analyse en Composantes Principales (PCA) :

$$W = arg\ max_T(J(T)) \tag{H.3}$$

$$W_{PCA} = arg\ max_W |W^T S_T W| \tag{H.4}$$

$$W_F LD = arg\ max_W \mid \frac{W^T W_{PCA}^T S_B W_{PCA}^T W}{W^T W_{PCA}^T S_W W_{PCA}^T W} \mid \tag{H.5}$$

$$W_{opt}^T = W_{FLD}^T W_{PCA}^T \tag{H.6}$$

Une fois W trouvée, on effectue une projection des images apprises ainsi que la projection de l'image test. Ainsi, la projection vectorielle d'une image apprise réajustée par rapport à

l a moyenne ϕ_i est définie par :

$$g(\phi_i) = W^T \phi_i \tag{H.7}$$

La phase de reconnaissance d'une image test ϕ_t s'effectue en projetant ϕ_t s'effectue en projetant W^T :

$$g(\phi_i) = W^T \phi_t \tag{H.8}$$

Enfin, on effectue une mesure de distance entre l'image test et l'image projetée sur l'espace vectoriel engendré, pour la distance Euclidienne, on calcule la distance d_{ti} :

$$d_{ti} = \| \, g(\phi_t) - g(\phi_i) \, \| \tag{H.9}$$

d'où

$$d_{ti} = \sqrt{\sum_{k=1}^{c} (g(\phi_t) - g(\phi_i))^2} \tag{H.10}$$

Finalement, une image test est dans la classe dont la distance est minimale par rapport à toutes les autres distances de classe.

Annexe I

Rappels et définitions

Les différentes définitions et rappels donnés ci-dessous sont extraits de la référence [17].

I.1 Caractéristique d'une variable aléatoire scalaire

Soit X une variable aléatoire scalaire prenant ses valeurs dans $\Re$. La fonction de répartition $F(x)$ associe à tout réel x la probabilité de l'évenement $X < x$. On note:

$$F(x) = P[X < x] \tag{I.1}$$

Propriétés:

- *quelq soit* $x_1 < x_2$, $P[x \leq X \leq x_2] = F(x_2) - F(x_1)$
- $\lim_{x \to +\infty} F(x) = 1$; $\lim_{x \to -\infty} F(x) = 0$.
- $F(x)$ est monotone, non décroissante, et peut étre continue ou discontinue selon que X prenne des valeurs continues ou discrètes.

Si $F(x)$ est dérivable, alors sa dérivée est appelée densité de probabilité et notée $p(x)$:

$$p(x) = \frac{dF(x)}{dx} \; soit : \; p(x)dx = P[x \leq X \leq x + dx] \tag{I.2}$$

Pour caractériser et manipuler mathématiquement une variable aléatoire X, on utilise également *les moments* de cette variable. Le moment d'ordre 1 est plus connu sous le nom de moyenne ou espérance mathématique. Le moment centré d'ordre 2 est appelée variance que l'on note $var_x = \sigma_x^2$; σ_x désigne l'écart type.
Soit:

- l'espérance mathématique ou moyenne:

$$E[X] = \int_{-\infty}^{+\infty} xp(x)dx = \int_{-\infty}^{+\infty} xdF(x), \tag{I.3}$$

- le moment d'ordre k:

$$E[X^k] = \int_{-\infty}^{+\infty} x^k p(x) dx, \tag{I.4}$$

- le moment centré d'ordre k:

$$E[(X - E[X]^k)] = \int_{-\infty}^{+\infty} (x - E[X])^k p(x) dx. \tag{I.5}$$

L'intérêt (mathématique) des variables aléatoires gaussiennes est qu'elles sont entièrement caractérisées par leurs moments d'ordre 1 et 2. Soit X une variable aléatoire gaussienne de moyenne m et d'écart type σ, alors:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{(x-m)^2}{2\sigma^2}}, E[x] = m, E[(x-m)^2] = \sigma^2 \tag{I.6}$$

Variable aléatoire discrète: si la variable aléatoire X prends ses valeurs dans un ensemble discret de N valeurs x_i,$i = 1, ..., N$ alors on ne parle plus de densité de probabilité et on défini directement la "probabilité" que $X = x_i$ que l'on note $P(X = x_i)$. Le calcul des moments fais alors intervenir une somme discrète:

$$E[X^k] = \sum_{i=1}^{N} x_i^k P(X = x_i) \tag{I.7}$$

I.2 Caractéristique d'une variable aléatoire à plusieurs dimensions

Soit $X = [X_1, ..., X_q]^T$ une variable aléatoire à q dimension prenant ses valeurs dans $\Re^q$.
Fonction de répartition

$$F(x_1, ..., x_q) = P[X_1 \leq x_1 \; et \; X_2 \leq x_2 \; et ... et \; X_q \leq x_q] \tag{I.8}$$

Densité de probabilité

$$p(x_1, ..., x_q) = \frac{\partial^q F(x_1, ..., x_q)}{\partial x_1 ... \partial x_q} \tag{I.9}$$

Les Moments

On note: $x = [x_1, ..., x_q]^T$ et on ne s'intéressera qu'au vecteur des moments d'ordre 1 (le vecteur moyen) et à la matrice des moments d'ordre 2 centrés (la matrice de covariance)

- Moyenne: $E[X] = [E[x_1], ..., E[X_q]]^T$.

