الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne démocratique et populaire

وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

جامعة سعد دحلب البليدة
Université SAAD DAHLAB de BLIDA

كلية التكنولوجيا
Faculté de Technologie

قسم الإلكترونيك
Département d'Électronique

Mémoire de Projet de Fin d'Études

présenté par

Kachouane Mouloud

&

Sahki Safia

pour l'obtention du diplôme de Master en Électronique spécialité Systèmes de Vision et Robotique

Thème

Détection et poursuite visuelle de piétons en temps réel pour robot de type voiture

Proposé par : Ouadah Noureddine & Namane Abderrahmane

Année Universitaire 2011-2012

Remerciements :

Nous tenons avant tout à remercier Mr N. Ouadah qui nous a permis de poursuivre notre projet au sein du CDTA ainsi que Mr M. Lakrouf , leur présence et leur aide nous a été très précieuse. Sans oublier toute l'équipe NCRM de la division robotique pour leur précieuse aide.

Nos remerciements les plus profonds à Mr A. Namane, pour ses directives et son soutien, non seulement durant notre projet de fin d'études, mais tout au long de notre formation de master. Notre profonde gratitude à l'ensemble des enseignants de master Systèmes de Vision et Robotique pour leur dévouement et leurs directives tout au long de nos années d'études.

Un grand merci également aux étudiants avec qui on a travaillé sur le « ROBUCAR », le personnel de CDTA et tous ceux qui ont contribué de prés ou de loin à l'élaboration de ce travail.

Nous ne pouvons clore cette partie sans remercier nos familles, et plus particulièrement nos parents pour nous avoir supportés et soutenus avec beaucoup de patience et de tendresse.

ملخص

يندرج العمل المعروض في هذه المذكرة في إطار البحث عن المشاة في وقت آني, عن طريق كاميرا مركبة على روبوت متحرك ذات تحكم ذاتي من نوع سيارة. من خلال استعمال أنظمة الرؤية عبر الكمبيوتر,قمنا باختبار خوارزميات خاصة بالبحث, تعتمد على التدريب بقاعدة بيانات غنية. ثم يتم ترجمة هذه الخوارزميات الى برامج C++/C باستخدام مكتبة OpenVC ،في نظام لينكس ,ثم تنفيذه في البرنامج الذي يسير هندسة متوازية الوحدات الذي يسير الروبوت GenoM.

في المرحلة الاخيرة , قمنا بدمج هذه الوحدة النمطية للبحث في مهمة روبوتية لتحسين فعاليتها وتقييم أدائها.

كلمات المفاتيح :

البحث عن المشاة في وقت آني ، روبوت متحرك ذات تحكم ذاتي ، كاميرا مركبة ، أنظمة الرؤية عبر الكمبيوتر, التدريب, OpenCV , GenoM.

Résumé :

Le travail présenté dans ce mémoire s'inscrit dans le cadre de la détection de piétons en temps réel pour un robot mobile autonome doté d'une caméra embarquée. A travers la vision par ordinateur, nous avons testé des algorithmes de détection se basant sur un apprentissage d'une base de données riche. Ces algorithmes sont ensuite traduit en programmes C/C++ utilisant la bibliothèque OpenCV sous Linux, puis implémentés sous GenoM, le logiciel qui gère l'architecture modulaire de commande du robot. En dernier stade, nous avons intégré ce module de détection dans une tâche robotique (GenoM) , afin d'améliorer son efficacité et évaluer ses performances.

Mots clés :

Détection piétons en temps réel, robot mobile autonome, caméra CCD, vision par ordinateur, apprentissage, OpenCV, GenoM.

Abstract:

The work presented in this memory is part of the human target detection "pedestrians" in real time for an autonomous mobile robot type car, whose perception is through a board monocular camera. Through computer vision, with minimal processing time, we needed a reliable algorithm, and then do an apprenticeship with a rich database. Then we written a C/C++ program in Linux using the OpenCV library and implement it in GenoM, the modular architecture software that manages the robot. At last, we improved detection by testing various parameters; finally, we used our module in a robotic task test and improve efficiency.

Keywords:

Pedestrian target detection inreal time, autonomous mobile robot, monocular camera, computer vision, learning, OpenCV, GenoM.

Liste des acronymes et abréviations :

ALTILF : Le Trésor de la langue française informatisé.

CCD : Charge-Coupled Device.

CDTA : centre de développement des technologies avancées

CMOS : Complementary Metal Oxide Semiconductor.

CVPR : Computer Vision and Pattern Recognition.

GenoM : Generator Of Modules.

HOG : Histogramme de Gradient Orienté.

INRIA : Institut de recherche en informatique et en automatique.

LAAS : Le Laboratoire d'Analyse et d'Architecture des Systèmes.

LBP : Local Binary Patterns (Motifs Binaires locaux).

LGP : Local Gradient Patterns (Modèle de Gradient Local).

MVS/SVM : Machines à Vecteur de Support.

NCRM : Navigation et Contrôle des Robots Mobiles Autonomes.

OpenCV : Open Source Computer Vision.

RGB/RVB : Red, Green, Blue/Rouge, Vert, Bleu.

SDI/F : Structure de Données Internes/fonctionnelle.

SIFT : Scale-Invariant Feature Transform.

SURF : Speeded Up Robust Feature

UVC : USB Video device Class.

VAO : Vision Assistée par Ordinateur.

VIAM : The Versatile Image Acquisition Module.

V4L : Video For Linux.

Table des matières

Chapitre II : Détection de cibles humaines

Chapitre III : Implémentation, tests et résultats

Liste des tableaux

Liste des figures

Introduction

Dans la nature, la perception visuelle est adaptée au mode de vie de chaque espèce animale, lui conférant des aptitudes particulières, différentes de celles des hommes.

Les yeux de l'aigle royal sont dotés d'un puissant système optique d'agrandissement d'images. Ils lui permettent d'avoir deux visions de la scène observée. Il a une vision type grand-angle, bien utile lorsqu'il doit localiser ses proies. Une seconde vision lui permet de faire le point sur sa proie (en l'occurrence, un lièvre) en zoomant l'image. Le facteur d'agrandissement est alors de quatre à huit fois celui du reste de l'œil. Cette vision est centrée sur les pattes en position de capture. Ce système de vision particulier, illustré par la figure 1, fournit à l'aigle des capacités de navigation adaptées à sa technique de chasse [1].

Figure 1 : Illustration du système de vision d'un aigle

Dans un contexte moins animalier, plus d'un tiers du cerveau humain est dédié au processus de la vision. Pas surprenant, puisque ce processus est la méthode la plus importante par laquelle le cerveau acquiert des informations sur le monde extérieur ! La vision chez l'homme est un processus dynamique pendant lequel les yeux échantillonnent en continu leur environnement.

La perception visuelle est aussi beaucoup étudiée et utilisée pour les systèmes artificiels de navigation, en particulier pour les véhicules dits intelligents, c'est-à-dire présentant un certain degré d'autonomie. La capacité d'un tel véhicule à se localiser et à se mouvoir dépend en grande partie de la connaissance de son environnement. La vision artificielle ou vision par ordinateur s'est avéré un moyen privilégié.

Parmi les outils les plus difficiles en vision par ordinateur et les plus utilisés pour ces véhicules, la reconnaissance automatique d'objets. Elle est en même temps une étape primordiale pour la mise en œuvre de plusieurs applications actuelles qui nécessitent une

interprétation de haut niveau d'images. Par conséquent, il existe un intérêt croissant sur ce domaine de recherche dans les dernières années et une vaste littérature.

L'étude menée dans ce projet, qui a été conduit dans l'équipe NCRM de la division « Productique et Robotique » du centre de développement des technologies avancées (CDTA), est basée sur la conception et le développement d'un module de détection et suivi de piétons pour un robot mobile de type voiture « Robucar », équipé d'une caméra embarquée.

Un tel module doit répondre aux exigences suivantes :

- *Opérationnel en continu.*
- *Filtre d'information* : Un système de vision active agit comme un filtre d'informations, ne retenant que ce qui est pertinent pour la tâche désirée.
- *Temps réel* : Afin d'être utile, un système de vision active doit retourner les résultats dans un délai fixe, qui dépend de l'application.

Ce mémoire est organisé en trois chapitres : Dans le premier, on présente des généralités sur la robotique mobile autonome et la vision assisté par ordinateur, ainsi que quelques outils de traitement d'image utilisé dans la conception de notre application. En deuxième chapitre, on expose l'état de l'art des méthodes de détection de cibles humaine, en démontrant que ce type de problème est d'actualité. Une étude comparative a été effectuée, afin de pouvoir sélectionner les méthodes répondant au mieux aux contraintes imposées dans ce travail. Le dernier chapitre a été réservé à l'étude puis l'implémentation des algorithmes choisis sous C/C++, en utilisant la librairie OpenCV, puis sur le robot, ainsi que les tests effectués pour optimiser notre application et les résultats obtenus. Enfin, on présente une conclusion générale résumant l'ensemble des travaux réalisés.

Chapitre 1 Robotique Mobile Autonome & VAO

I.1. Introduction

Qu'est-ce que la vision ?

« Voir » c'est, entre autres, discerner et reconnaître les formes, les couleurs et les textures du monde qui nous entoure.

On pourrait croire, à tort, que nous avons besoin uniquement de nos yeux pour cela, mais la réalité est tout autre : nos yeux ne servent qu'à capter le signal contenu dans les rayons lumineux qui viennent frapper nos rétines, pas à en extraire des informations. Ça, c'est le travail de la zone de notre cerveau que nous appelons le cortex visuel, cela veut dire que la vision, c'est l'association entre le sens de la vue et un ensemble de processus cognitifs pour lesquels nous avons besoin d'utiliser notre cerveau.

Aujourd'hui, cela fait déjà bien longtemps que nous avons donné le sens de la vue aux ordinateurs (et notamment aux robots) grâce aux caméras numériques, mais si l'on se fie à la définition que nous venons de formuler, cela ne suffit pas à les doter de vision : il leur manque encore la capacité à extraire des informations des images et des flux vidéo, de manière à percevoir, analyser et comprendre le monde qu'ils observent.

C'est ici qu'intervient ce vaste domaine de recherche qu'est la vision par ordinateur, à la croisée des chemins entre les mathématiques, le traitement du signal et l'intelligence artificielle.

La vision par ordinateur se base essentiellement sur le traitement d'image pour effectuer des taches comme la segmentation d'objets, la reconnaissance de formes, la détection du mouvement, le suivi de cibles et autres techniques. Pour pouvoir effectuer ces taches, on utilise des outils qu'on appelle : outils de traitement d'images, donc on parlera de quelques notions de base de traitement d'image, plus précisément les outils qu'on utilise au cours de notre projet (la détection de cibles), mais avant on présente ce qu'est un robot mobile autonome, puisque on devra par la suite implémenter notre travail sur un robot à ordinateur embarqué.

I.2. Robot mobile autonome

I.2.1. Définition

La robotique est l'ensemble des techniques permettant la conception, la réalisation de machines automatiques ou de robots [2]. L'ATILF[1] définit le robot de la manière suivante : "Appareil effectuant, grâce à un système de commande automatique à base de micro-processeur, une tâche précise pour laquelle il a été conçu dans le domaine industriel, scientifique ou domestique" [3].

Le terme « robot » apparaît pour la première fois dans une pièce de Karel Capek en 1920, et vient du tchèque « robota » [4], Dans cet ouvrage les robots sont considérés comme des serviteurs dociles et efficaces pour réaliser des taches pénibles [4]. La tortue construite par Grey Walter dans les années 1950, est l'un des tout premiers robots mobiles autonomes [5]. Grey Walter n'utilise que quelques composants analogiques, dont des tubes à vide, mais son robot est capable de se diriger vers une lumière qui marque un but, de s'arrêter face à des obstacles et de recharger ses batteries lorsqu'il arrive dans sa niche. Toutes ces fonctions sont réalisées dans un environnement entièrement préparé, mais restent des fonctions de base qui sont toujours sujets de recherche pour les rendre de plus en plus génériques.

Un robot mobile est un système mécanique, électronique et informatique agissant physiquement sur son environnement en vue d'atteindre un objectif qui lui a été assigné. Cette machine est polyvalente et capable de s'adapter à certaines variations de ses conditions de fonctionnement. Elle est dotée de fonctions de perception, de décision et d'action. Ainsi, le robot devrait être capable d'effectuer des tâches diverses, de plusieurs manières, et accomplir correctement sa tâche, même s'il rencontre de nouvelles situations inattendues [5].

Ces systèmes sont dotés de trois groupes d'éléments :

- **Capteurs** : ou encore les senseurs sont des dispositifs utilisés pour capter des informations de l'environnement extérieur ou intérieur du robot.

[1] **ATILF :** Le Trésor de la langue française informatisé

- **Actionneurs** : les actionneurs sont des pièces électromécaniques ou électroniques qui permettent aux robots de produire des actions (mouvement, déplacement).

- **Systèmes de traitement** : les systèmes de traitement viennent établir le lien entre les capteurs et les actionneurs. Cette interface est réalisée à deux niveaux**,** au niveau matériel à partir d'un circuit à microprocesseur, et au niveau logiciel par un programme indiquant comment les informations issues des capteurs doivent être traitées pour contrôler les actionneurs.

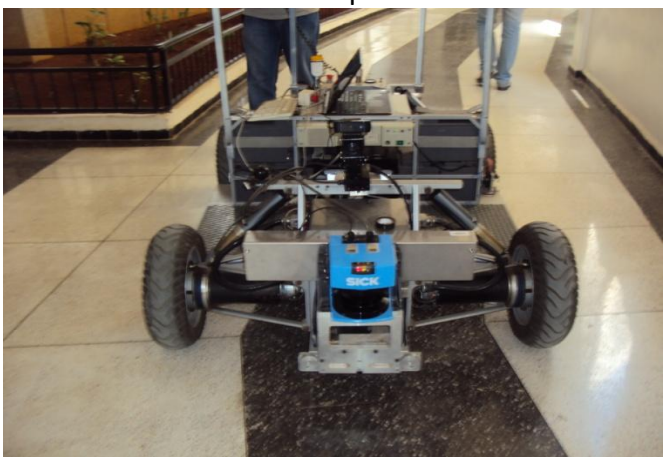


Figure I.2 : Robucar du CDTA, robot mobile autonome

I.2.2. Domaines d'utilisation

Aujourd'hui, le marché commercial de la robotique mobile est toujours relativement restreint, mais il existe de nombreuses perspectives de développement qui en feront probablement un domaine important dans le futur. Les applications des robots peuvent se trouver dans de nombreuses activités "ennuyeuses, salissantes ou dangereuses", mais également pour des applications ludiques ou de service, comme l'assistance aux personnes âgées ou handicapées. Parmi les domaines concernés, citons :

- L'industrie et l'agriculture : entrepôts, récolte de productions agricoles, mines ;
- Services : ménage, accueil ;
- Loisirs ;
-Transport : avions, bateaux et voitures autonomes ;
- Domaine spatial ;
- Militaire : sous-marins, drones… ;

A cela, s'ajoute à l'heure actuelle de nombreuses plates-formes conçues essentiellement pour les laboratoires de recherche. La figure I.2 montre quelques exemples de robots réels.

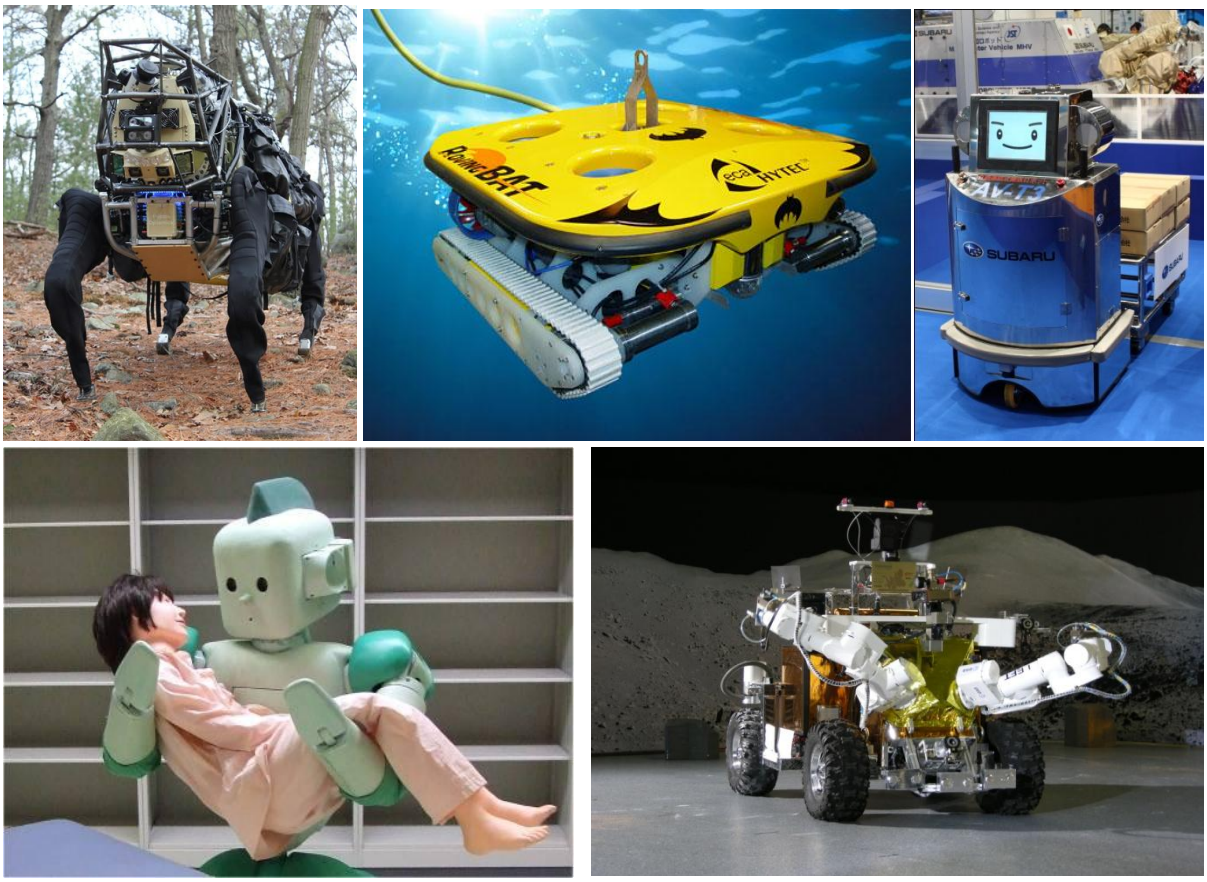


Figure I-3 : Exemples de robots utilisés pour différentes applications

I.2.3. Autonomie

Capacité propre d'un système sans équipage, à capter, percevoir, analyser, communiquer, planifier, prendre des décisions et agir afin d'atteindre les buts qui lui ont été assignés par un opérateur humain à l'aide d'une interface homme/machine dédiée. L'autonomie est échelonnée sur plusieurs niveaux, qui sont caractérisés par des facteurs incluant la complexité de la mission, les difficultés environnementales et le niveau d'interaction homme/robot nécessaire à l'accomplissement de la mission [6].