- Covariance: $Cov_x = E[(X - E[X])(X - E[X])^T]$. La matrice de covariance est définie positive et symétrique.

Indépendance Deux variables aléatoires X_1 et X_2 sont indépendantes si et si seulement si:

$$F(x_1, x_2) = F(x_1)F(x_2) \tag{I.10}$$

Une condition nécessaire d'indépendance s'écrit:

$$E[X_1X_2] = E[X_1]E[X_2] \tag{I.11}$$

I.3 Signal aléatoire (processus stochastique)

Etant donné une variable aléatoire X, le signal aléatoire ou processus stochastique $x(t)$ est un signal fonction du temps t tel que pour tout t fixé, $x(t)$ corresponde à une valeur de la variable aléatoire X.

I.4 Moments d'un signal aléatoire

Le moment d'ordre 2 d'un signal aléatoire est appelé la fonction d'auto-corrélation.
Soit $w(t)$ un signal aléatoire, alors:

$$\begin{cases} moment\ d'ordre\ 1:\ m(t) = E[w(t)] \\ moment\ d'ordre\ 2:\ \phi_{ww}(t,\tau) = E[w(t)w(t+\tau)^T]. \end{cases} \tag{I.12}$$

Remarque 1. si w(t) est un signal vectoriel à q composantes alors $\phi_{ww}(t,\tau)$ est une matrice de taille $q*q$ définie positive pour chaque valeur de t et de τ. Les termes de la diagonales sont les fonctions scalaires d'auto-corrélation de chaque composantes et les termes hors-diagonaux sont les fonctions scalaires d'inter-corrélation entre composantes.

Un signal aléatoire gaussien centré, c'est-à-dire à moyenne nulle, est donc entièrement défini par sa fonction d'auto-corrélation.

I.5 Bruit blan

Un bruit blanc est un signal aléatoire de variance infinie dont la fonction d'auto-corrélation est proportionnelle à un Dirac (c'est-à-dire un spectre complexe constant sur toute la plage des fréquences). Cela traduit que les valeurs du signal pris à deux instants, même très proches, ne sont pas du tout corrélées. Les bruits blancs gaussiens centrés $w(t)$ et $v(t)$ que nous allons utiliser dans le cadre du filtre de Kalman sont donc entièrement définis par leur

densités spectrales respectives $W(t)$ et $V(t)$.
Les matrices $W(t)$ et $V(t)$ deviennent constantes dans le cas de bruits blancs stationnaires. Le bruit blanc gaussien normalisé est tel que $W(t) = I_{q*q}$ (q: nombre de composantes dans le bruit).

I.6 Filtre de Kalman

La fonction de filtrage consiste à estimer une information (signal) utile qui est polluée par un bruit. Alors que le filtre de Kalman vise à éstimer de façon optimale l'état du système linéaire (cet état correspond donc à l'information utile) [17].

I.6.1 Position du problème

Considérons le modèle d'état d'un système supposé linéaire et stationnaire :

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) + w(t) & (\textit{équation de l'état}) \\ y(t) = Cx(t) + Du(t) + v(t) & (\textit{équation de la mesure}) \end{cases} \tag{I.13}$$

où :

- $x \epsilon \Re^n$ est le vecteur d'état du système,
- $u(t) \epsilon \Re^m$ est le vecteur des entrées déterministes et connues (commandes,...),
- $w(t) \epsilon \Re^q$ est le bruit des signaux aléatoires inconnus qui viennent perturber directement l'équation d'état du système.
- $y(t) \epsilon \Re^p$ est le vecteur des mesures,
- $v(t) \epsilon \Re^p$ est le bruit de mesure qui polluent les mesures $y(t)$.

Des rappels et des définitions sur les différentes notions de base nécessaire pour la compréhension du filtre de *Kalman* sont donnés dans l'annexe I.

I.6.2 Filtre de Kalman discret

le filtre de Kalman vise à estimer de façon optimale l'état $x \epsilon \Re$ d'un processus à temps discret géré par une équation linéaire stochastique [17].

$$\begin{cases} x_k = Ax_{k-1} + bv_k + w_{k-1} \\ y_k = Cx_k + v_k \end{cases} \tag{I.14}$$

w_k et v_k représentent le bruit des signaux aléatoire inconnus qui perturbent l'état du système et les mesures (respectivement).
Dans ce mémoire on considère que v_k et w_k comme des bruits blanc indépendants avec une distrubition de probabilité normales alors:

$$\begin{cases} P(w) \sim N(0, Q) \\ P(v) \sim N(0, R) \end{cases} \tag{I.15}$$

- Q: représente la matrice de covariance du bruit de l'état,
- R: représente la matrice de covariance du bruit de la mesure

En pratique les deux matrices Q et R peuvent être changées avec le pas de temps ou bien à chaque mesure. En revanche dans ce mémoire on va les considérer comme constantes. le même cas pour les autres matrice A et C.
On définit $\hat{x}_k^- \epsilon \Re^n$ comme etant l'état estimé a priori au pas k, et $\hat{x}_k$ comme étant l'état estimé a posteriori compte tenu de la mesure y_k. Alors on peut définir l'erreur de l'estimation a priori et posteriori comme [11]:

$$\begin{cases} e_k^- = x_k - \hat{x}_k^- \\ e_k = x_k - \hat{x}_k \end{cases} \tag{I.16}$$

La covariance de l'erreur estimée a priori est donc :

$$P_k^- = E[e_k^- e_k^{-T}] \tag{I.17}$$

et la covariance de l'erreur estimée a posteriori comme :

$$P_k^- = E[e_k e_k^T] \tag{I.18}$$

A partir des équations du filtre de Kalaman, et ayant comme objectif, trouver une équation qui calcul une estimation a posteriori de l'état comme une combinaison linéaire d'une estimation a priori et une différence pondérée entre la mesure réel et la mesure predite, on peut trouver l'équation suivante (pour plus d'information vous pouvez voir [33]):

$$\hat{x}_k = \hat{x}_k^- + K(y_k - C\hat{x}_k^-) \tag{I.19}$$

La différence dans l'équation (I.19) est appelée l'innovation de la mesure ou bien le résidu, il reflète la différence entre la mesure prédite et la mesure réel. Un résidu de zéro signifie que les deux sont totalement en accord.
La matrice K_{n*m} dans l'équation (I.19) est choisie comme le gain qui minimise la covariance de l'erreur a posteriori (l'équation (I.18)) [33].

alors l'expression de k s'écrit:

$$K_k = P_k^- C^T (C P_k^- C^T + R)^{-1} \tag{I.20}$$

$$K_k = \frac{P_k^- . C^T}{C P_k^- C^T + R} \tag{I.21}$$

En analysant l'équation (I.21) on peut voire que, si la covariance de l'erreur de mesure approche de zero, le gain K donne plus de poids au résidu.

$$\lim_{R_k \to 0} K_k = C^{-1} \tag{I.22}$$

D'autre part, si la covariance de l'erreur a priori, P_k^- approche de zéro, le gain K donne moins d'importance au résidu.

$$lim_{P_k^- \to 0} K_k = 0 \tag{I.23}$$

Une autre manière de voir le rôle du gain K, est que si la covariance R approche de 0, la mesure actuelle est de plus en plus "trusted", quand à elle, la mesure prédite est de moins en moins "trusted". D'autre part si la covariance P_k^- approche de 0 la mesure actuelle z_k est de moins en moins "trusted", alors que la mesure prédite est de plus en plus "trusted".

Annexe J

ROS

J.1 Un système d'exploitation OS pour robot

Définition 6. *En informatique, un système d'exploitation (souvent appelé OS pour operating system) est un ensemble de programmes qui dirige l'utilisation des capacités d'un ordinateur. Il reçoit et gère les différentes demandes d'utilisation des capacités matérielles (stockage en mémoire, calcul du processeur, communication vers des périphériques,..).*

De maniéré analogue, un système d'exploitation pour robot est définit comme étant aussi un ensemble de programmes (softwares) qui gèrent l'allocation des ressources matérielles du robot composées essentiellement par un ensemble de capteurs et d'actionneurs.

J.2 l'interet d'un OS pour les robots

Avant les OS robotiques, chaque concepteur de robot, chaque chercheur en robotique passait un temps non négligeable à concevoir matériellement son robot ainsi que le logiciel embarqué associé. Cela demandait des compétences en mécanique, électronique et programmation embarquée. Généralement, les programmes ainsi conçus correspondaient plus à de la programmation embarquée, proche de l'électronique, qu'à de la robotique proprement dite, telle que nous pouvons la rencontrer aujourd'hui dans la robotique de service. L'incapacité de comparer les résultats, peu de points communs entre les programmes développés, et enfin la réutilisation des programmes était non triviale car fortement liés au matériel sous-jacent. L'idée principale d'un *OS* robotique est d'éviter de réinventer la roue à chaque fois et de proposer des fonctionnalités standardisées faisant abstraction du matériel, tout comme un *OS* classique pour *PC*, d'où l'analogie de nom.