I.2.4. Versatilité[2]

Un robot doit être capable de réaliser plusieurs taches en même temps, ou la même tache de différentes manières.

[2] Versatilité : caractère versatile : changeant.

I.2.5. Notion d'holonomie

En robotique, une plateforme est dite holonome lorsque 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 planification de trajectoire [7].

I.2.6. Non holonomie

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, 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 [7].

I.2.7. La Perception

La notion de perception en robotique mobile est relative à la capacité du système à recueillir, traiter et mettre en forme des informations utiles au robot pour agir et réagir dans le monde qui l'entoure. Alors que pour des tâches de manipulation on peut considérer que l'environnement du robot est relativement structuré, ce n'est plus le cas lorsqu'il s'agit de naviguer de manière autonome dans des lieux partiellement connus. Aussi, pour extraire les informations utiles à l'accomplissement de sa tâche, il est nécessaire que le robot dispose de nombreux capteurs mesurant aussi bien son état interne que l'environnement dans lequel il évolue. Le choix des capteurs dépend bien évidemment de l'application envisagée. Pour se focaliser sur le problème de navigation [8].

a- Capteurs

Dans un système autonome, il est nécessaire de connaître en temps réel toutes les caractéristiques liées à la fois a la nature de l'environnement (état des surfaces, inclinaison du sol, présence ou non d'obstacles, …) et à celle du robot (coordonnées, position de tous les segments, inclinaison par rapport au sol, accélération, …).

Les capteurs sont capables de fournir dans des temps assez courts des informations très importantes sur l'état des lieux, et l'état du robot.

Les capteurs sont des composants de la chaîne d'acquisition dans une chaîne fonctionnelle. Ils prélèvent une information sur le comportement de la partie opérative et la transforme en une information exploitable par la partie commande [9].

On peut caractériser les capteurs selon deux critères [10]:

Placés sur le robot ils seront de deux natures différentes, soit ils informent le robot sur la nature de l'environnement, alors ce seront des capteurs ***extéroceptifs***, soit sur son état interne, alors ce seront des capteurs ***proprioceptifs*** [11].

De ces deux grandes familles de capteurs, on peut en extraire plusieurs types, selon leurs principes fonctionnement, la grandeur à mesurer (vitesse, position, distance…), ainsi que leurs mécanismes. L'arborescence suivante montre d'une manière générale, les capteurs existants ainsi que leurs classifications :

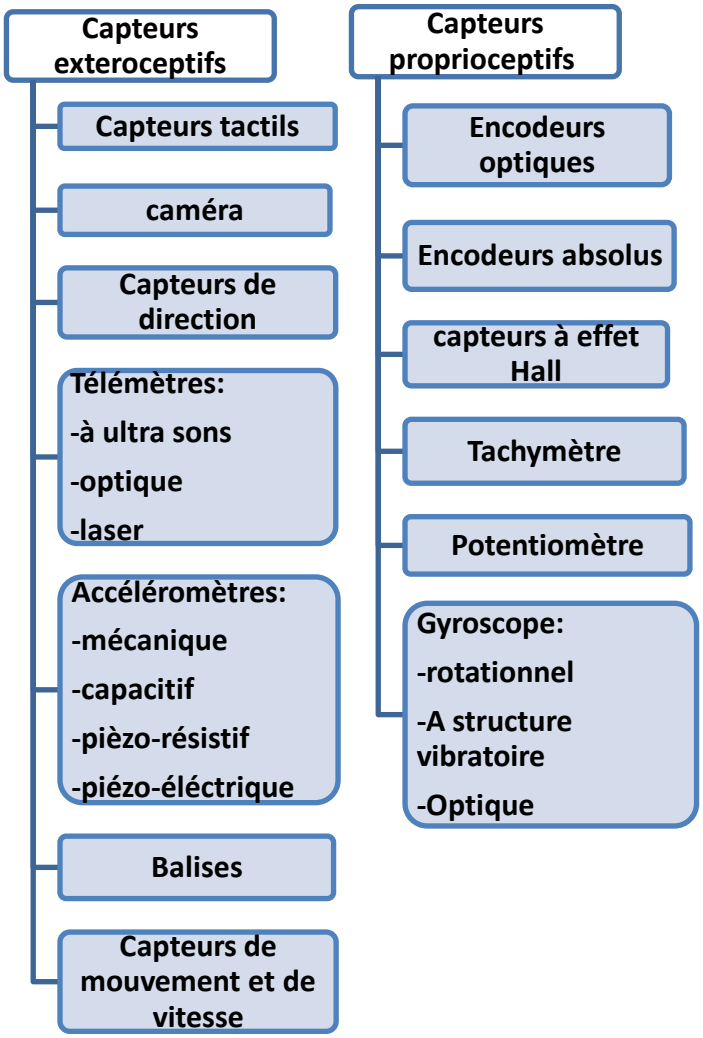


Figure I-4 : Types de capteurs

- **Capteurs proprioceptifs**

Les capteurs proprioceptifs sont installés à bord du robot et mesure les états internes de celui-ci (vitesse des roues, l'angle des articulations, la charge des batteries…) [12].

Encodeurs optique : C'est les capteurs les plus utilisés pour mesurer la vitesse ou la position angulaire de l'axe d'un moteur ou celle d'un système de guidage [12]. Le principe de fonctionnement de ces encodeurs est basé sur l'utilisation de photodiodes ou phototransistors qui deviennent 'passants' lorsqu'ils sont soumis à une certaine intensité lumineuse. Pour transformer la vitesse ou la position d'un robot en une donnée exploitable par un processeur, une source de lumière fixe est altérée par des zones opaques et transparentes d'un disque solidaire de ce robot.

Encodeurs absolus : Ce sont des encodeurs typiquement utilisés pour connaitre la position angulaire du disque tournant et ceci à n'importe quel instant, indépendamment d'une référence quelconque. Le principe du capteur repose sur l'utilisation d'un disque multipistes concentriques, alternées par des zones opaques et transparentes, le nombre de pistes détermine la résolution de l'encodeur [12].

Tachymètre : Bien que le terme tachymètre puisse désigner différent types de capteurs de vitesse, nous considérons dans notre cas qu'un tachymètre est le dispositif dont le principe de fonctionnement repose sur la mesure de la différence de potentiel produite par l'effet du mouvement des fils conducteurs à l'intérieur d'un champ magnétique [12]. La conception de ce mécanisme est semblable à celle d'un moteur à courant continu, sauf que dans ce cas nous voulons générer une tension à partir de la rotation de l'axe d'un moteur.

Potentiomètre : Lorsqu'il s'agit de mesurer l'angle de rotation d'une articulation, le plus simple des capteurs est constitué d'une résistance dont le curseur est directement lié à la partie mobile de sorte que la valeur de la résistance peut être mesurée en lisant la valeur de la résistance entre l'une des extrémités du potentiomètre et le curseur de ce dernier [12]. En connectant une source de tension continue aux bornes du potentiomètre, cette valeur sera proportionnelle à l'angle de rotation.

Gyroscope : Capteur de position angulaire par rapport à un référentiel inertiel (Galiléen) [12], nous pouvons classer les gyroscopes en trois catégories différentes : Gyroscopes rotationnels, Gyroscopes à structure vibratoire et Gyroscopes optiques.

- **Capteurs extéroceptifs**

Ces capteurs permettent au robot de mesurer les paramètres extérieurs à partir de l'environnement dans lequel il évolue, les types les plus utilisés sont :

Capteur de direction : Pour déterminer l'orientation d'un robot mobile, il existe des boussoles électroniques dont celle basées sur l'exploitation de l'effet Hall. Ce composant est constitué de deux capteurs à effet Hall disposés perpendiculairement l'un par rapport à l'autre [12]. Lorsque le robot se trouve à une position avec une orientation donnée, chacun des deux capteurs indique une tension entre ses bornes avec un signe et une intensité en fonction de la composante du champ magnétique terrestre qu'il peut capter.

Télémètre : Afin de connaitre la distance qui les sépare d'un objet ou obstacle, les robots mobiles sont souvent munis de capteurs de distance fonctionnant selon différents principes.

Accéléromètre : C'est un capteur qui, fixé à un mobile ou tout autre objet, permet de mesurer l'accélération de ce dernier. Il existe différents types, qui se base tous sur la loi fondamentale de la dynamique.

Balises : L'une des approches utilisées pour résoudre le problème de localisation en robotique mobile, consiste à utiliser des balises actives ou passives. En ce basant sur l'interaction entre les capteurs à son bord et des balises de l'environnement, le robot peut calculer sa position d'une manière assez précise [12]. Cette technologie permet aux robots de se localiser avec une précision jusqu'au centimètre dans des zones pouvant couvrir des kilomètres carrés.

Capteurs de mouvement et de vitesse : Certains capteurs permettent de mesurer directement le mouvement relatif entre le robot et son environnement. Puisque ces capteurs détectent un mouvement relatif à un repère lié au robot, son déplacement et sa vitesse peuvent être estimés. Il y a plusieurs capteurs permettant de mesurer certains aspects dus aux variations de mouvement. Par exemple un capteur pyroélectric détecte les changements de température lorsqu'un objet plus ou moins chaud se déplace à proximité du capteur.

- **Caméra**

La vision par ordinateur ou vision artificielle est la dernière née des disciplines relevant des théories de la perception visuelle. Elle est la science qui développe les

bases algorithmiques et théoriques grâce auxquelles l'information utile relative à l'environnement peut être automatiquement extraite et analysée à partir d'une image ou d'une séquence d'images [13].

Une caméra peut être utilisée de différentes manières pour la navigation d'un robot mobile. Elle peut être utilisée pour détecter des amers visuels (des points particuliers qui servent de repère, tels que des portes ou des affiches) à partir desquels il sera possible de calculer la position du robot [13], ainsi que tout type d'obstacle pouvant se trouver dans l'environnement qui entoure le robot, L'image acquise doit être traitée. On désigne par le traitement l'ensemble des techniques permettant de modifier une image numérique dans le but de l'améliorer ou d'en extraire des informations.

I.3. Vision Assisté par Ordinateur « VAO »

La vision nous permet de percevoir et d'interpréter le monde qui nous entoure. La VAO est un ensemble d'outils qui permet à l'ordinateur d'imiter la perception humaine afin d'extraire des informations d'une image brute pour pouvoir prendre des décisions. C'est un problème difficile en raison du fait que l'information disponible : des images 2D fournies par des capteurs (CCD, CMOS etc.), correspondent à une projection du monde 3D. La projection 3D-2D entraîne une perte d'informations importante, de plus, l'information disponible n'est pas parfaite (numérisation des capteurs, déformation des objectifs, bruitages). Nous pouvons alors dire que la VAO constitue une chaîne de traitements allant de l'acquisition de l'image brute jusqu'à son interprétation par la machine [14].

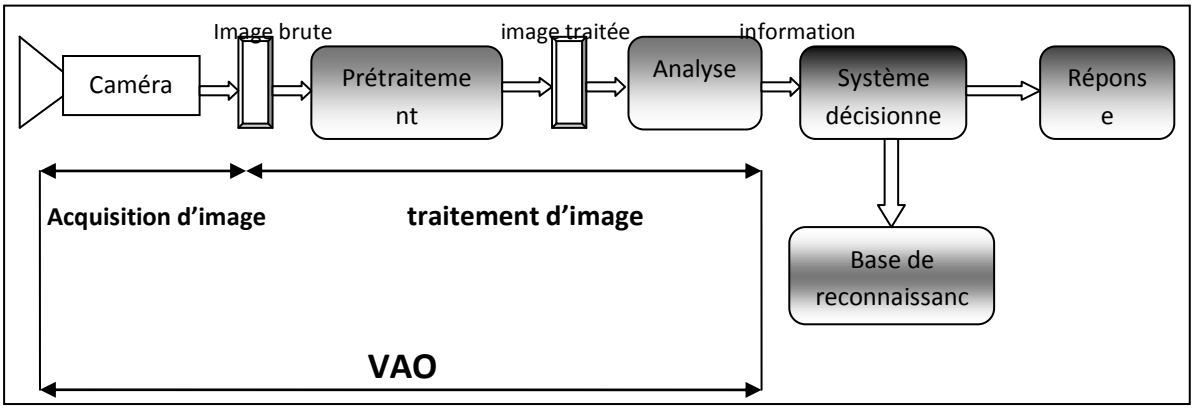


Figure I.5: La chaîne de VAO

I.3.1. Domaines d'applications VAO

- Contrôle de présence/absence : Sur des chaînes de production, on vérifie en bout de chaîne avec une caméra vidéo la présence d'une pièce dans un ensemble plus complexe.
- Construction et correction de cartes géographiques d'après des images satellites ou des images aériennes.
- Surveillance et évaluation de la production agricole. Il est possible de déterminer le degré de maturation des cultures, la quantité d'eau nécessaire pour l'irrigation, le rendement moyen etc.
- Reconnaissance de l'écriture.
- Recherche d'image par le contenu.
- Analyse de la vidéo
- Segmentation et suivi de cellules vivantes en microscopie.
- Aide à la conduite de voiture.
- En robotique : segmentation de l'environnement, détection d'obstacles, détection, reconnaissance et suivi de cibles etc.

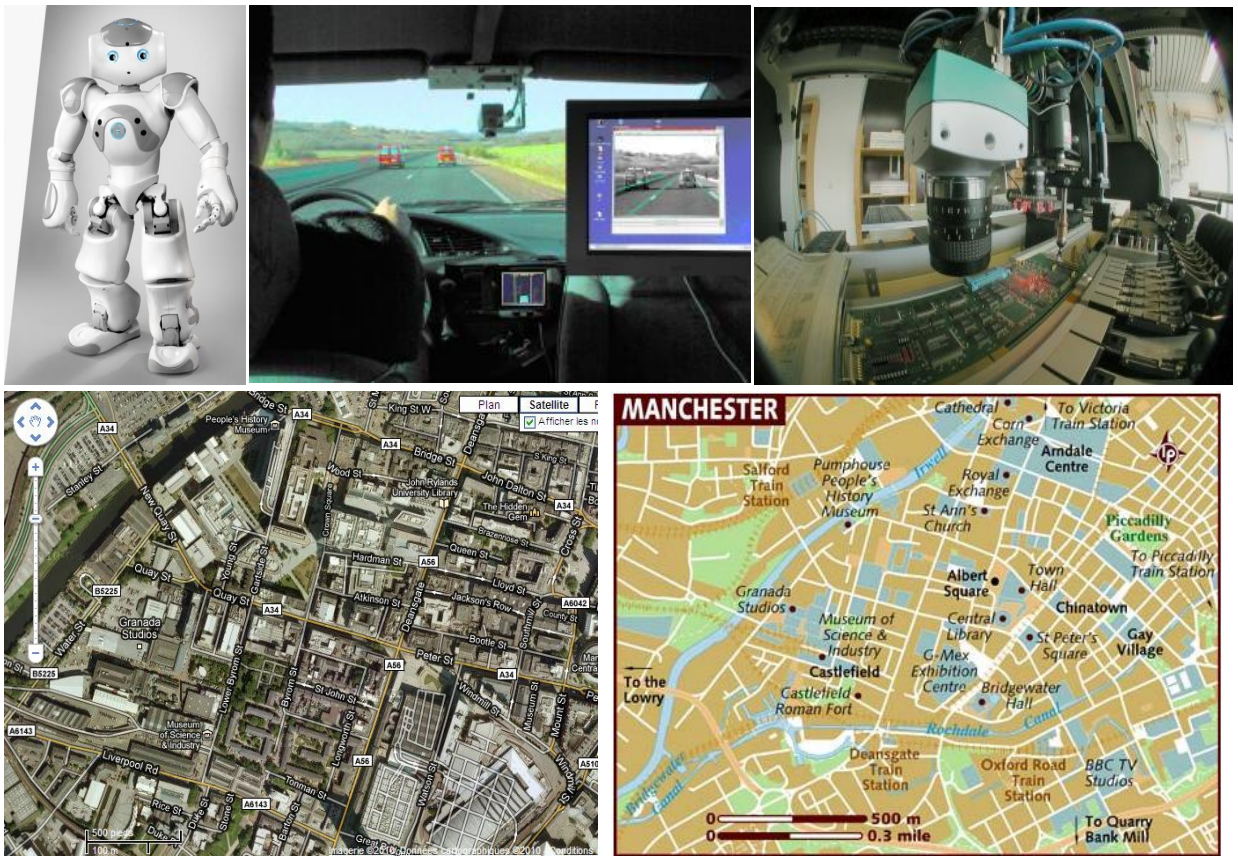


Figure I-6 : Exemples d'application VAO

I.3.2. Image numérique

Le terme image numérique désigne, dans son sens le plus général, toute image qui a été acquise, traitée et sauvegardée sous une forme codée représentable par des nombres (valeurs numériques).

La numérisation est le processus qui permet de passer de l'état d'image physique (tel que l'image optique) qui est caractérisée par l'aspect continu du signal qu'elle représente (une infinité de valeurs dans l'intensité lumineuse par exemple), à l'état d'image numérique qui est caractérisée par l'aspect discret (l'intensité lumineuse ne peut prendre que des valeurs quantifiées en un nombre fini de points distincts). C'est cette forme numérique qui permet une exploitation ultérieure par des outils logiciels sur ordinateur [15]. L'image en pixels se présente sous forme d'une grille constituée d'une multitude de points, comparable à une mosaïque : c'est ce qu'on appelle les pixels.

I.3.3. Types d'image

a- Image à niveau de gris

Le niveau de gris est la valeur de l'intensité lumineuse en un point, celle-ci varie de la lumière vers l'absence totale de la lumière. Une image en niveau de gris est une image composée de points gris plus ou moins foncés. Pour chaque point, l'ordinateur enregistre une valeur de gris entre le noir et le blanc. Par convention la valeur '0' représente le noir (intensité lumineuse nulle) et la valeur 255 le blanc (l'intensité lumineuse maximale), chaque pixel étant représenté par un octet, (en général on sauvegarde les images à 256 teintes de gris) [16].