Figure J.1: Montre comment les chercheurs en robotique réinventent la roue.

Un autre avantage de l'OS robotique est de rassembler des savoir-faire de différentes disciplines. En effet, concevoir et programmer un robot, c'est :

- gérer le matériel en écrivant les pilotes;
- gérer la mémoire et les processus.
- gérer la concurrence et la fusion de données;
- proposer des algorithmes de raisonnement abstrait faisant largement appel à l'intelligence artificielle.

La robotique nécessite donc des compétences très différentes, compétences généralement hors de portée d'une seule personne.

J.3 Les distributions de ROS

Distribution	Date de sortie
ROS Kinetic Kame	Mai, 2016
ROS Jade Turtle	Mai , 2015
ROS Indigo Igloo	Juillet , 2014
ROS Hydro Medusa	Septembre , 2013
ROS Groovy Galapagos	Décembre , 2012
ROS Fuerte Turtle	Avril , 2012
ROS Electric Emys	Octobre , 2011
ROS C Turtle	Octobre , 2010

En ce qui concerne notre projet on va travailler avec *ROS indigo igloo* recommandé pour sa stabilité.

Figure J.2: ROS indigo igloo

J.4 ROS support

ROS est supporté par Les systèmes d'exploitations et les langages de programmation suivants:

- Ubuntu;
- Mac OS X systems;
- Fedora;
- Gentoo;
- Arch Linux;
- Windows (Partiellement).

Les langages de programmation :

- C++;
- Python;
- LISP;
- Java (alpha release);
- Lua;

J.5 Les caractéristiques de ROS

Le secret de la réussite de *ROS* réside dans ses caractéristiques qui le distinguent des autres systèmes d'exploitation pour robot et qui se résument dans les points suivant ; [25]

- Peer to Peer ;
- Basé sur des outils ;
- Multi-langages ;

- Léger ;

- Gratuit et open-source.

Reprenons chacun de ces points.

- *Peer-to-Peer:* Une tache robotique suffisamment complexe est composée de plusieurs programmes simples qui s'exécutent en général de manière concurrente. Ces programmes, appelés nœuds, communiquent entre eux de manière synchrone ou asynchrone en utilisant l'architecture Peer-to-Peer.

- textit Basé sur des outils: *ROS* est qualifié de neutre d'un point de vue langage. Toutefois, il est très fréquent que les programmes sous *ROS* soient écrits en *C++* ou en *Python*.

- *Multi-langages:* Afin de gérer au mieux la complexité de *ROS*, les concepteurs ont opté pour une architecture dite à *micro-noyau* (ou microkernel) où un grand nombre d'outils permettent essentiellement d'effectuer des tâches de navigation dans l'arborescence du code, d'obtenir les paramètres de configuration, de visualiser la topologie de la connexion Peer-to-Peer, de mesurer la bande passante, de tracer graphiquement des données de message, ...etc.

- textitLéger: Comme un *OS* pour robot a principalement pour but la réutilisation des programmes, les développeurs de *ROS* ont fait en sorte que les pilotes et autres algorithmes soient contenus dans des exécutables indépendants. Cela assure une réutilisation maximale mais surtout, le maintien d'une taille réduite étant donné que les parties communes entre les programmes ne sont stockées qu'une seule fois.

- textitGratuit: et open-source comme nous l'avons déjà mentionné, *ROS* est entièrement gratuit et open-source. Le code est accessible au publique sur le site officiel du système d'exploitation.

Le tableau suivant résume les points forts de *ROS* par rapport aux autre systèmes d'exploitations pour Robot :

OS conventionnel	ROS
OS avec un objectif général	Exclusif pour les robots
Ortho-Operational	Meta-operational
Programmation avec la Langue maternelle	Indépendant du langage de programmation
architecture séquentielle	Architecture asynchrone
Propriétaire / open-Source	Open-source sous une licence BSD
Des programmes lourds	Nœud
programmes de communication	Message
le noyau est inclus	Sans noyau

J.6 Feuille de route pour le développement de ROS

J.6.1 Niveau communautaire

Ce concept est représenté par les ressources de ROS qui permettent aux communautés distinctes d'échanger les logiciels et les connaissances.
Ces ressources comprennent :

- *Les distributions:* Ce sont les collections de déférentes versions de *Stack* qu'on peut installer. En effet la distribution joue le même rôle que la distribution *Linux*.
- *Les Référentiels: ROS* repose sur un réseau fédéré de référentiels de code, où les différentes institutions peuvent développer leurs propres composants logiciels du robot.
- *ROS Wiki:* La communauté *ROS* Wiki est le principale forum de documentation et d'informations à propos du *ROS*, alors n'importe qui peut créer un compte et contribuer avec ses propre documentations, et fournir des corrections ou bien des mise-à-jour, écrire des tutoriaux..etc.
- *Blog:* Le Blog de *Willow Garage* fournit des mise-à-jours régulières, incluant des photos et des vidéos.