Couleur	BLANC	GRIS 12.5%	GRIS 25%	GRIS 37.5%	GRIS 50%	GRIS 62.5%	GRIS 75%	GRIS 87.5%	NOIR
Niveau de gris	255	224	192	160	128	96	64	32	0

Tableau I-1 : Les niveaux de gris

b- Image binaire

Parmi les images numériques, les images binaires sont les plus simples. Bichromes (la plupart du temps noire et blanche) elles sont ontologiquement numériques c'est-à-dire que leur codage et leur décodage peuvent être faits directement vers la base 2.

c- Image Truecolor

Image Truecolor, ou image RGB, représente directement les valeurs de couleurs, au lieu d'être exprimée à travers une palette de couleurs ou colormap. Une image truecolor est un tableau de dimension m*n*3, avec m*n les dimensions de l'image et 3 représente les composantes de couleurs R, V et B. Donc pour chaque pixel (x,y) de l'image, la couleur est représentée par le triplet (x, y, 1:3).

R	V	B	Couleur
0	*0*	*0*	**Noir**
255	*0*	*0*	**Rouge**
0	*255*	*0*	**Vert**
0	*0*	*255*	**Bleu**
128	*128*	*128*	**Gris**
255	*255*	*255*	**Blanc**

Tableau I-2 : Les couleurs RVB

I.3.4. Traitement d'image

Le traitement d'images désigne une discipline de l'informatique et des mathématiques appliquées qui étudie les images numériques et leurs transformations, dans le but d'améliorer leur qualité ou d'en extraire de l'information. Il n'y a pas création d'informations, mais mise en évidence de l'information pertinente déjà présente.

a- Histogramme

Un histogramme est un graphique statistique permettant de représenter la distribution des intensités des pixels d'une image, c'est-à-dire le nombre de pixels pour chaque intensité lumineuse, il représente le niveau d'intensité en abscisse en allant du plus foncé au plus clair.

Ainsi, l'histogramme d'une image en 256 niveaux de gris sera représenté par un graphique possédant 256 valeurs en abscisses, et le nombre de pixels de l'image en ordonnées.

L'histogramme est le premier outil permettant de se rendre compte de la qualité intrinsèque d'une image, ainsi que l'électrocardiogramme qui permet de voir la qualité

d'information en fonction du niveau de densité et de vérifier si l'image présente des détails dans les ombres et dans les hautes lumières [17].

Pour les images en couleur plusieurs histogrammes sont nécessaires. Par exemple pour une image codée en RGB, on a un histogramme représentant la distribution de la luminance et trois histogrammes représentant respectivement la distribution des valeurs respectives des **composantes rouges, bleues et vertes [18].**

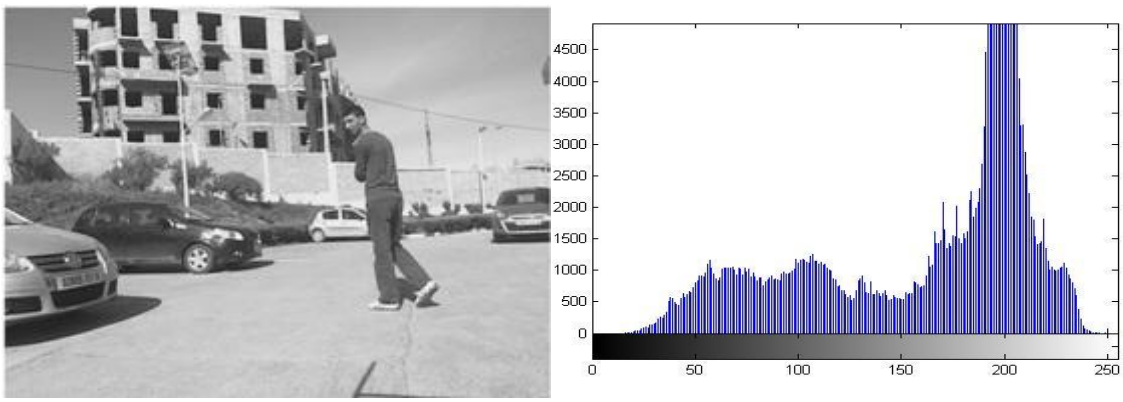


Figure I-7 : Image d'un piéton et son histogramme

- **Egalisation d'histogramme**

Elle a pour but d'harmoniser la répartition des niveaux de luminosité de l'image, de telle manière à tendre vers un même nombre de pixel pour chacun des niveaux de l'histogramme. Cette opération vise à augmenter les nuances dans l'image [18].

Figure I-8: Image piéton à niveau de gris avant et après égalisé

- **Rehaussement de contraste**

  Ou étirement de l'histogramme, on l'appelle aussi expansion de la dynamique, il consiste à recadrer l'image en étalant les fréquences d'apparition des pixels sur la largeur de l'histogramme, afin que la plus faible valeur soit à 0 et la plus haute soit à 255 (maximum).

Figure I-9: Image piéton en couleurs avant et après rehaussement de contraste

- **Correction gamma**

Cette correction rend l'image moins sombre, et corrige les erreurs de luminosité dûes à l'affichage sur ecran. On fait ça en décalant un peu l'histogramme vers la zone lumineuse. Ce prétraitement est fait pour avoir des images plus claire.

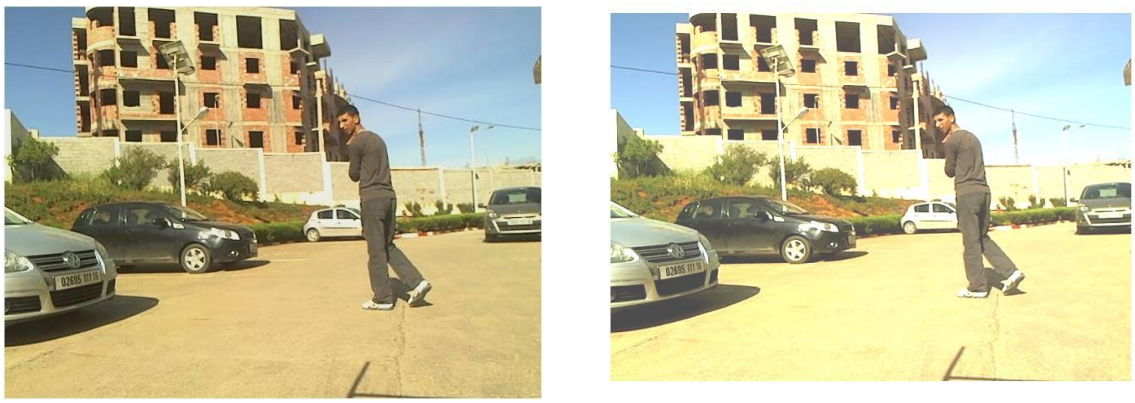


Figure I-10: Image piéton couleurs avec et sans correction gamma

b- Seuillage

Une image numérique, pour pouvoir être exploitée a généralement besoin d'être simplifiée, c'est le but du seuillage. Ce dernier consiste à transformer l'image codée sur 6, 8 ou 16 bits, en une image binaire ou les pixels à 1 correspondent aux objets et les pixels à 0 au fond de l'image. A la différence des autres traitements de l'image numérique, le seuillage est un passage obligatoire pour toute analyse morphologique ultérieure. Le seuillage permet de sélectionner les parties de l'image qui intéressent l'opérateur, par exemple 2 types de grains (blancs et sombres) dans un mélange. On peut donc, par exemple, attribuer à tous les pixels de l'image numérique qui ont un niveau de gris compris entre deux valeurs i_1 et i_2, choisies par l'opérateur, la valeur 1; à tous les autres pixels est attribuée la valeur 0. Après seuillage, les parties de l'image sélectionnées seront traduites en noir et blanc. L'image, digitalisée par l'ordinateur (0 et

1), est appelée image binaire. Cette dernière, tout comme l'image numérique contient des informations superfétatoires, qu'il convient d'éviter, ou masquées qu'il faut révéler [19].

Figure I-11: image d'un piéton binarisée

c- Segmentation

La segmentation, consiste à Identifier et définir des objets dans l'image en divisant l'image en zones homogènes afin de séparer les divers composants visibles et de les identifier. Deux approches sont nécessaires pour la réalisation de la segmentation, regroupement des pixels présentant une caractéristique commune, dérivée par exemple de l'intensité des pixels (région) et mise en place des frontières aux positions qui rendent localement maximale la variation d'un critère (contour). Ces deux conceptions sont duales : une région définit son contour, un contour définit une région [20].

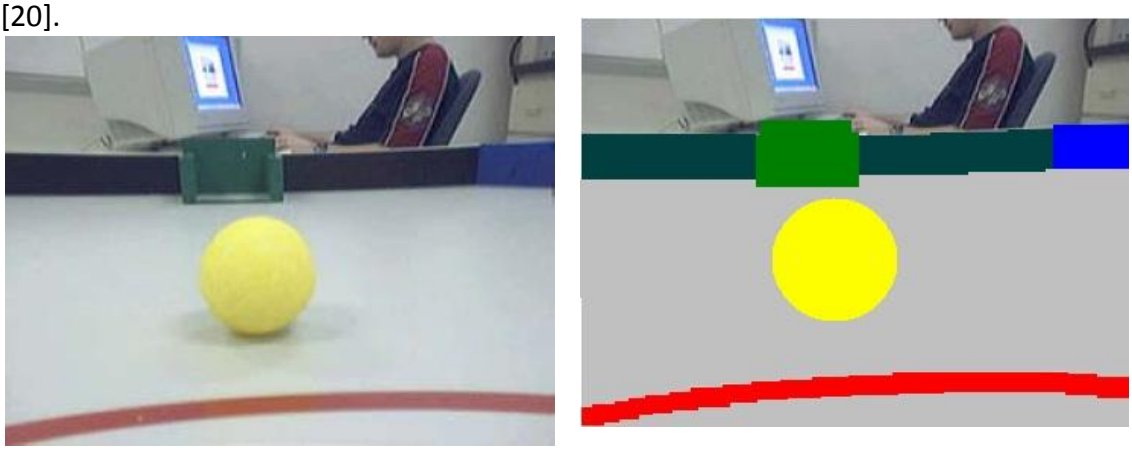


Figure I-12: Image prise d'une caméra embarquée sur robot segmentée

d- Opérateurs morphologiques

Les opérateurs morphologiques tels que la dilatation, l'érosion, la fermeture et l'ouverture sont utilisés pour l'amélioration des images binaires en éliminant les faux pixels considérés comme des bruits.

- Dilatation

La dilatation consiste à éliminer les points noirs isolés : on dilate les parties blanches ce qui élimine les points noirs. L'opération de dilatation est définit par [21]:

$$I \oplus S = \{x | [(\check{S})_x \cap I] \neq \phi\} \quad \text{(I-1)}$$

Ou I est l'image qui sera dilatée, S est un ensemble de points de coordonnées connues. L'effet de base de l'opérateur de dilatation sur une image binaire est d'agrandir progressivement les limites des régions noires de l'image. Donc les régions blanches de l'image grandissent en dimensions.

- Erosion

Il s'agit ici d'éliminer les points blancs isolés. La méthode est assez similaire à celle de la dilatation. L'opération d'érosion est définit par [21]:

$$I \ominus S = \{x | [(\check{S})_x \subseteq I]\} \quad \text{(I-2)}$$

L'effet de base de l'operateur de l'érosion est d'éroder les limites des régions de couleur blanche. Donc les régions d'images élémentaires de couleur blanche diminuent en dimensions, et les régions élémentaires de couleur noire deviennent plus grandes. La dilatation et l'érosion peuvent être combinées pour créer d'autres opérateurs morphologiques plus complexes pouvant résoudre une variété de problèmes [21].

– L'ouverture est une opération d'érosion suivie d'une dilatation, elle est définie par [20]:

$$B \circ I = (B \ominus I) \oplus I \quad \text{(I-3)}$$

– La fermeture est une opération de dilatation suivie d'une érosion, elle est donnée par [20]:

$$A \bullet I = (A \oplus I) \ominus I \quad \text{(I-4)}$$

e- Détection de contours

La détection est basée sur la dérivation selon les deux coordonnées. Si on considère classiquement les signaux comme des sommes de sinusoïdes, la dérivation apparaît comme un filtre passe-haut qui introduit donc du bruit à l'origine de faux contours. Pour l'amateur il est recommandé, avant d'utiliser un filtre simple, d'atténuer ce bruit par passage dans un filtre flou. Des méthodes plus élaborées ont été systématisées pour les professionnels.

- ***Filtre dérivées premières***. Le filtre le plus simple consiste à calculer les différences entre pixels voisins sur les horizontales puis sur les verticales. Chaque extremum correspond à un point d'un contour.

- ***Filtre de Sobel :*** la technique précédente est améliorée en remplaçant le filtre rectangulaire par un filtre triangulaire.
- ***Filtre dérivées secondes :*** celles-ci se calculent simplement en différences finies et c'est maintenant un changement de signe qui correspond à un point d'un contour. On les utilise généralement à travers leur somme qui est le laplacien.

f- Filtrage

Le principe du filtrage est de modifier la valeur des pixels d'une image, généralement dans le but d'améliorer son aspect. En pratique, il s'agit de créer une nouvelle image en se servant des valeurs des pixels de l'image d'origine. On peut citer quelques types de filtrage :

- *Filtrage locale*

Chaque pixel de la nouvelle image est calculé en prenant en compte seulement un voisinage du pixel correspondant dans l'image d'origine. Il est d'usage de choisir un voisinage carré et symétrique autour du pixel considéré. Ces voisinages sont donc assimilables à des tableaux à deux dimensions (matrices) de taille impaire.

- *Filtres linéaires*

Un filtre linéaire transforme un ensemble de données d'entrée en un ensemble de données de sortie selon une opération mathématique appelée convolution. Lorsqu'il s'agit de données numérisées comme dans le cas du traitement d'image, la relation entre les valeurs des pixels de sortie et celle des pixels d'entrée est décrite par un tableau de nombres, généralement carré, appelé matrice de convolution. Le temps de calcul est souvent réduit lorsqu'on veut séparer un filtre en deux filtres dont la convolution mutuelle permet de le reconstituer. Cette remarque est utilisée en particulier pour créer un filtre à deux dimensions à partir de deux filtres à une seule dimension (vecteurs) dans le sens horizontal et le sens vertical.

- *Lissage*

Ceux-ci sont des filtres passe-bas qui coupent plus ou moins les plus hautes fréquences. Ils sont utilisés pour atténuer les bruits d'origines les plus diverses qui polluent l'information, en particulier dans la détection de contours considérée ci-après.

Techniquement, il s'agit de traductions discrètes de filtres continus qui, comme ceux-ci, ne modifient pas le niveau global du signal. Les termes de la matrice de convolution sont donc généralement des entiers à diviser par leur somme.

I.3.5. Reconnaissance d'objets

La reconnaissance d'objets est une branche de la vision artificielle et un des piliers de la vision industrielle. Elle consiste à identifier des formes pré-décrites dans une image numérique, et par extension dans un flux vidéo numérique.

Il ne faut pas confondre reconnaissance d'objets (en anglais : « object recognition » ou « shape recognition ») et reconnaissance de formes (« pattern recognition » en anglais). La première s'attache à reconnaître des formes géométriques dans une image, alors que la seconde cherche à identifier des motifs dans des données statistiques. La confusion vient du fait qu'on utilise souvent la reconnaissance de formes comme technique appliquée à la reconnaissance d'objets.

Tout d'abord objet d'algorithmes dirigés par l'homme, jusqu'en 1995 (tentatives de reproduire par un algorithme un raisonnement humain d'identification, comme par exemple dans « un vélo possède deux roues, un cadre ... »), la reconnaissance d'objets a fait l'objet de progrès importants par la suite au travers de la mise en œuvre de techniques d'apprentissage, comme par exemple les séparateurs à vaste marge. Ces techniques visent à faire exploiter des bases d'exemples positifs et négatifs (contre-exemples) par un algorithme de recherche de critères discriminants, c'est-à-dire de critères permettant de séparer au mieux les exemples des contre-exemples.

I.4. Conclusion

La robotique mobile autonome ne cesse d'évoluer, et pour ce, l'utilisation de la Vision pour la perception, segmentation puis reconnaissance de l'environnement devient de plus en plus nécessaire pour augmenter l'intelligence du robot, pour arriver par la suite à éviter les obstacles ou poursuivre des cibles avec efficacité et précision.

Après avoir présenté les robots mobiles autonomes, ainsi que quelques notions sur la vision par ordinateur, on détaille en chapitre suivant une des techniques de la vision utilisé souvent en robot mobile qui est la détection de cible. On fait un état de l'art des dernières méthodes utilisé surtout pour la détection de cibles humaines.

Chapitre 2 Détection de cibles humaines

II.1. Introduction

Dans la littérature, de nombreuses méthodes de suivi d'objets ont été présentées ; une grande partie d'entre elles, peuvent être utilisées pour suivre des objets précis en temps réel. Malheureusement, la plupart de ces méthodes sont fondées sur un seul modèle de cible ou une seule modalité ; elles sont limitées à certains environnements spécifiques, typiquement pour la vidéo surveillance, des scènes statiques, contrôlés et connues à priori, sur lesquelles la méthode suit tous les objets dynamiques qui apparaissent.

On peut admettre, sans aucun risque de se tromper, que jusqu'à aujourd'hui il n'existe pas une méthode suffisamment générale pour traiter avec succès et robustesse, la grande variété des cibles et conditions qui existent dans le monde réel.

II.2. Méthodes de suivi

Plusieurs classifications des méthodes de suivi visuel d'objets ont été proposées dans la littérature ; elles dépendent autant des auteurs, que du but pour lequel ces méthodes ont été conçues. Nous considérons la classification donnée dans [22], où selon les auteurs, les méthodes de suivi visuel peuvent être divisées en quatre classes :

- Méthodes de suivi fondées sur des modèles. Ces méthodes repèrent des caractéristiques connues dans la scène et les utilisent pour mettre à jour la position de l'objet. Parmi ces méthodes, citons celles qui exploitent les modèles géométriques fixes [23], et les modèles déformables [24].
- Méthodes de suivi de régions ou blobs. Cette sorte de méthodes se caractérise par la définition des objets d'intérêt comme ceux qui sont extraits de la scène en utilisant des méthodes de segmentation. Citons les nombreuses méthodes qui détectent une cible à partir de son mouvement sur un fond statique ou quasiment statique [25].
- Méthodes de suivi à partir des mesures de la vitesse. Ces méthodes peuvent suivre les objets en exploitant les mesures de leur vitesse dans l'image, avec des mesures telles que le flux optique ou des équivalents [26].
- Méthodes de suivi de caractéristiques. Ces méthodes suivent certaines caractéristiques de l'objet, comme des points, des lignes, des contours … [27], caractéristiques ou primitives de l'image auxquelles il est possible aussi d'imposer

des restrictions globales [28]. Ces caractéristiques peuvent être aussi définies par la texture ou la couleur [29].