Annexe K

Description de la Kinect

Kinect, contraction de "*kinetic*" et "*connect*" désigne un dispositif électronique basé sur une technologie logicielle développée par *Rare*, une filiale de *Microsoft Game Studios* appartenant à *Microsoft*, et sur une caméra spécifique créée par *PrimeSense*, qui interprète les informations sur la scène 3D obtenue à travers une lumière infrarouge structurée et projetée en continu.
Ce périphérique de *Microsoft* est constitué d'une barre horizontale reliée à une petite base avec un pivot motorisé, celui-ci permet à la caméra d'effectuer des petits mouvements vers le haut ou le bas, afin d'adapter la perception de la caméra en fonction de votre position dans la pièce ou bien pour être placé au-dessus ou en dessous de l'affichage vidéo (téléviseur, écran d'un ordinateur). Le dispositif comporte une caméra *RGB*, un capteur de profondeur « *3D depth sensor* » et une série de *microphone multi-réseau* exécutant un logiciel propriétaire, qui fournissent la capture du mouvement du corps en 3D, la reconnaissance faciale et la reconnaissance vocale.

Figure K.1: La Kinect de Microsoft

K.1 Les composants de la Kinect

La kinect est composé de trois parties essentielles;

- La camera *RGB* (Red Green Blue) ;
- Le capteur de profondeur « *3D depth sensor* » ;
- Le réseau *des microphones* ;

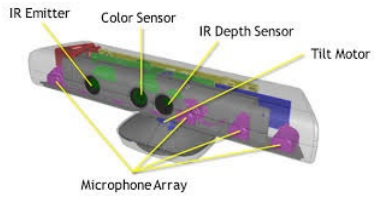

Figure K.2: Un petit récapitulatif des technologies embarquées dans la Kinect.

Reprenons ces 3 composants en détail :

K.1.1 La camera RGB (Red Green Blue)

La première des deux caméras embarquées dans la technologie Kinect est une caméra couleur *RGB* « standard » avec un capteur photographique de type *CMOS*. Elle se situe au centre de la barre horizontale.
Afin de bien comprendre le principe de la Kinect on va d'abord expliquer le fonctionnement du capteur photographique, qui est un composant électronique sensible à la lumière, qui convertie le rayonnement (Ultraviolet, Lumière visible ou Infra Rouge) en un signal analogique .Ce signal est ensuite numérisé par un convertisseur analogique-numérique afin d'obtenir une image numérique. La lumière est composée de 3 couleurs primaires qui sont le Rouge, le Vert et le Bleu, comme L'association de ces 3 couleurs permet d'obtenir la lumière blanche (le noir représente l'absence totale de couleur) chaque couleur est donc définie selon le système *RGB* c'est-à-dire en fonction de la proportion respective en rouge, vert et bleu. Le mélange de ces trois couleurs compose aussi chacun des pixels de nos écrans LCD, LED...etc.

Pour convertir le signal analogique reçu en image numérique, le capteur photographique va filtrer la lumière selon ces 3 couleurs et sortir trois signaux numériques correspondant à chacune d'elles. Actuellement, deux grandes familles de capteurs sont disponibles : les *CCD* et les *CMOS*.
Les *CCD* sont surtout utilisés dans les appareils compacts mais sont de plus en plus délaissés dans les reflex. Le *CMOS* est représenté sous la forme d'un petit « écran » de taille variable. Certains capteurs *CMOS* peuvent actuellement atteindre une résolution hallucinante de 25 millions de pixels et la technologie ne cesse de progresser.

Figure K.3: les trois couleurs primaires de la lumière sont le rouge, le vert et le bleu.

Figure K.4: Exemplaire d'un capteur photographique de type CMOS.

La lumière (l'image) arrive en face du capteur, celle-ci est d'abord purifiée par un filtre Infra Rouge (bloquant les ondes infrarouges et laissant passer la couleur), puis traverse un *mini filtre* de couleur rouge, vert ou bleu placé en face du capteur lui-même. Ainsi les millions de capteurs présents à la surface du *CMOS* vont émettre un signal électrique relatif à une des 3 couleurs, qui sera ensuite converti numériquement.

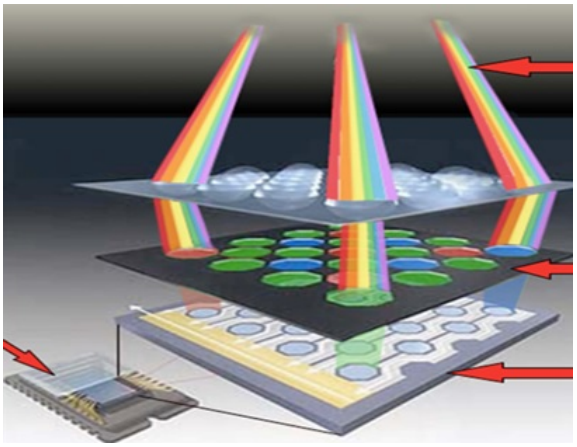

Figure K.5: Fonctionnement d'un capteur photographique de type CMOS pour les appareils photos numériques et les cameras couleur.

K.1.2 3D depth sensor

Le capteur de profondeur se compose d'un projecteur laser infrarouge combiné à un capteur *CMOS* monochrome, qui capture des données vidéo en 3D dans toutes les conditions de lumière ambiante. Le rayon de détection de ce capteur est réglable, et le logiciel de la *Kinect* est capable de calibrer automatiquement le capteur en fonction du *gameplay* ou de

l'environnement physique du joueur, pouvant accueillir la présence des meubles ou d'autres obstacles.

Le fonctionnement un capteur *CMOS* infrarouge est identique à une caméra *RGB* sauf qu'on laisse passer uniquement les infrarouges. La caméra infrarouge permet d'obtenir une image représentant les dégagements thermiques émis par l'objet observé. Cependant, les images obtenues ne sont pas colorées, ce sont les utilisateurs qui décident de rajouter les niveaux de couleurs en fonction de la température mesurée.

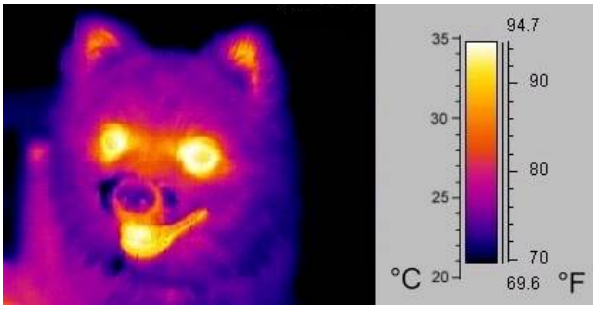

Figure K.6: Exemplaire d'un capteur photographique de type CMOS.

La *Kinect* ne s'appuie pas sur la chaleur émise par notre corps. En effet, tout l'intérêt de la *Kinect* provient de son émetteur (lampe) de lumière infrarouge. La scène est bombardée par les rayons infrarouges non visibles à l'œil humain. Une partie de ces rayonnements va être réfléchie par l'ensemble des surfaces touchées. Plus l'objet sera loin et plus la quantité de rayonnement infrarouge réfléchie (renvoyée vers la caméra) sera faible. A l'inverse, plus l'objet sera proche et plus la quantité de rayonnement infrarouge réfléchie sera importante. Ainsi la caméra infrarouge va mesurer la distance de l'objet en fonction de l'intensité.

Alors, il est possible de cartographier avec précision la distance pour tout objet éloigné

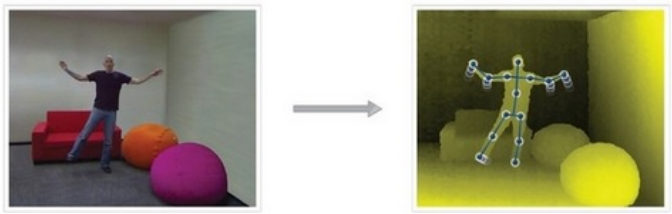

Figure K.7: Une scène réelle (à gauche) et l'image recueillie par la camera Kinect (à droite)

de 1,5 à 2 mètres de la caméra jusqu'à environ 4-5 mètres de profondeur. Au-delà de 5 mètres, le rayonnement *IR* réfléchi devient trop faible pour être mesurable avec précision. Pour tout objet dont la distance est inférieure à 2 mètres, le phénomène inverse est observé et le signal devient totalement saturé. La publication récente des distances recommandées confirme cette limitation technique : la distance pour pouvoir jouer à deux joueurs est de 2,4 mètres.