Cette classification n'est pas exhaustive, et à ce jour, il existe de nombreux recouvrements entre les classes, c'est-à-dire, des méthodes qui peuvent être classifiées dans deux classes ou plus. Nous considérerons que ces méthodes sont des combinaisons des approches existantes.

C'est à partir des définitions des environnements et des cibles qu'il est possible de s'apercevoir, que certaines méthodes, comme le suivi de blobs, seront difficilement utilisables. Par ailleurs, il est très difficile d'utiliser des méthodes fondées sur la différence objet/fond, parce que le robot est en mouvement et donc, le fond ou l'arrière plan n'est pas statique (du point de vue de l'image), et même les cibles peuvent être statiques par rapport au fond.

De manière similaire, les méthodes fondées sur des mesures de vitesse, seront difficilement exploitables pour la navigation de robots mobiles Il existe quelques approches pour le suivi à partir de flux optique, qui ont été essayées pour la navigation d'un robot [30] mais, la plupart d'entre elles utilisent une méthode de suivi de caractéristiques comme des lignes droites, et c'est à partir de ces primitives éparses dans l'image, que le calcul de la vitesse est fait.

Nous favorisons donc, l'utilisation des deux autres sortes de méthodes afin de réaliser les tâches de suivi depuis un robot se déplaçant dans les environnements décrits précédemment : suivi fondé sur un modèle de la cible et suivi de primitives d'images. Une analyse des méthodes de suivi ainsi que des méthodes de segmentation et reconnaissance des cibles pour la navigation robotique est décrit dans [31].

II.3. Détection de personne

II.3.1. Introduction

La détection de personne est un domaine de la vision par ordinateur consistant à détecter un humain dans une image numérique. C'est un cas particulier de la détection d'objet, où l'on cherche à détecter la présence et la localisation précise, dans une image, d'une ou plusieurs personnes, en général dans une posture proche de celle de la station debout ou de la marche. On parle également de détection de piéton, en raison de l'importance des applications en vidéosurveillance et surtout en notre cas, pour les systèmes de vision embarqués dans des véhicules.

Étudiée à partir de la fin des années 1990 [32], la détection de personne s'est révélée être un sujet assez difficile, en raison de la grande variété d'apparences des personnes, de l'articulation du corps humain et des phénomènes d'occultations. Bénéficiant des progrès méthodologiques réalisés en détection de visage, la détection de personne a inspiré des méthodes spécifiques, comme les histogrammes de gradient orienté, particulièrement performants. Les méthodes les plus efficaces construisent des modèles statistiques par apprentissage supervisé, à partir de caractéristiques de forme ou d'apparence, calculées sur de nombreux exemples d'images de personnes.

II.3.2. Problématique

La détection de personne est un sujet particulièrement difficile, en raison notamment de la grande variabilité d'apparences et de situations possibles :

- ✓ Grande variabilité de l'apparence des êtres humains, ainsi que de leurs vêtements.
- ✓ Articulation du corps humain (bras, jambes, torse).
- ✓ Occultations par des objets (mobilier urbain par exemple).
- ✓ Occultations par d'autres personnes et phénomènes de foule.

La détection doit s'effectuer dans des conditions difficiles et en environnement non constraint, en utilisant du matériel de prise de vue fournissant des images de faible qualité : caméra embarquée dans un véhicule.

Le problème est donc de trouver une représentation d'un humain qui soit à la fois suffisamment générique pour englober tous les types de situations, et suffisamment discriminante pour ne représenter que les humains [33]. Pour cela, on utilise en général une représentation intermédiaire, basée sur le calcul d'une ou plusieurs caractéristiques, permettant de mieux résumer l'information que les seules valeurs des pixels [33], [34].

II.3.3. Historique

Les premiers travaux sur la détection de personnes datent de la fin des années 1990 [35]. Dans l'une des premières méthodes proposées, la stéréovision est utilisée pour détecter des objets au moyen d'une transformée de Hough. La méthode peut détecter des piétons, mais n'est pas exclusive à ce type d'objet. En 1998, Heisele et Wöhler utilisent le mouvement des jambes des piétons pour réaliser la détection et la

classification, avec des contraintes sur la localisation des piétons par rapport au sol [35]. Ces méthodes restent cependant spécifiques à une application et peu génériques.

À partir des années 2000, le domaine bénéficie des avancées effectuées en détection de visage et notamment de la méthode de Viola et Jones, qui est étendue en 2005 à la détection de cibles en utilisant le mouvement [36]. La méthode permet une détection plus générique, ne nécessitant pas d'information a priori sur la structure de la scène, avec un temps d'exécution proche du temps-réel. La recherche se concentre sur ce type de méthode de détection générique. En 2005, des chercheurs de l'INRIA proposent les histogrammes de gradient orienté (HOG) [37], dont les bons résultats en font rapidement une méthode standard [38]. En 2008, des chercheurs de l'Université Rutgers utilisent un descripteur construit comme une matrice de covariance, qui permet d'obtenir des performances encore meilleures, mais en environnement connu apriori seulement [39].

Également en 2008, des chercheurs de l'université de Pékin utilisent avec succès les motifs binaires locaux (LBP), un type de caractéristiques qui s'était révélé efficace en détection de visage surtout [40].

II.4. Techniques de détection de personnes

Il existe un grand nombre de techniques pour la détection de personnes dans une image ou une vidéo numérique, répondant aux différentes contraintes des applications.

Le point commun entre ces techniques est de calculer un certain nombre de caractéristiques à partir des pixels de l'image. Celles-ci peuvent être globales ou locales, et rendre compte d'informations de forme, de couleur, de texture ou de mouvement. Les caractéristiques des zones contenant des personnes sont en général utilisées par une méthode d'apprentissage supervisé, qui peut être l'implémentation d'un modèle génératif ou discriminant [38], pour déterminer un modèle de personne. Ce dernier sert alors à classer une zone de l'image comme étant une personne ou non, à partir d'un vecteur de caractéristiques calculé sur cette zone. Celle-ci peut être déterminée par un prétraitement, par exemple par la détection du mouvement au moyen d'une étape de soustraction de fond. Une alternative est d'explorer toutes les zones possibles de l'image, comme c'est le cas avec les méthodes de détection par fenêtre glissante.

II.4.1. Prétraitement

Certaines méthodes utilisent une étape de présélection des zones à analyser. Ceci permet de réduire la complexité algorithmique ou d'écarter d'emblée toute zone sans intérêt, en particulier en fonction de critères géométriques.

II.4.2. Utilisation du mouvement

Quand le système acquiert une vidéo, il est intéressant de tirer partie du mouvement de la personne pour faciliter sa détection. Une étape de segmentation spatiale de l'image en avant et arrière-plan est effectuée préalablement à la détection proprement dite [41].

II.4.3. Soustraction de fond

Elle consiste à estimer un modèle de l'arrière-plan (le fond), qu'il suffit ensuite de soustraire à l'image courante pour en obtenir les parties en mouvement [41]. Ce type de méthode est par ailleurs limité à des scénarios ou la caméra est fixe, ce qui est nécessaire afin d'estimer le fond, mais qui restreint en général leur usage à une application de vidéosurveillance.

II.4.4. Utilisation de variante du mean-shift

Cette méthode est dite simple et performante en temps réel, basée sur la rétroprojection ; elle est utilisée pour des scènes à arrière plan très distingué de l'objet à détecter vu sa sensibilité à la couleur du fond (peut précise en milieu complexe) [42].

Le principe est le suivant :

- Division de la région d'objet en partitions fixes et appliquer la méthode pour chacune d'elles.
- Calculer ensuite l'histogramme H_i de chaque partition qu'on appelle cellule sachant que H_i est l'histogramme de la distribution de la couleur et H_s l'histogramme d'une région :

$$W_i(x)=\sqrt{\frac{Hi(I(x))}{Hs(I(x))}} \qquad \text{II-5}$$

- Ajuster l'échelle en utilisant le coefficient de ***Bhattacharya***[1] ; on fait ensuite la somme des coefficients tel que :

$$\rho=\sum_{i=1}^{n}\sum_{k=1}^{m}\sqrt{b_k\, b_k^c} \qquad \text{II-6}$$

m: nombre d'histogrammes.

[1] Bhattacharya : une mesure statique du recouvrement de deux ensembles d'échantillons

b_k ; b_k^c : densité normalisée d'histogramme H_i et H^c_i.

Cette méthode consiste donc à estimer chaque itération des maximums locaux des densités de probabilité de la distribution multimodale.

II.4.5. Scale-invariant feature transform (SIFT)

Que l'on peut traduire par « transformation de caractéristiques visuelles invariante à l'échelle », est un algorithme utilisé dans le domaine de la vision par ordinateur pour détecter et identifier les éléments similaires entre différentes images numériques (éléments de paysages, objets, personnes, etc.). Il a été développé en 1999 par le chercheur David Lowe.

Figure II-13 : Application du SIFT detector sur une image

L'étape fondamentale de la méthode proposée par Lowe consiste à calculer ce que l'on appelle les « descripteurs SIFT » des images à étudier. Il s'agit d'informations numériques dérivées de l'analyse locale d'une image et qui caractérisent le contenu visuel de cette image de la façon la plus indépendante possible de l'échelle (« zoom » et résolution du capteur), du cadrage, de l'angle d'observation et de l'exposition (luminosité). Ainsi, deux photographies de la tour Eiffel auront toutes les chances d'avoir des descripteurs SIFT similaires, et ceci d'autant plus si les instants de prise de vue et les angles de vue sont proches. D'un autre côté, deux photographies de sujets très différents produiront selon toute vraisemblance des descripteurs SIFT très différents eux aussi (pouvoir discriminant). Cette robustesse, vérifiée dans la pratique, est une exigence fondamentale de la plupart des applications et explique en grande partie la popularité de la méthode SIFT.

Les applications de la méthode sont nombreuses et ne cessent de s'étendre ; elles couvrent au début du XXIe siècle des domaines tels que *la détection d'objet*, la cartographie et la navigation, l'assemblage de photos, la modélisation 3D, la recherche d'image par le contenu ou le *tracking video*.

II.4.6. Détection par fenêtre glissante

La technique la plus générique et aussi la plus employée est la détection par fenêtre glissante, consistant en un parcours exhaustif de l'image, par application du détecteur à de très nombreuses positions et échelles. La détection par fenêtre glissante est surtout utilisée dans des applications où l'information a priori sur le contenu est pauvre, en particulier quand le choix du capteur n'est pas maitrisé. Le balayage de l'ensemble de l'image est alors une méthode privilégiée [38].

II.4.7. Utilisation des caractéristiques

a- Caractéristiques globales

On peut trouver dans la littérature quelques exemples qui utilisent une représentation globale par exemple, Gavrilla [43] a proposé un système de détection de piétons appelé **PROTECTOR**, basé sur la détection des contours.

Parfois utilisée, elle reste cependant assez limitée dans les types d'images qu'elle peut traiter, et a le défaut d'être peu robustes aux occultations [33].

b- Pseudo-Haar

Les caractéristiques pseudo-Haar sont décrites pour la première fois dans un article de Paul Viola et Michael Jones paru en 2001 dans la revue scientifique International Journal of Computer Vision (IJCV), dans laquelle ils décrivent une nouvelle méthode de détection de visage [34]. Le cœur de cette méthode s'inspire des travaux de Papageorgiou, Oren et Poggio [44], qui ont décrit des caractéristiques construites à partir d'un ensemble d'ondelettes de Haar.

L'avantage déterminant des caractéristiques pseudo-Haar est la rapidité de leur calcul. Accouplées à la technique des images intégrales, elles peuvent être calculées en un temps constant d'environ 60 instructions de processeur pour une caractéristique à deux rectangles. En revanche, leur simplicité ne leur permet pas de retenir de l'information très complexe.

c- La méthode de Viola et Jones

La méthode de Viola et Jones est une méthode de détection d'objet dans une image numérique, proposée par les chercheurs Paul Viola et Michael Jones en 2001 [44]. Elle fait partie des toutes premières méthodes capables de détecter efficacement et en temps réel des objets dans une image. Inventée à l'origine pour détecter des visages, Viola et Jones l'on généralisé en 2005 pour détecter d'autres types de cibles, comme des voitures ou des avions et même des personnes.

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[2]. 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.

d- Les histogrammes de gradient orienté

Une amélioration importante des performances est obtenue par l'introduction de cette caractéristique, calculée à partir d'histogrammes de l'orientation du gradient de l'image, sur une grille dense avec des recouvrements. Cette représentation s'est révélée particulièrement efficace, bien qu'un peu lente lors de la détection. Utilisés avec un classifieur assez simple, basé sur les machines à vecteur de support (MVS).

Un histogramme de gradient orienté « HOG » est une caractéristique utilisée en vision par ordinateur pour la détection d'objet. La technique calcule des histogrammes locaux de l'orientation du gradient sur une grille dense, c'est-à dire sur des zones régulièrement réparties sur l'image. La méthode s'est montrée assez efficace pour la détection de personnes. Les HOG ont été proposés par Navneet Dalal et Bill Triggs, chercheurs à l'INRIA de Grenoble, à la conférence CVPR juin 2005 [37].

[2] Un classifieur est une fonction qui, retourne vrai si l'image contient l'objet d'intérêt et faux sinon

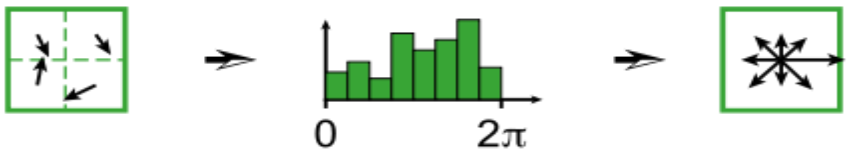


Figure II-14 : Orientation d'histogrammes de gradient

e- La covariance de région

C'est un descripteur construit comme une matrice de covariance de caractéristiques simples (intensité, gradient), calculée localement sur une région d'intérêt, à l'aide d'une image intégrale [45]. Ce descripteur est appliqué à la détection de personnes, en construisant les matrices de covariance de chaque région à partir d'un ensemble de 8 caractéristiques. La classification est réalisée à partir d'une cascade de classifieurs boostés, en utilisant une variante d'AdaBoost : logitBoost, et en modifiant l'algorithme de classification pour prendre en compte le fait que le descripteur appartient à une variété riemannienne et non à un espace vectoriel classique[39].

f- Les motifs binaires locaux

Les motifs binaires locaux (local binary patterns en anglais), Sont un type de caractéristiques construites comme un code binaire à partir du seuillage local des intensités. Les LBP permettent d'éviter certaines pertes d'informations liées à l'utilisation du gradient [40]. Des chercheurs de l'université de Pékin ont montré que moyennant une adaptation de la représentation des LBP au problème de détection de personne, ces caractéristiques permettaient d'obtenir des résultats supérieurs aux HOG et à la covariance de région [40]. Ce descripteur a été mentionné pour la première fois en 1993 pour mesurer le contraste local d'une image mais réellement popularisé trois ans plus tard par Ojala pour analyser les textures. Le principe général est de comparer le niveau de luminance d'un pixel avec les niveaux de ses voisins. Cela rend donc compte d'une information relative à des motifs réguliers dans l'image, autrement dit une texture. Selon l'échelle du voisinage utilisé, certaines zones d'intérêt tel que des coins ou des bords peuvent être détectées par ce descripteur.

II.4.8. Combinaison de caractéristiques

Plusieurs études ont montré que l'utilisation simultanée ou alternée de plusieurs types de descripteurs permettait une amélioration significative des résultats.

a- HOG et LBP

Cette combinaison permet d'obtenir des résultats supérieurs à une utilisation indépendante [46], connu sous le nom de LGP : Local Gradient Patterns (modèles de gradient local), à chaque bit de la LGP est attribué la valeur 1 si le gradient voisin d'un pixel donné est supérieure à la moyenne de huit gradients voisins, et 0 sinon. Cette représentation est insensible aux variations d'intensité globales comme les représentations d'autres tels que des motifs binaires locales (LBP). Cette méthode réduit le taux d'erreur de détection de faux positifs en accumulant les preuves à partir des résultats de détection multi-échelles. Cette méthode donne des résultats entre 5 et 25% meilleurs que ceux de LBP et HOG ; mais actuellement, impossible de l'implémenter en temps réel.

b- Combinée des HOG et des pseudo-Haar

Elle permet également d'obtenir de meilleurs résultats [47], [48]. L'idée sous-jacente à l'utilisation combinée de plusieurs caractéristiques est qu'elles encodent une information différente dans leur représentation (hautes ou basses fréquences...) et que ces informations sont plus ou moins pertinentes selon les cas [47].

La combinaison de descripteurs permet de pallier certaines déficiences d'une famille de descripteurs.

Il est également possible d'utiliser plusieurs familles de descripteurs de façon alternée, en utilisant par exemple les plus rapides au début de la détection et les plus complexes à la fin.

II.4.9. Détection par parties

Certaines méthodes modélisent une personne comme un ensemble de parties, et cherchent à détecter ces parties du corps indépendamment les unes des autres, avant de fusionner l'ensemble des indices.

a- Wu et Nevatia

Proposent des détecteurs à base de classifieurs faibles boostés, utilisant des caractéristiques locales basées sur la forme, nommées edglets. Ils séparent le corps en trois parties : tête-épaules, torse, jambes [49].

b- Mikolajczyk, Schmid et Zisserman

Divisent quant à eux une personne en sept parties, et entraînent pour chacune de ces parties un détecteur basé sur la méthode de Viola et Jones appliquée à des caractéristiques d'orientation similaires aux SIFT. La combinaison des réponses des détecteurs de parties est réalisée dans un cadre probabiliste bayésien [50].

Ces approches, grâce à leur fonctionnement par parties, sont certes construites pour être robustes aux occultations ainsi qu'aux variations d'apparences dues à l'articulation du corps humain, mais la détection d'une partie du corps reste une tâche très difficile.

II.5. Modèles pour la classification

La majorité des techniques de détection de personnes utilise un modèle statistique entraîné avec des vecteurs de caractéristiques calculés sur de nombreux exemples d'images de personnes. Le modèle détermine ainsi la fonction de décision qui permet de séparer au mieux les exemples positifs (représentant une personne) des exemples négatifs (le reste).

Les modèles discriminants cherchent à déterminer la meilleure frontière de décision permettant de trouver l'hyperplan optimal séparant les vecteurs positifs des vecteurs négatifs dans l'espace de représentation des vecteurs.

Les machines à vecteur de support font partie des modèles les plus employés et les plus efficaces dans ce domaine. Ils ont été utilisés pour la détection de personnes avec un noyau linéaire, des noyaux gaussiens ou polynomiaux [38].