L'avantage de l'utilisation de la lampe *IR* est de pouvoir jouer dans toutes les conditions de luminosité ! Même si certaines fonctionnalités devraient être affectées comme la reconnaissance faciale ou le scan des objets (utilisant la caméra *RGB*, dépendante du spectre de lumière visible).
En conclusion la lampe infrarouge va projeter ses rayons sur la scène, la caméra infrarouge va ensuite filmer cette scène et la puce *PS1080 SoC* va traiter les données afin d'estimer la distance de chaque objet par rapport à la caméra.

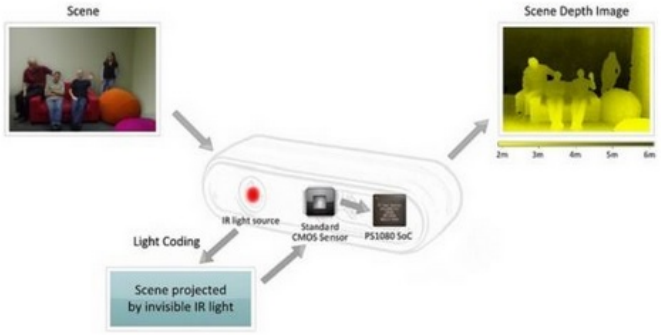

Figure K.8: Schéma résume la technologie Kinect

K.1.3 Le réseau des microphones

la Kinect embarquera vraisemblablement 2 microphones ainsi que 4 détecteurs digitaux externes de sources audio. La combinaison de l'ensemble de ce système audio permet ainsi à *la Kinect* de détecter la localisation spatiale d'une source sonore mais aussi d'éliminer les bruits de fond parasites grâce à un traitement de données efficace.

Annexe L

Algorithme ≪ Mean Shift ≫

L'algorithme *Mean Shift* est une procédure itérative de recherche de maximum locaux (appelés modes) de la densité de probabilité d'un échantillon de données, la procédure de la recherche des maximum locaux revient à la recherche des zéros du gradient de la densité, la méthode *Mean Shift* qu'est un estimateur du gradient de densité non paramétrique développé par *Fukunaga* et *Hostetler* en 1975 [] et exploité récemment par *Comainiciu et Meer* [2, 3] pour le traitement d'image. Le cadre *Mean Shift* est intéressent car il prend en compte simultanément des informations spatiales (posistion des pixels dans le domaine spatiale R^s) et d'amplitude (niveau de gris, couleur ou information spectrale dans le domaine d'amplitude R^r). Le domaine résultant est représenté par un espace euclidien R^d de dimension d, avec $d = s + r$. La méthode *Mean Shift* a l'avantage de ne reposer sur aucun a priori sur la distribution des intensités des pixels.

L.0.4 Mode

mode ou bien maximum local d'une fonction de densité de probabilité est par définition la valeur dont la réalisation est la plus probable.

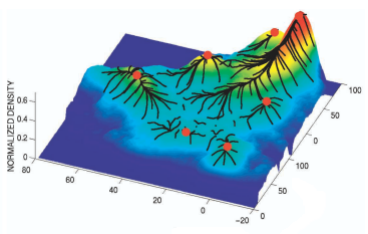

Figure L.1: les modes d'une densité de probabilité.

L.1 Principe de l'algorithme

La méthode *mean shift* est décrite dans [3]. C'est une technique d'estimation non paramétrique basée sur l'utilisation d'un noyau. Elle consiste à détecter les modes de la fonction de densité. L'algorithme *mean shift* est basé sur l'estimation multidimensionnel de densité pour un ensemble de n point $x_{i(}i = 1, ...n)$ dans R^d :

$$\hat{f}(x) = \frac{1}{n}\sum_{i=1}^{n} K_H.(x - x_i) \tag{L.1}$$

Le noyau K est une fonction symétrique, positive ou nulle, centrée sur 0 et dont l'intégrale vaut 1. H est une matrice de largeurs de bande, symétrique, définie positive qui normalise le support du noyau :

$$K_H(u) = (det[H])^{-1/2}.K(H^{-1/2}.u) \tag{L.2}$$

Le gradient de l'estimation non paramétrique par noyau est donné par :

$$\hat{\nabla} f(x) \equiv \nabla \hat{f(x)} = \frac{1}{n}\sum_{i=1}^{n} \nabla K_H(x - x_j) \tag{L.3}$$

L.2 L'expression du gradient pour les noyaux simples

Avec la forme K_H donnée précédemment, le gradient s'exprime :

$$\nabla K_h(x) = |H|^{-1/2}.\nabla K(H^{-1/2}.x) \tag{L.4}$$

Si le noyau k a comme profil la fonction k, définie tel que :

$$K(x) = C_{k,d}.k(x^T.x) \tag{L.5}$$

Alors la forme K_H devient avec k' est la dérivé de k:

$$\nabla K_h(x) = 2.C_{k.d}|H|^{-1/2}.H^{-1}.x.k'(x^T.H^{-1}.x) \tag{L.6}$$

Alors l'expression du gradient devient :

$$\nabla \hat{f}(x) = \frac{2.C_{k,d}.H^{-1}}{n.|H|^{1/2}}\sum_{i=1}^{n}(x - x_i).k'[(x - x_i)^T.H^{-1}(x - x_i)]$$