Un autre type de modèle discriminant très utilisé est la cascade de classifieurs boostés, construite comme dans la méthode de Viola et Jones [38]. L'emploi d'un classifieur dans une structure en cascade permet d'obtenir des temps de détection très courts [51], les performances en détection dépendant des caractéristiques et des classifieurs faibles utilisés. Les réseaux de neurones peuvent également être employés, par exemple un perceptron multicouche modifié afin que les neurones cachés ne reçoivent leurs entrées que d'une région spatiale locale [38].

Des modèles génératifs ont également été utilisés, mais ils restent néanmoins moins répandus. Ceux-ci modélisent explicitement la loi jointe des observations et des classes. Certains auteurs utilisent un modèle génératif pour générer des échantillons afin d'améliorer l'apprentissage d'un modèle discriminant, qui est utilisé pour la classification proprement dite [51].

II.6. Évaluation et performances

Afin de connaitre précisément les performances des différents algorithmes, des bases d'images et de vidéos contenant des personnes ont été constituées et la position de chaque personne annotée manuellement. Ceci permet de juger les performances de l'algorithme de détection, par rapport à une « vérité terrain» estimée par un humain. Certaines de ces bases ont été rendues publiques, ce qui permet la comparaison entre plusieurs algorithmes. L'une des bases les plus utilisées quelle que soit l'application envisagée est la base INRIA, constituée par Dalal et Triggs en 2005 [52], [37], [48]. Des bases plus spécialisées existent, comme les bases Daimler, plus orientées vers la détection de piétons vus d'une automobile [53], [38]. La base Caltech est également destinée à cette application et contient 10 heures de vidéo, enregistrées à partir d'un véhicule en mouvement [48]. Ces bases doivent être représentatives de situations réelles, et contenir des cas difficiles : occultations, groupes de personnes, variations de luminosité, etc. Des bases spécifiques existent pour les données de stéréovision [54].

La mesure des performances peut se faire par image ou par fenêtre. La mesure par image consiste à mesurer la performance finale dans l'image : une personne est correctement détectée (vrai positif), manquée (faux négatif), ou le détecteur s'est trompé (faux positif, fausse alarme). Une détection est considérée comme correcte si elle partage une surface commune avec la vérité terrain supérieure à un seuil défini (un taux de 50 % est considéré comme raisonnable [48]). La mesure par fenêtre est spécifique aux algorithmes fonctionnant par fenêtre glissante : on mesure les performances dans chaque fenêtre testée, qui est soit positive ou négative. Cette mesure est assez utilisée en détection de personne, suite à son utilisation par Dalal et Triggs pour les HOG. Elle est toutefois critiquée par certains auteurs, qui lui reprochent

de ne pas mesurer véritablement la performance finale de la détection, ni d'être suffisamment générique, de nombreux algorithmes n'utilisent pas de fenêtres glissantes [48].

La détection de personnes est une tâche assez difficile, les performances des différents algorithmes restent assez modestes. Ainsi, les HOG, considérés comme l'une des meilleures méthodes existantes [48], [38], [47], peut obtenir un taux de détection assez élevée. Les performances dépendent toutefois beaucoup de la base de test et de la base d'apprentissage. Les algorithmes ont des difficultés à gérer les occultations, les faibles résolutions, les objets de ratio largeur/hauteur non standards [48]. Les méthodes obtenant les meilleurs résultats sont celles qui utilisent plusieurs types de caractéristiques différentes (HOG+Haar, HOG+LBP) [48], avec des temps de traitement trop élevées, ce qui ne permet pas de les implémenter en temps réel.

II.7. Conclusion

Après cette modeste étude ; On a dû choisir des méthodes et les implémenter pour détecter nos deux types de cibles : Piétons (personnes) et Visages.

On a pris en considération pour choisir ces méthodes plusieurs facteurs importants qu'on résume :

- La possibilité d'implémentation sous OpenCV puis sur le robot (sous GenoM).
- Le taux de détection assez élevé.
- Le taux de détection en milieu complexe (milieu urbain, avec différents taux de luminosité).
- La capacité de détection en mouvements, la caméra étant embarquée sur un véhicule mobile et les cibles (les humains) en déplacement aussi.
- Le temps de traitement et la possibilité d'implémentation en temps réel (ou proche du temps réel).

Enfin on a choisis d'essayer la méthode de Viola et Jones pour les deux types de cibles, HOG pour les piétons et LBP pour la détection de visages. Donc deux méthodes pour chaque type de cibles. On détaillera ces méthodes ainsi que leurs implémentations sous OpenCV et système embarqué du robot.

Chapitre 3

Implémentation tests & résultats

III.1 Introduction

Les algorithmes qu'on a choisis bien qu'assez robustes (selon les auteurs), utilisent des calculs un peu lents. Leur implémentation sur le robot nécessite une minimisation du temps de calcul, et c'est la caractéristique principale de la librairie OpenCV : traitement d'images en temps réel, et pour laquelle on l'a choisit ; surtout qu'elle est utilisable sous Linux : le système d'exploitation sous lequel on doit implémenter les algorithmes de commande du robot.

Dans ce travail nous avons comparé entre deux algorithmes pour chaque cible (piétons et visages), afin de choisir le plus robuste et qui respecte l'aspect temps réel.

III.2 Mise en situation

III.2.1 Définition de l'objectif

L'objectif de l'application est de permettre à un véhicule autonome « Robucar » de détecter-traquer les piétons dans le champ de vision (caméra embarquée), sachant qu'il évolue dans un milieu urbain, avec fonds d'images très variés (différente textures) et en présence de plusieurs objets fixes et mobiles : voitures, piétons, vélos, arbres, etc.

S'ajoute à ça, la différence de la luminance entre les endroits dus à l'ombre par exemple, les changements climatiques et l'heure de détection (journée ou la nuit). Ce travail représente une première étape pour la mise en place d'un module de détection, prédiction de chemin puis suivi ou évitement de piétons.

En évoluant dans notre travail, on a décidé d'ajouter une deuxième cible à détecter, et puisqu'on voulait suivre des cibles humaines, on a ajouté la détection de visages.

III.2.2 Temps de réponse et performances

Le module de détection s'exécute dans une architecture de commande du robot, donc on doit minimiser les calculs afin d'améliorer ses performances.

La conception doit s'orienter depuis le début vers les choix qui permettent au système d'être le plus rapide possible (temps réel), et avec taux de performance élevé, puisque le robot se déplace avec une grande vitesse. La tâche sera d'autant plus ardue qu'il s'agit là de deux éléments antagonistes ou l'amélioration de l'un se fait souvent au détriment de l'autre.

III.2.3 Matériels utilisés

Le matériel utilisé pour les expérimentations est composé d'un ordinateur portable et d'une caméra. Celles-ci intègrent la commande du robot et ont été faites sur un robot mobile autonome nommé 'Robucar' du CDTA.

Le Robucar est un prototype de petit véhicule électrique construit sur la base d'un châssis tubulaire des petites voitures utilisées dans les terrains de Golf. Il comporte une architecture parallèle permettant le développement des applications temps réel.

Figure III-15 : Le matériel disponible

Le Robucar comporte une tourelle contrôlable par ordinateur sur laquelle sera fixée une caméra. La tourelle à 2 axes utilisée est une tourelle PTU-D46-70. Elle est aussi appelée tourelle «Pan-Tilt » car elle est composée de deux parties:

- La partie « PAN » qui permet une rotation sur le plan horizontale.
- La partie « TILT » qui permet une rotation sur le plan verticale.

Figure III.16: Tourelle PTU-D46-70

III.3 Environnement logiciels

La figure ci-dessous montre l'architecture du système sur lequel on a travaillé.

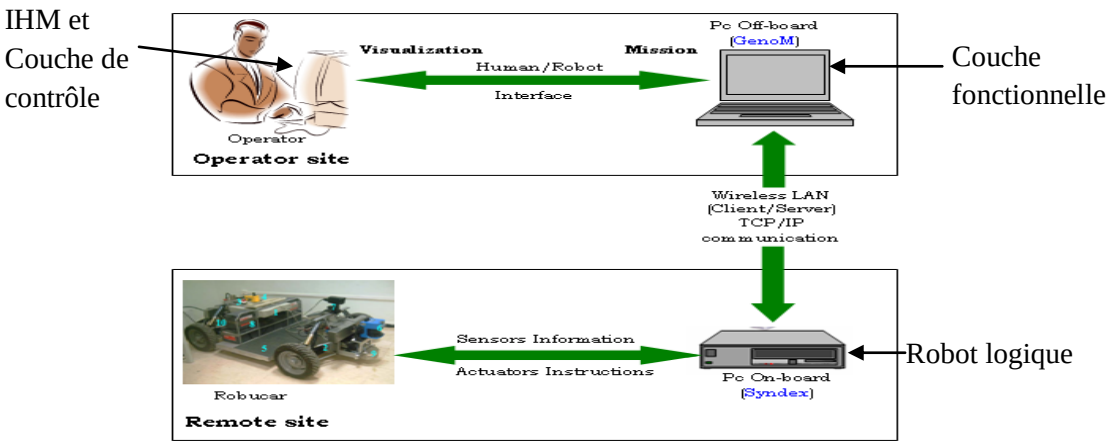


Figure III-17 : Architecture du système robotique

Les logiciels de développement sont, OpenCV pour le traitement d'image et GenoM pour l'implémentation sur le système embarqué et ceci tournant sous le système d'exploitation libre, Linux.

III.3.1 Qu'est-ce que GenoM :

GenoM est un outil, qui offre un environnement logiciel de développement permettant de définir, produire et contrôler des modules qui encapsulent des algorithmes. Cette structure modulaire permet de faire appel aux services de modules déjà existants via des protocoles d'échange de données standardisés par GenoM. [55]

a - Qu'est ce qu'un module:

Un module permet d'intégrer les fonctions de traitement sous la forme de services dans une structure standard, et d'accéder à ces services ainsi qu'aux données produites par le biais d'interfaces standards. Ces fonctions sont contrôlées au moyen de requêtes, c'est dans cette partie du module que nos algorithmes sont introduits. Lorsqu'un service, requis au moyen d'une requête, se termine, une réponse est retournée au client à la quelle est associé un bilan d'exécution qui caractérise la façon dont s'est déroulé le service.

On distingue deux types de requête : les requêtes d'exécution démarrent effectivement un service, alors que les requêtes de contrôle contrôlent leur exécution

(modification de paramètres, interruption). Ces données peuvent être transmises à la fin de l'exécution par la réplique, ou durant l'exécution par des posters.

Un poster est une structure C qui est mise à jour par une activité et qui est accessible en lecture par n'importe quel élément du système (un autre module, un opérateur,…). Toutes les données transmises par et vers le module (via les requêtes/répliques et les posters) sont mémorisées dans une base de données interne dénommée Structure de Données Interne Fonctionnelle ou SDIf.

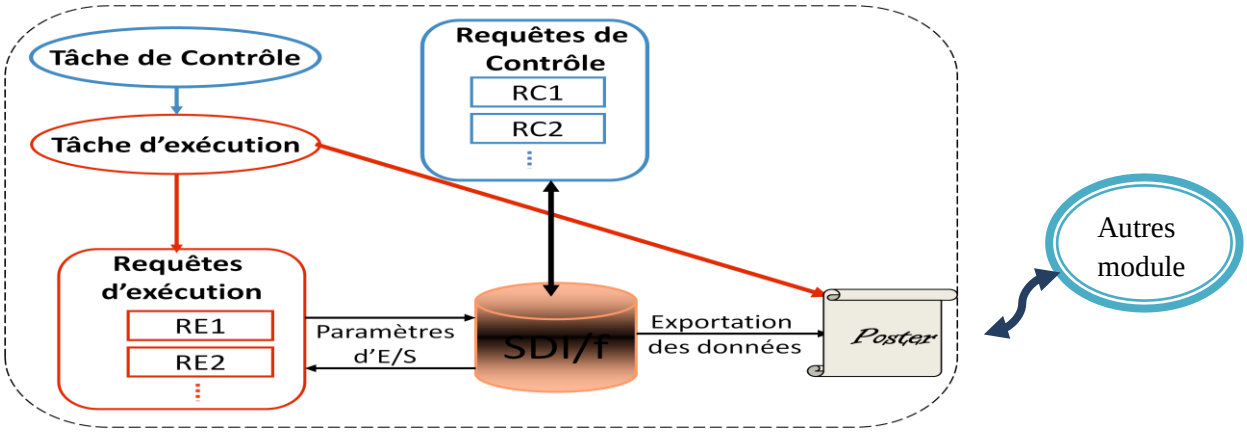


Figure III.18 : Module GenoM

b - Architecture modulaire existante :

Le projet dans lequel nous avons contribué, est constitué d'un nombre de modules qui sont remis à jour, à fin de les améliorer et d'ajouter de nouvelles taches au robot. Voici l'architecture qui existe actuellement.

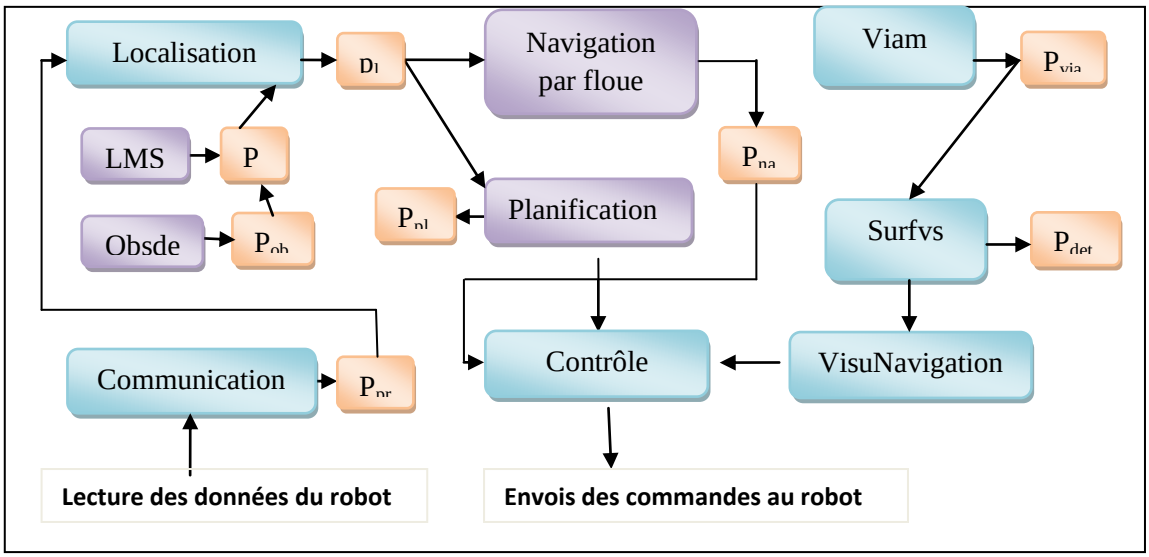


Figure III-19 : *Architecture modulaire du Robucar*

III.3.2 La librairie OpenCV :

Un des objectifs buts d'OpenCV est d'aider les utilisateurs à construire rapidement des applications sophistiquées de vision à l'aide de simples opérations de vision par ordinateur.

OpenCV (Open Source Computer Vision) est une bibliothèque proposant un ensemble de plus de 2500 algorithmes de vision par ordinateur, elle a était écrite en C et C++, puis des interfaces ont été développées pour Python, Ruby, Matlab et autre langages. Elle est distribuée sous une licence BSD (libre) pour les plates-formes Windows, GNU/Linux, Android et MacOS. Cette bibliothèque est orientée vers des applications en temps réel [56].

Initialement écrite en C il y a 10 ans par des chercheurs de la société Intel, OpenCV est aujourd'hui développée, maintenue, documentée et utilisée par une communauté de plus de 40 000 membres actifs. C'est la bibliothèque de référence pour la vision par ordinateur, aussi bien dans le monde de la recherche que celui de l'industrie.

III.4 Les détecteurs de cibles

III.4.1 Construction d'un détecteur de cibles

Les trois méthodes qu'on a choisi se basent sur l'apprentissage, voici leurs étapes :

a - Les caractéristiques

Le détecteur va tenter d'isoler certaines caractéristiques propres à l'objet qu'il doit détecter.

Pour un visage, les caractéristiques remarquables sont, par exemple, la partie supérieure de la tête qui contient une bande plus sombre correspondant à nos yeux ainsi que l'ombre de leurs orbites. De même, une différence de luminosité similaire est remarquable sur le nez et sur la bouche :

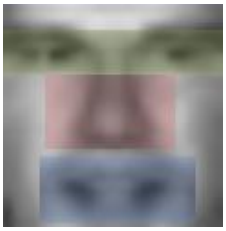


Figure III-20 : détecteur de visage

Pour les personnes, on se base essentiellement sur la tête, les épaules et les pieds. Les caractéristiques utilisées pour la détection d'un visage (ou d'une personne) devraient donc reprendre ces attributs.

b - L'apprentissage

C'est dans cette étape qu'il est prévu de trouver les caractéristiques les plus redondantes, et calculer pour chacune de celles-ci la valeur optimale d'un seuil à atteindre pour l'amplitude du dégradé de telle sorte qu'elle génère un minimum d'erreurs et de faux positifs. Cette opération est appelée l'entrainement d'un classifieur.

Pour cela, on devra lui fournir un nombre important d'échantillons de test (plusieurs milliers) qui seront labélisés comme étant ou non des images de visages (ou personnes).

C'est la tâche la plus complexe dans ce travail, tant au niveau de la conception que de la puissance de calcul nécessaire (un apprentissage peut prendre de plusieurs heures à plusieurs semaines, en fonction des l'algorithme ressources disponibles).

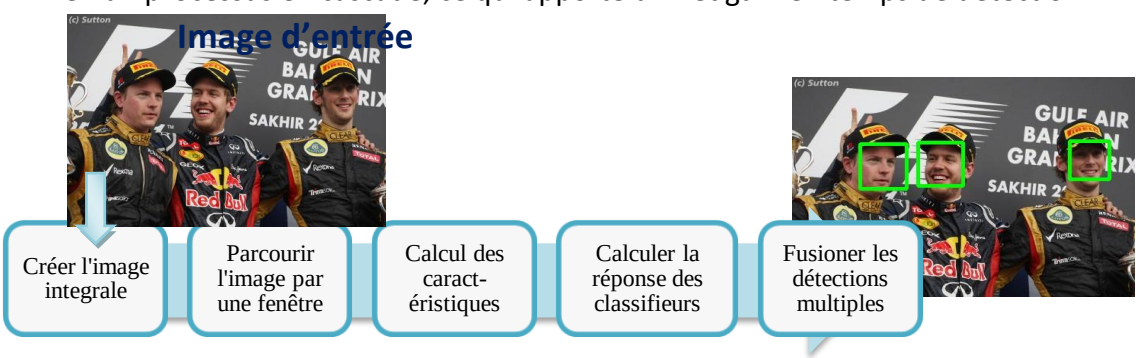
c - Le détecteur

Une fois que d'apprentissage aura généré un classifieur optimal, il suffira au détecteur de tester si celui-ci valide certaines parties de l'image.