En remplaçant $g(x) = -k'(x)$:

$$\nabla \hat{f}(x) = \frac{2.C_{k,d}.H^{-1}}{n.|H|^{1/2}}.\sum_{i=1}^{n} g(d^2(x, x_i, H)) \left(\frac{\sum_{i=1}^{n} g(d^2(x, x_i, H)).x_i}{\sum_{i=1}^{n} g(d^2(x, x_i, H))} - x \right) \tag{L.7}$$

Ou d représente la distance de *Mahalanobis* :

$$d(x, x_i, H) = ((x - x_i)^T.H^-1(x - xi))^{1/2} \tag{L.8}$$

L'objectif est de trouver les valeurs de x pour lesquelles le vecteur gradient de l'estimation est nul, cette condition est vrai si le vecteur entre crochet dans la dernière expression du gradient est nul ce dernier est appelé vecteur *mean shift* :

Le vecteur *mean shift* $M(x)$ est défini par :

$$M(x) = [\sum_{i=1}^{n} g(d[x, x_i, H]^2)]^{-1}.\sum_{i=1}^{n} .x_i.g(d[x, xj, H]^2) - x$$

donc la solution pour l'annulation du gradient est se fait par itération :
expression A:

$$x^{t+1} = \frac{\sum_{i=1}^{n} g(d^2(x^t, xi, H)).x_i}{\sum_{i=1}^{n} g(d^2(x^t, xi, H))}$$

avec $x^0 = x$.

L.2.1 Expression pour les noyaux composés

Les expressions précédentes ne sont pas valables si les noyaux multidimensionnels sont composés par de produits de noyaux.
Le gradient devient un vecteur à d dimension décomposé en c composantes. Les expression prennent la nouvelle forme suivante :
pour tout $j = 1..c$

$$\nabla K_H(x) = 2.|H|^{-1/2}(\prod_{i=1}^{c} C_{k_j,d_i}).k_H(x).(\frac{H_j^{-1}.x_j.k_j'(x_j^T.H^{-1}.x_j}{k_j.(x_j^T.H_j^{-1}.x_j)})$$

avec $k_H(x) = \prod_{j=1}^{c} .k_j(x_j^T.H_j^{-1}.x_j)$

$$\nabla \hat{f}(x) = \frac{2.(\prod_{i=1}^{c} C_{k_j,d_i})}{n.|H|^{1/2}}.(H_j^{-1}.\sum_{i=1}^{n} N(x_j, x_{i,j}, k_h))[\frac{\sum_{i=1}^{n} N(x_j, x_{i,j}, k_h).x_{i,j}}{\sum_{i=1}^{n} N(x_j, x_{i,j}, k_h)} - x_j]$$

Pour alléger l'expression on met :

$$N(x_j, x_{i,j}, k_h) = k_h(x - xi).\frac{g_j(d^2(x_j, x_{i,j}, H_j))}{k_j(d^2(x_j, x_{i,j}, H_j))}$$

donc la solution pour l'annulation du gradient est se fait par itération :
expression B:

$$x_j^{t+1} = \frac{\sum_{i=1}^{n} N(x_j^t, x_{i,j}, k_h).x_{i,j}}{\sum_{i=1}^{n} N(x_j^t, x_{(i,j)}, k_H)}$$

avec $x_j^{[0]} = x_j$ et $j = 1..c$ A chaque itération, on remplace le point courant x par la moyenne des écarts entre x et les points x_i. Autrement dit, à chaque itération, le point x est décalé sur le centre de gravité -pondéré- de l'ensemble des points $x_(i)$. Le vecteur *mean shift* peut être aussi définit comme étant la différence entre le point x à $t+1$ et à t.

$$M_H(x^{[t]}) = x^{[t+1]} - x^{[t]}$$

ci-après est décrit la procédure *mean shift* originale de *Fukunaga* (aussi appelée *mean shift simple*)

---

soit X l'ensemble des $x_{i(i=1,\ldots n)}$ dans R^d donc un échantillon. $t = 0$
répéter :
pour chaque point de X :
calculer $x_i^{([t+1])}$ en utilisant l'équation A ou B.
$t = t + 1$
Tant que $||M(x_i^{t-1})|| > \epsilon$, pour tout $i = 1.\ldots\ldots n$ et ϵ le seuil de déplacement minimum.
Finalement $x_i^{([conv])} = x_i^{[t]}$