La fonction retournera une liste de rectangles qui représenteront chacun une zone de la photographie qui semble représenter l'objet en question.

III.4.2 La méthode de Viola et Jones

Cette méthode propose une architecture pour combiner les classifieurs boostés en un processus en cascade, ce qui apporte un net gain en temps de détection.



FigureIII21 : *Organigramme du détecteur de visages de Viola & Jones*

a - Caractéristiques

Les caractéristiques proposées par Viola et Jones, considèrent des fenêtres de détection (ou masques) délimitant des zones rectangulaires adjacentes ; les intensités de pixels de ces rectangles sont additionnées, formant des sommes dont la différence constitue une caractéristique. La figure ci-dessous donne des exemples de masques à 2, 3 ou 4 rectangles, dans lesquelles la somme de pixels délimités par la zone sombre est soustraite de la somme des pixels délimités par la zone claire. Ces masques ou fenêtres de détection, généralement de très petite taille au départ (typiquement 20 × 20 pixels), sont appliqués à toutes les positions de l'image, puis leurs tailles agrandies. Une caractéristique est donc un nombre réel qui code les variations du contenu pixellique à une position donnée dans la fenêtre de détection. La présence de contours ou un changement de texture dont ainsi traduits numériquement par les valeurs des caractéristiques pseudo-Haar. Par exemple, un masque à 2 rectangles permet

d'indiquer où se situe la frontière entre une région sombre et une région claire.

Figure III-22 : Exemple des types de caractéristiques utilisées par Viola et Jones

b - L'image intégrale

A première vue, pour calculer chaque caractéristique, il semblerait nécessaire de parcourir l'ensemble des pixels de la zone, ce qui nécessite une puissance de calcul non négligeable. Pour remédier à ce problème, Viola et Jones ont introduit le concept d'image intégrale.

Une image intégrale est une matrice qui va contenir pour chaque pixel la somme de sa valeur et de l'ensemble des valeurs des pixels se trouvant au-dessus de lui et à sa gauche.

2	4	1	3
4	5	1	2
3	6	3	2
2	1	4	6

2	**6**	**7**	**10**
6	**15**	**17**	**22**
9	**24**	**29**	**36**
11	**27**	**36**	**49**

Figure III-23 : l'image intégrale

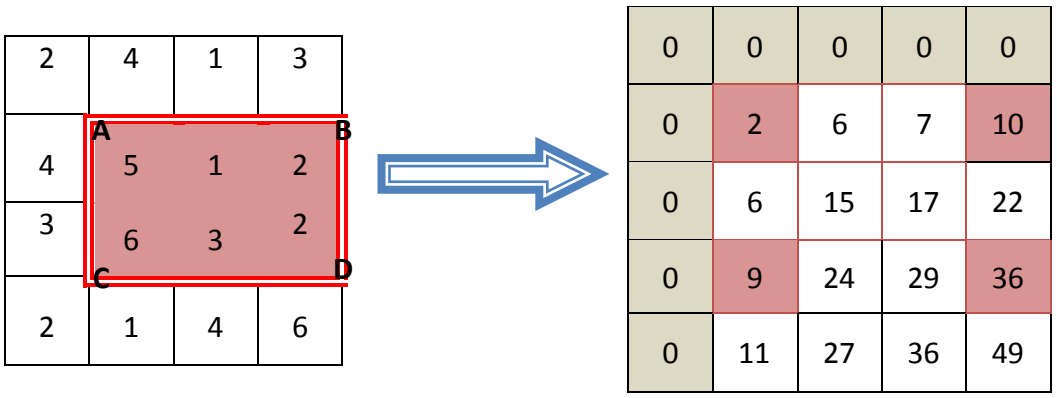
c - Calculs

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 [2] ou de 20 × 15 pixels [36]. 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.

En phase de détection, l'ensemble de l'image est parcouru en déplaçant la fenêtre de détection d'un certain pas dans le sens horizontal et vertical (ce pas valant 1 pixel dans l'algorithme original [34]). Les changements d'échelles se font en modifiant successivement la taille de la fenêtre de détection. Viola et Jones utilisent un facteur multiplicatif de 1.25, jusqu'à ce que la fenêtre couvre la totalité de l'image.

Finalement, et afin d'être plus robuste aux variations d'illumination, les fenêtres sont normalisées par la variance [34].

La conséquence de ces choix techniques, notamment le recours aux images intégrales, est un gain notable en efficacité, les caractéristiques étant évaluées très rapidement quelle que soit la taille de la fenêtre.



Somme = (36+2) – (10+9) = **19**

Figure III-24 : Exemple de calcul d'une caractéristique Haar

d - Sélection de caractéristiques par boosting

Le deuxième élément clé de la méthode de Viola et Jones est l'utilisation d'une méthode de boosting [34] afin de sélectionner les meilleures caractéristiques. 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. 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).

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, qui réduisent 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 élisant celle de ces dernières qui génère l'erreur la plus faible sur tout 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 ($x_1,\dots x_n$) et leurs étiquettes associées ($y_1,\dots,y_n$) , 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. 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 $y_i=h(x_i)\ \forall i \in \{1\dots n\}$. En pratique, le classifieur n'est pas parfait et l'erreur engendrée par ce classifieur est donnée par :

$$\mathcal{E}j=\sum^{n}_{i=1} w_i \left| h_i(x_i) - y_i \right| \qquad \text{III-7}$$

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 le classifieur présentant l'erreur la plus faible :

$$\mathcal{E}t = \min_j(\mathcal{E}j) \qquad \text{III-8}$$

Le classifieur fort final $h(x)$ est construit par seuillage de la somme pondérée des classifieurs faibles sélectionnés :

$$h(x) = \begin{cases} \sum_{t=1}^{T} \alpha_t h_t(x) \geq \sum_{t=1}^{T} \alpha_t \\ 0 \text{ sinon} \end{cases} \qquad \text{III-9}$$

Les $\boldsymbol{\alpha_t}$ sont des coefficients calculés à partir de l'erreur $\boldsymbol{\epsilon_t}$.

e - 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 (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 [33]. Cette structure en cascade peut également s'interpréter comme un arbre de décision dégénéré, puisque chaque nœud ne comporte qu'une seule branche [33].

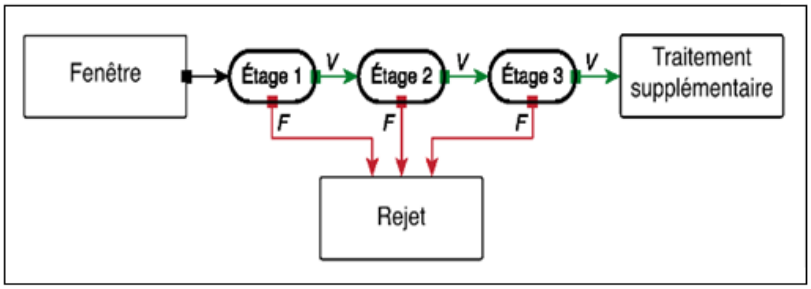


Figure III-25 : Cascade de classifieurs AdaBoost

En pratique, la cascade est constituée d'une succession d'étages, chacune étant formée d'un classifieur fort appris par AdaBoost. L'apprentissage du classifieur de l'étage est réalisé avec les exemples qui ont passé l'étage ***n*** et ***n-1*** ; ce classifieur doit donc faire face à un problème plus difficile : plus on monte dans les étages, plus les classifieurs sont complexes [34].

Le choix du nombre ***K*** d'étages est fixé par l'utilisateur ; L'utilisateur doit également spécifier le taux de détection minimal $\mathbf{d_i}$ et le taux de fausse alarme maximal $\mathbf{f_i}$ à atteindre pour l'étage **i**. Le taux de détection de la cascade est alors donné par : $$D=\prod_{i=1}^{k} d_i \quad \text{III-10}$$

Et le taux de fausse alarme par **:** $$F=\prod_{i=1}^{k} f_i \quad \text{III-11}$$

En pratique, les taux $\mathbf{d_i}$ et $\mathbf{f_i}$ sont les mêmes pour tous les étages. Indirectement, ces taux déterminent également le nombre de caractéristiques utilisées par les classifieurs forts à chaque étage : les itérations d'Adaboost continuent jusqu'à ce que le taux de fausse alarme cible soit atteint. Des caractéristiques/classifieurs faibles sont ajoutés jusqu'à ce que les taux cibles soient atteints, avant de passer ensuite à l'étage suivant.

Pour atteindre des taux de détection et de fausse alarme corrects en fin de cascade, il est nécessaire que les classifieurs forts de chaque étage aient un bon taux de détection; ils peuvent par contre avoir un taux de fausses alarmes élevé. Si l'on prend l'exemple d'une cascade de 32 étages, pour obtenir une performance finale ***D=0.9*** et $\boldsymbol{F=10^{-6}}$, chaque classifieur fort doit atteindre $\boldsymbol{d_i=0.997}$, mais peut se permettre $\boldsymbol{f_i=0.65}$ (i.e.$\boldsymbol{0.65^{32}=10^{-6}}$ et $\boldsymbol{0.997^{32}=0.9}$). Chaque étage ajouté diminue donc non seulement le nombre de fausses alarmes, mais aussi le taux de détection.

III.4.3 Les Motifs Binaires Locaux

Un algorithme écrit par des chercheurs à l'université de Pékin en 2008 pour détection d'objets dans les images. Utilisé souvent en systèmes embraquées pour sa rapidité de calcul et utilisé en temps réel.

Appart les caractéristiques propres à cet algorithme, on suit les mêmes étapes que celles de Viola et Jones. On utilise AdaBoost pour l'apprentissage, puis on a ajouté la cascade de classifieurs pour diminuer le temps de détection. La détection se fait également de la même façon en parcourant l'image avec une fenêtre glissante.

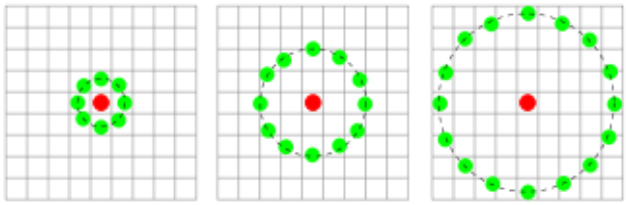
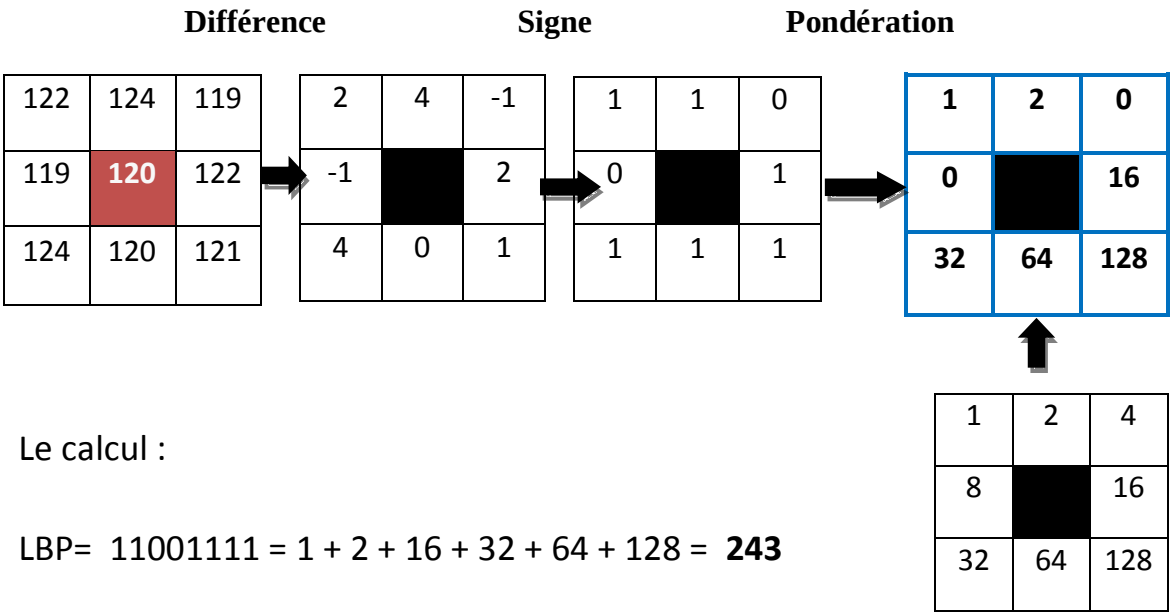


Figure III-26 : Exemples de voisinages utilisés pour définir une texture et calculer un LBP

f - Algorithme



Le calcul :

LBP= 11001111 = 1 + 2 + 16 + 32 + 64 + 128 = **243**

Figure III-27 : Algorithme du calcul de Motif Local Binaire

L'opérateur LBP basique prend comme entrée un carrée de 9 pixels et a pour sortie un nombre binaire (8 bits). Ce dernier est présenté en figure II-34. La motivation qui a poussé à utiliser cet opérateur est qu'un visage peut être vu comme un assemblage de micro-patterns[1] dont la description par LBP est à la fois bonne, robuste face aux variations de gris et rapide à générer

Cet opérateur simple a été étendu pour rester fiable à différentes échelles. Ainsi, P points décrivent le nombre binaire et ceux-ci sont distribués le long d'un cercle de rayon R. Cet entourage sera notée (P;R). Comme ces P points ne tombent pas nécessairement au centre d'un pixel de l'image, leurs valeurs sont obtenues par interpolation bilinéaire. Ceci est illustré à la figure III-33.

Ensuite ne sont gardés que les patterns 'uniformes' ne présentant au plus que deux transitions 1-0 ou 0-1 lorsqu'ils sont regardés comme des cercles. Ainsi 00110000 et 10000001 sont uniformes tandis que 10010001 et 11011101 ne le sont pas. En pratique,

[1] Feature : Mot anglais désigne : motif ou texture

la conversion d'un pattern non uniforme donne le résultat 0. On se rend compte ainsi que 90% des patterns (8; 1) appartiennent à la catégorie des uniformes. Cette extension permet d'interpréter les LBP en termes de coins et arrêtes inclinées.

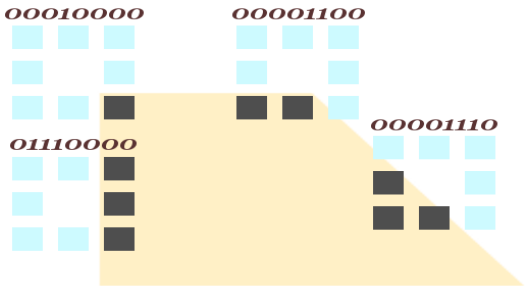


Figure III-28 : Exemple de détection de contours avec LBP

Ensuite pour inclure une information quant à la disposition spatiale des textures et éviter de se limiter à une description holistique des textures qui sourirait des limitations connues des méthodes de ce type, l'image convertie est divisée en plusieurs sous-régions pour lesquelles autant d'histogrammes LBP seront faits. Ces derniers seront concaténés pour former une matrice à 2 dimensions appelée 'histogramme spatialement amélioré'. Notons que les sous-régions peuvent se recouvrir et ne doivent pas nécessairement être rectangulaires. La comparaison de deux histogrammes spatialement améliorés suppose d'une part d'établir une méthode de mesure de distance entre deux histogrammes simples et d'autre part l'utilisation de poids pour rassembler les distances obtenues pour chaque sous-région. La méthode 2 en 1 proposée par Ahonen et al, est la distance carrée de Chi balancée :

$$X^2_w(x,\xi) = \sum_{j,i} \frac{w_j\,(x_{i,j} - \xi_{i,j})^2}{x_{i,j} - \xi_{i,j}} \qquad \text{III-12}$$

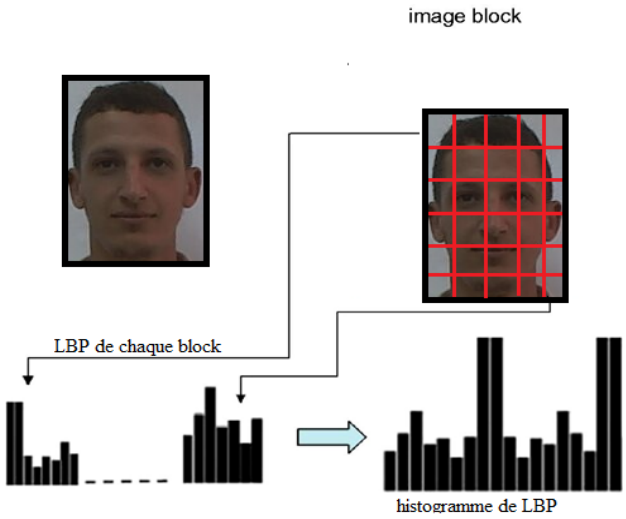


Figure III-29 : Formation de l'histogramme LBP

Ou x et ξ sont des histogrammes spatialement améliorés normalisés à comparer, i correspond à la $i^{ème}$ valeur du $j^{ème}$ sous-histogramme et w_j est le poids accordé à la sous-région j.

III.4.4 Les Histogrammes de Gradient Orienté

a - Algorithme

L'idée importante derrière le descripteur HOG est que l'apparence et la forme locale d'un objet dans une image peut être décrite par la distribution de l'intensité du gradient ou la direction des contours.

Ceci peut être fait en divisant l'image en des régions adjacentes de petites tailles, appelées cellules, et en calculant pour chaque cellule l'histogramme des directions du gradient ou des orientations des contours pour les pixels à l'intérieur de cette cellule.

La combinaison des histogrammes forme alors le descripteur HOG. Pour de meilleurs résultats, les histogrammes locaux sont normalisés en contraste, en calculant une mesure de l'intensité sur des zones plus larges que les cellules, appelées des blocs, et en utilisant cette valeur pour normaliser toutes les cellules du bloc. Cette normalisation permet une meilleure résistance aux changements d'illuminations et aux ombres.

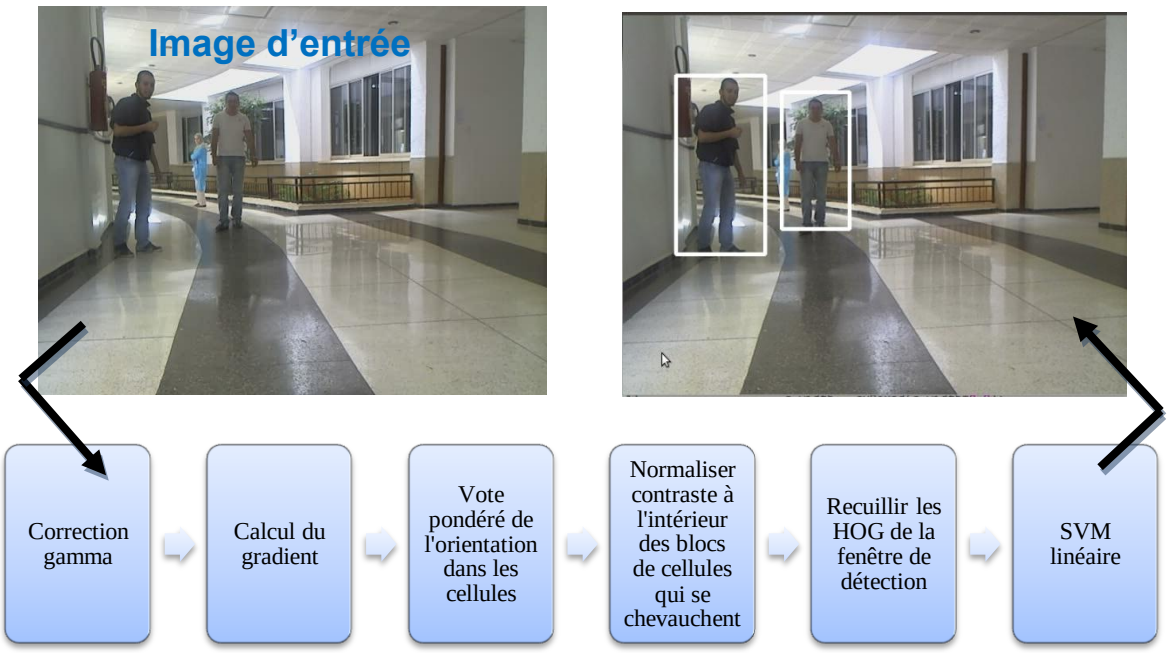


Figure III-30 : Organigramme de détection de personnes utilisant HOG

b - Correction gamma

Une étape de prétraitement peut être effectuée, afin que les couleurs de l'image soient normalisées, est une correction gamma correcte. Cette étape n'est pas très nécessaire, la normalisation du descripteur lui-même s'avérant suffisante, mais elle augmente quand même légèrement le taux de détection. Dalal et Triggs ont également testés plusieurs espaces de couleurs, le meilleur étant RGB.

c - Calcul du gradient

La première étape de la méthode est le calcul du gradient, la méthode la plus courante pour cela consistant à appliquer un filtre dérivatif 1-D centré, dans les directions horizontales et verticales. Les masques suivants sont utilisés pour cela: ***[-1, 0, 1] et [-1, 0, 1]*** ^{T}.

Dans le cas des images couleurs, le gradient est calculé séparément pour chaque composante, et on retient pour chaque pixel le gradient de plus grande norme.

D'autres types de masques plus complexes ont été testés, comme des filtres de Sobel 3x3, ou des masques diagonaux, ou non centrés [37]. On a testé avec filtre dérivatif 1-D et filtre de Sobel.

d - Construction de l'histogramme

La seconde étape est la création des histogrammes de l'orientation des gradients. Ceci est fait dans des cellules carrées de petite taille (de 4x4 à 12x12 pixels). Chaque pixel de la cellule vote alors pour une classe de l'histogramme, en fonction de l'orientation du gradient à ce point. Le vote du pixel est pondéré par l'intensité du gradient en ce point. Les histogrammes sont uniformes de 0 à 180° (cas non signé) ou de 0 à 360° (cas signé).

Dalal et Triggs font remarquer qu'une quantification fine de l'histogramme est nécessaire, et ils obtiennent leurs meilleurs résultats avec un histogramme à 9 classes. Prendre en compte le signe du gradient n'améliore pas les performances pour la détection de personnes, mais peut être significatif pour d'autres types d'objets [37].

e - Formation des blocs

Une étape importante est la normalisation des descripteurs, pour éviter les disparités dues aux variations d'illumination, ainsi que l'introduction de redondance dans le descripteur.

Pour cela, les auteurs regroupent plusieurs cellules dans un bloc, et effectuent la normalisation sur celui-ci. Les blocs se recouvrent, donc une même cellule participe plus d'une fois au descripteur final. Deux types de géométrie de blocs sont proposées: rectangulaire (R-HOG) ou circulaire (C-HOG).

f - Normalisation des blocs

Trois types de normalisation sont explorés. Le vecteur non normalisé contenant tous les histogrammes d'un seul bloc est désigné par *v* sa k-norme par $||v||_k$ et ϵ est une constante de faible valeur. Le facteur de normalisation est alors défini par:

$$\text{L2-norme: } f = \frac{v}{\sqrt{||v||_2^2 + \epsilon^2}} \qquad \text{III-13}$$

$$\text{L1-norme: } f = \frac{v}{(||v||_1 + \epsilon)} \qquad \text{III-14}$$

$$\text{L1-racine: } f = \sqrt{\frac{v}{(||v||_1 + \epsilon)}} \qquad \text{III-15}$$

Les normes L2-norme, et L1-racine obtiennent des performances similaires, tandis que L1-norme obtient de moins bons résultats, mais toutefois bien meilleurs que l'absence de normalisation [37].

g - Classification

L'étape finale dans le processus de détection d'objet est l'utilisation des descripteurs HOG pour entraîner un classifieur supervisé. Cette étape ne fait pas partie de la définition du descripteur HOG à proprement parler et différents types de classifieurs peuvent être utilisés. Dalal et Triggs choisissent volontairement un classifieur simple, un SVM à noyau linéaire, afin de mesurer essentiellement l'apport des HOG. Ils précisent en particulier qu'il serait intéressant de développer une méthode à base de cascade comme la méthode de Viola et Jones, en utilisant les HOG [37].

Ils précisent que l'utilisation d'un noyau gaussien améliore les performances de 3%, pour un taux de faux positifs par fenêtre de 10^{-4}, mais un coût de calcul bien plus élevé.

III.5 Implémentation

III.5.1 Apprentissages sous OpenCV :

On a utilisé des modèles de classification assez connu qui sont AdaBoost et SVM, intégrés en OpenCV avec une licence gratuite BSD.

a - Apprentissage Viola & Jones (AdaBoost)

Voici les étapes qu'on a suivies :

1. Création de deux dossiers négative et positive dans lesquels on place des images où, respectivement, il n'y a pas l'objet à détecter (fonds), et uniquement l'objet à détecter.
2. Création de deux fichiers textes neg.txt et pos.txt dans lesquels on met les liens des images.
3. Création d'un script qui permet d'obtenir et créer des images échantillons, dans lesquels les images positives sont superposées sur les fonds négatifs : c'est la base d'apprentissage. Cette base est retranscrite dans un fichier samples.dat, fichier indispensable au lancement de la fonction opencv_createsamples.
4. Utilisant la fonction opencv_createsamples, on obtient un fichier training.vec indispensable au lancement de l'apprentissage avec la fonction opencv_haartraining.
5. Opencv_haartranning calcule les classifieurs; ce calcul peut durer des semaines.
6. Utilisation de convert_cascade : permet d'obtenir un fichier xml.

On a fait des apprentissages pour la détection de visages et personnes avec cette méthode :

- Le détecteur de visages Viola et Jones :

On a créé 2 fichiers xml issus de 2 apprentissages : le premier pour la détection de visage en face (prise frontale), on a utilisé 1000 exemples positifs et 2000 négatifs, cet apprentissage a duré 10 jours, le deuxième apprentissage on l'a fait pour les visages

pris de profil, avec une base de 500 images positifs et 500 images de fonds (négatives) et a duré 6 jours.

- Le détecteur de personnes Viola et Jones :

C'est l'apprentissage qui nous a pris le plus de temps, on a essayé plusieurs apprentissages avec plusieurs bases, celle qui a donné les meilleurs résultats (relativement) est de 200 images positives et 1000 images de fonds, et a duré 35 jours pour arriver au 15ème stage, et on l'a arrêté à ce stade. D'autres apprentissages n'ont pas dépassé le 11ème stage après plus d'un mois.

b - Apprentissage LBP (Adaboost)

Utilise exactement les mêmes procédures que Viola et Jones, le seul changement est dans les étapes de l'apprentissage (l'étape 5 du procédé de Viola et Jones) où on utilise dans ce cas la fonction opencv_traincascade avec l'option features[2] : LBP.

On a fait les mêmes apprentissages de détection de visages, en créant donc 2 fichiers xml, le premier lbp_frontalface.xml en 15 heures, le deuxième lbp_profileface.xml en 12 heures.

c - Apprentissage HOG (SVM)

On a utilisé la base d'images de piétons d'INRIA qui comporte une base d'apprentissage composée de 1218 images positives (de personnes) et 614 négatives et une base de test de 288 images positives et 453 négatives.

1. Préparation de la base de données : on doit recadrer les images positives et négatives, pour cela on a choisis la taille 64*128 pixels.
2. Le vecteurs de HOG (HOG parametters): on a calculé les HOG utilisant la fonction OpenCV HOGDescriptor::compute avec les paramètres suivant (ces paramètres ont été choisis après plusieurs tests qu'on site en partie III-7):
    - Taille des blocs =(16,16); taille des cellules =(8,8).
    - Correction gamma (booléen)=1 (activée).
    - Type de normalisation de l'histogramme= L2-norme, V_{max} =0.2.
    - Nombre de bins =9.

[2] Mot anglais = caractéristiques ou traits

- Taille de fenêtre =(64,128).

3. L'apprentissage de SVM : on a créé un classifieur entrainé utilisant la fonction SVMLight.
4. On teste le classifieur avec les images de test négatives, et on remet les faux positifs pour entrainer le classifieur encore, ça s'appelle l'apprentissage fort (hard train).
5. Utilisant la fonction detectMultiScale, on change chaque fois le seuil global et ré entraine le classifieur avec les vecteurs des faux positifs jusqu'à avoir les meilleurs résultats possibles.

Détecteur		Nombre d'images positives	Nombre d'images négatives	Temps d'apprentissage
Détecteur de visage Viola et Jones	Frontal Profil	1000 500	2000 500	10 jours 6 jours
Détecteur de visage LBP	Frontal Profil	1000 500	2000 500	15 heures 12 heures
Détecteur de personnes Viola et Jones		200	1000	33 jours
Détecteur de personnes HOG		1218	614	3 jours

Tableau III-3 : Comparaison des temps d'apprentissage

III.5.2 Les détecteurs :

d - Viola et Jones [34]

La détection s'applique sur une image de test, dans laquelle on souhaite déceler la présence et la localisation de la cible (visage ou personne). En voici les étapes :

- Parcours de l'ensemble de l'image à toutes les positions et échelles, avec une fenêtre de taille 24 × 24 pixels, et application de la cascade à chaque sous-fenêtre, en commençant par le premier étage :
  - Calcul des caractéristiques pseudo-Haar utilisées par le classifieur de l'étage courant,
  - Calcul de la réponse du classifieur,
  - Passage ensuite à l'étage supérieur si la réponse est positive, à la sous-fenêtre suivante sinon,

- Enfin l'exemple est déclaré positif si tous les étages répondent positivement ;

- Fusion des détections multiples : l'objet peut en effet générer plusieurs détections, à différentes positions et échelles ; cette dernière étape fusionne les détections qui se chevauchent pour ne retourner qu'un seul résultat.

e - Détecteur LBP

Le détecteur de LBP est presque similaire à celui de Viola et Jones, sauf qu'il utilise une autre caractéristique, donc des vecteurs différents d'un classifieur Viola et Jones. Il comporte exactement les mêmes étapes à part le calcul des LBP à la place des haar. Donc un même programme a pu être utilisé pour les deux méthodes, (l'algorithme est expliqué en III.4.3).

f - Détecteur HOG

La même chose aussi, sauf qu'il utilise un classifieur conçu par SVM, qui contient les vecteurs HOG, il n'utilise pas la cascade classifieur, ce qui justifie sa lenteur (l'algorithme est expliqué en partie III.4.4).

III.5.3 Module de détection sous GenoM

Pour implémenter nos algorithmes sous GenoM, il nous fallait au début un module qui lit le flux d'images depuis une Webcam USB. Pour cela, on a utilisé le module VIAM-GenoM : The Versatile Image Acquisition Module, développé au LAAS CNRS, qui est un module de polyvalente acquisition d'images. On l'a adapté aux caméras USB (conçu pour cameras Fire-wire), et on a utilisé des filtres essentielles pour avoir une bonne qualité d'image en sortie.

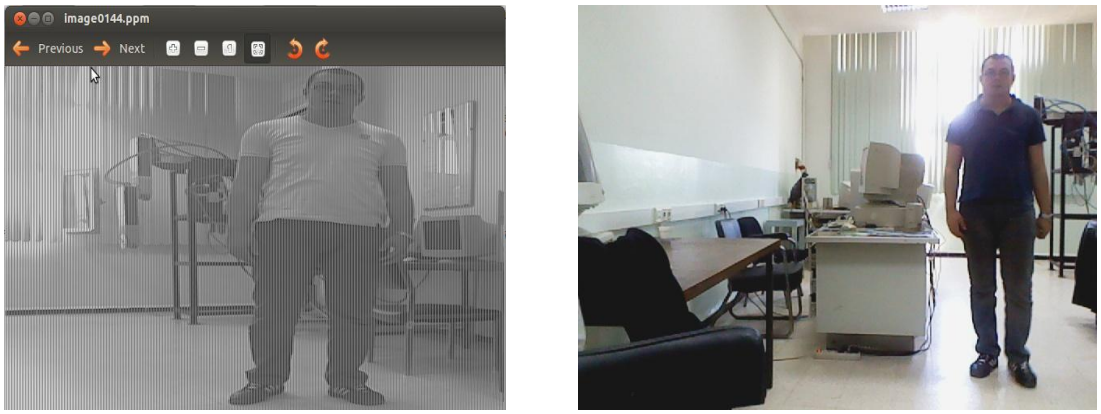


Figure III-31 : Exemple d'image issu de VIAM avant et après adaptation

Dans une première étape, nous avons implémenté nos algorithmes dans le module SURFVS. Cela nous a présenté quelques difficultés pour adapter les programmes OpenCV à GenoM. Pour cela, on a dû ajouter beaucoup de librairies. Ainsi un guide d'utilisation de notre module qui inclut un script TCL[3] a été ajouté en annexe.

Dans la deuxième étape poursuite de cibles, en utilisant la PTU. Le principe de cette tâche de poursuite est de contrôler les deux degrés de liberté de la caméra en rotation autour des axes $\vec{x}$ et $\vec{y}$, respectivement appelés panoramique (pan) et inclinaison (tilt), de façon à amener et à maintenir au centre de l'image la projection d'un objet d'intérêt (la cible).

Dans l'otique d'assurer une cadence de traitement rapide d'images (respect du temps réel) et de simplifier les traitements, l'objet d'intérêt considéré se résumera à un point particulier ; le centre de gravité de la cible : M(x, y) de coordonnées (x, y) en pixels dans le repère image.

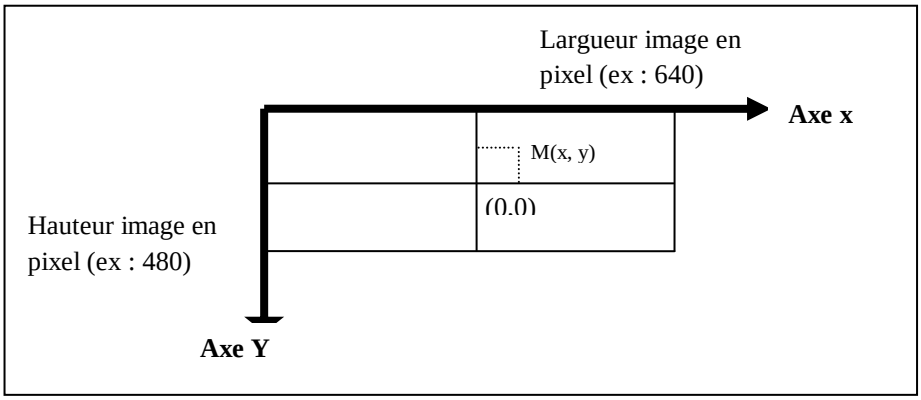


Figure III.32: coordonnées du point M(x, y) dans le repère image

L'asservissement visuel réalisé peut être résumé par son schéma synoptique :

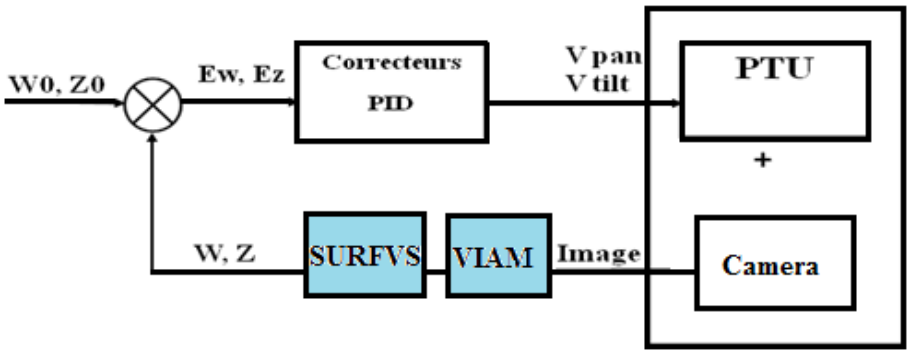


Figure III.33: Boucle d'asservissement de système de détection-suivi d'une cible avec PTU

[3] Script TCL : c'est un fichier texte contenant des lignes de commandes à exécuter.

Si à chaque itération (traitement d'une nouvelle image), il est possible d'extraire la position en pixels du point M dans l'image, alors il est possible de commander les mouvements de la tourelle de façon à amener et à maintenir le point M au centre de l'image.

III.6 Tests et résultats

III.6.1 Tests

On s'est basé entièrement sur ces tests pour paramétrer nos détecteurs, en essayant d'équilibré pour avoir des taux de détection assez élevé à temps d'exécution minimums.

a - Détection de piétons

On n'a testé que le détecteur de HOG en cette section, puisqu'on a choisis d'implémenter cette méthode seulement sur le robot. Le détecteur de piétons de Viola et Jones a été écarté à cause de sa très faible performance; ce qui va être démontré en partie (III.6.2-a).

On a cité plusieurs paramètres pour le détecteur de HOG qui sont : Correction gamma, gradient, blocs, cellules et seuil global. Dans chaque test, on a fixé tout les paramètres appart celui qu'on teste, en comptant chaque fois le nombre de détections, fausses détections (faux positifs) et le temps de détection.

Tout ces test on été fait sur une base qu'on a conçu, contenant 112 images de 320 x 240 pixels, prises au CDTA depuis le robot mobile (robucar), en deux milieux différents : un large couloir (milieu intérieur) et un parking (milieu extérieur).

Ces 112 images contiennent 216 cibles à détecter (personnes) et on a pris en compte une détection étant bonne si le cadre de détection couvre plus de la moitié de la cible.

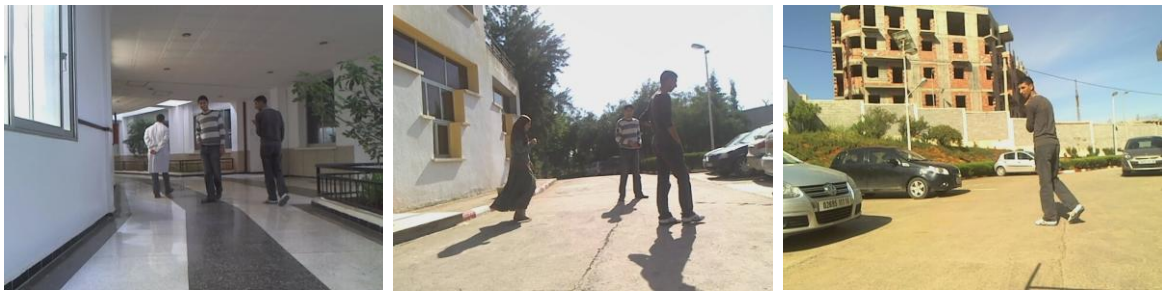


Figure III-34 : Quelques exemples de la base de tests

b - Correction gamma :

Correction gamma	Détection	Faux positifs	Temps de détection
Avec Correction	**192 (88%)**	**17 (8%)**	**150**
Sans correction	**105 (49%)**	**11 (5%)**	**125**

Tableau III- 4 : Performances du HOG avec et sans correction gamma

Ces résultats confirment l'importance de la correction gamma, surtout en milieu intérieur (faible luminance).voici un exemple :

Figure III-35 : Résultat de détection sans et avec correction gamma

c - Gradient :

On a testé deux filtres, et voici les résultats :

le filtre gradient	Détection	Faux positifs	Temps de détection
Sobel	**180 (83%)**	**16 (7.5%)**	**176**
Dérivateur 1-D	**192 (88%)**	**17 (8%)**	**150**

Tableau III- 5 : Tests de HOG avec différents gradients

On remarque que le filtre dérivateur D-1 donne des résultats légèrement meilleurs.

d - Le seuil global :

Le seuil global	Détection	Faux positifs	Temps de détection
1	**0 (0%)**	**0 (0%)**	**50**
1.01	**201 (93%)**	**38 (18%)**	**532**
1.02	**197 (91%)**	**36 (17%)**	**378**
1.03	**198(91%)**	**29 (13%)**	**280**
1.04	**197(91%)**	**23 (10%)**	**182**
1.05	**192 (88%)**	**17 (8%)**	**150**
1.06	**177(82%)**	**17 (8%)**	**127**
1.07	**150(69%)**	**12 (6%)**	**110**
1.08	**122(56%)**	**14 (7%)**	**100**
1.09	**122(56%)**	**13 (6%)**	**91**
1.1	**120(55%)**	**10 (5%)**	**86**
1.15	**104 (48%)**	**6 (3%)**	**62**
1.2	**86 (40%)**	**5 (3%)**	**50**

Tableau III- 6 : Tests de HOG avec différents seuils

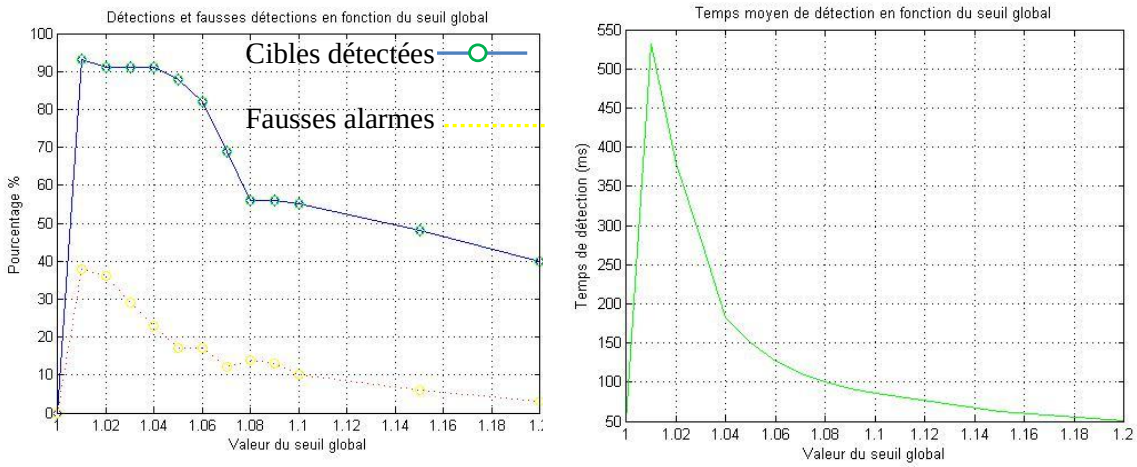


Figure III-36 : Performance et taux de détection de HOG par rapport au seuil global

On remarque du premier graphe, que les meilleurs taux sont entre le seuil= 1,01 et 1,06 où le taux de fausses détection est moyen. Dans le deuxième graphe on voit que les temps les moins élevés sont enregistré quand le seuil est grand.

En combinant les résultats des 2 graphes, on a choisis la valeur 1,05 qui nous donne un taux de détection assez élevé, un taux de faux positifs moyen et un temps de détection proche du temps réel.

e - Taille des cellules

Taille de cellules	Détection	Faux positifs	Temps de détection
6 x 6	**195 (90%)**	**20 (9%)**	**700**
7 x 7	**196 (90%)**	**17 (8%)**	**600**
8 x 8	**192 (88%)**	**17 (8%)**	**150**
9 x 9	**190 (88%)**	**16 (7%)**	**1000**
10 x 10	**160 (74%)**	**13 (6%)**	**547**

Tableau III- 7 : Tests de HOG avec différentes tailles de cellules

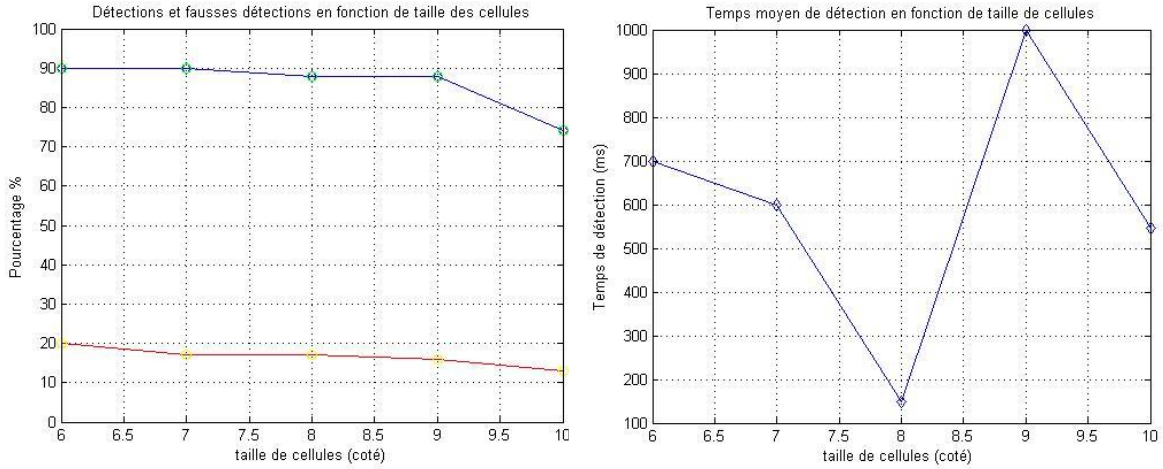


Figure III- 37 : Tests de HOG avec différentes tailles de cellules

On remarque ici que la taille des cellules choisis influe sur la qualité de détection, et encore plus sur le temps de détection. On a choisis la taille 8x8 pour avoir un temps de calcul raisonnable, les tailles inférieurs bien que légèrement meilleures, leurs temps de calcul ne permet pas une application temps réel.

f - Taille des blocks :

Taille des blocks	Détection	Faux positifs	Temps de détection
24 x 24	**180 (83%)**	**16 (7%)**	**135**
28 x 28	**189 (87%)**	**17 (8%)**	**143**
30 x 30	**190 (88%)**	**16 (7%)**	**145**
32 x 32	**192 (88%)**	**17 (8%)**	**150**
36 x 36	**195 (90%)**	**18 (8%)**	**160**

Tableau III- 8 : Tests de HOG avec différentes tailles de blocks

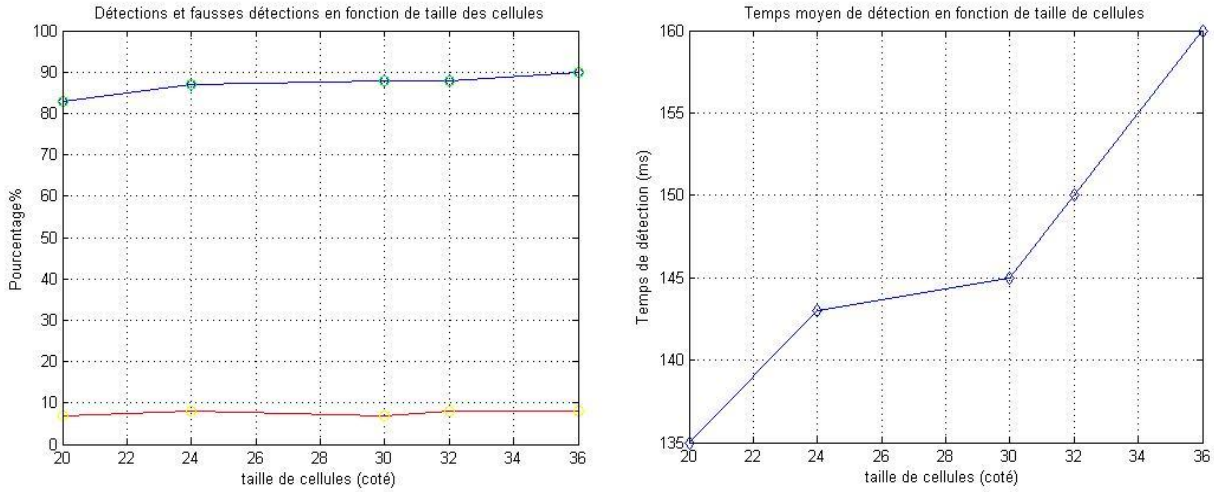


Figure III-38 : Tests de HOG avec différentes tailles de blocks

Contrairement aux tailles de cellules, les plus grandes tailles de blocs donnent les meilleurs taux de détection, le temps lui, n'est pas très affecté par ce changement.

En essayant les différentes combinaisons possibles entre la taille des blocs et celles des cellules, on a trouvé que celle qui donne les meilleurs résultats possible est : bloc 36x36, cellule 9x9 avec contrainte de temps d'exécution : 3 secondes en moyenne.

On a choisi la combinaison : 32 x 32, 8 x 8 ; ça nous a donné d'assez bons résultats avec temps d'exécution proche du temps réel (140ms => 7 frames/secondes).

III.6.2 Résultats

a - Détection de piétons

Après avoir choisis tout les paramètres, qu'on estime optimaux par rapport à la contrainte temps réel, on a testé notre programme sur plusieurs bases de données, parmi elles, une téléchargé depuis internet (la base d'INRIA), et d'autres construites lors des sorties avec le robot. Voici présentées, quelques exemples d'images de ces bases ainsi que les résultats.

Base INRIA :

La base de tests d'INRIA est composée d'une base positive dont les images contiennent des personnes et une base négative où on trouve des images d'objets ayant une ressemblance avec la forme d'humains (arbres, poteaux, plaques de signalisation…). Pour tester la robustesse de nos détecteurs et choisir un d'entre eux pour l'implémenter sur le robot, celle-ci nous donnes les résultats suivant :

Le détecteur		Nombre d'images	Nombre de personnes	Nombre de détections	Taux de détection
HOG	**Base positive**	**900**	**1178**	**1012**	**85%**
	Base négative	**452**	**0**	**28**	**6%**
Viola & Jones	**Base positive**	**1600**	**1178**	**153**	**12%**
	Base négative	**452**	**0**	**5**	**1%**

Tableau III- 9 : Comparaison des détecteurs de piétons

On remarque que les HOG donnent des résultats largement supérieurs à ceux de Viola & Jones ; le taux de Viola et Jones est bas malgré le temps qu'a pris l'apprentissage, même si on a utilisé la même base pour l'apprentissage de HOG et qui a donné de bons résultats.

Remarque : la base d'apprentissage d'INRIA et la base de tests sont différentes.

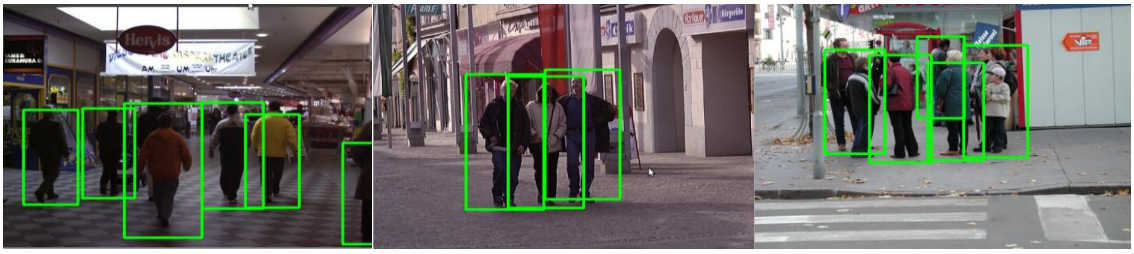


Figure III-39 : Application du détecteur de HOG sur les exemples de base INRIA

Exemples de détection sur les bases de données conçues :

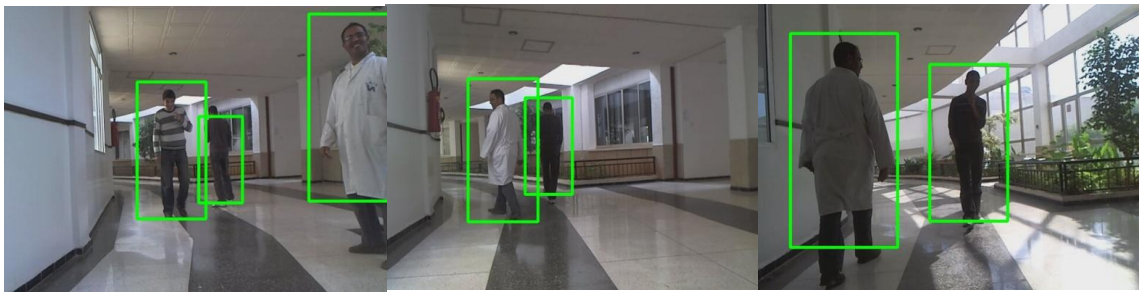


Figure III-40 : Application du détecteur de HOG sur les exemples de base Vid_int 1

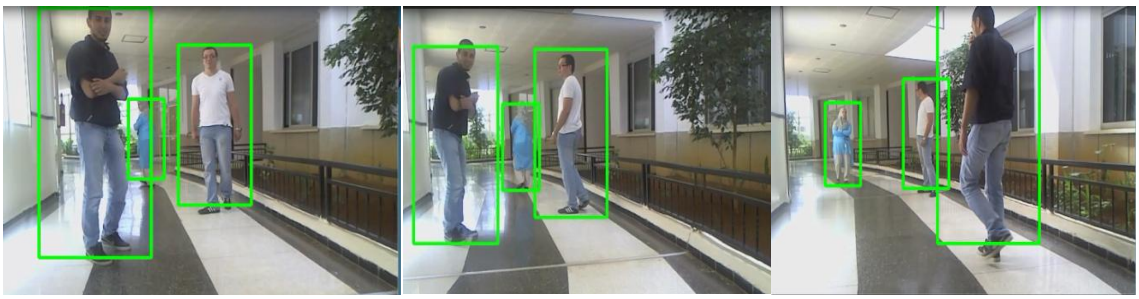


Figure III-41 : Application du détecteur de HOG sur les exemples de base Vid_int 2

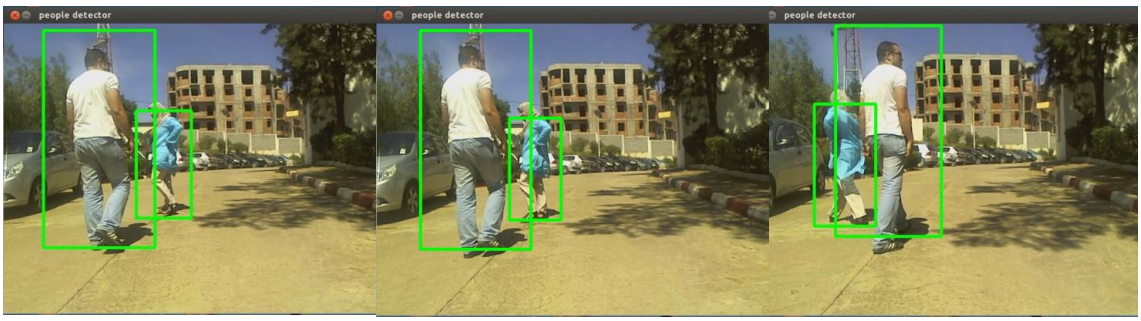


Figure III-42: Application du détecteur de HOG sur les exemples de base Vid_ext 1

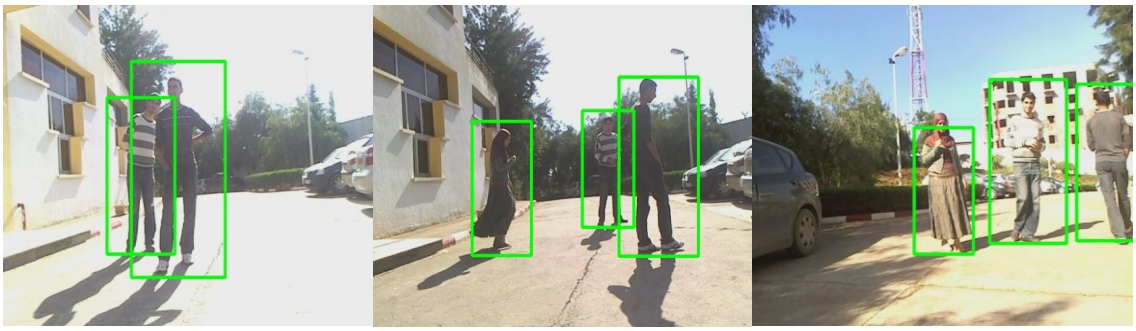


Figure III-43 : Application du détecteur de HOG sur les exemples de base Vid_ext 2

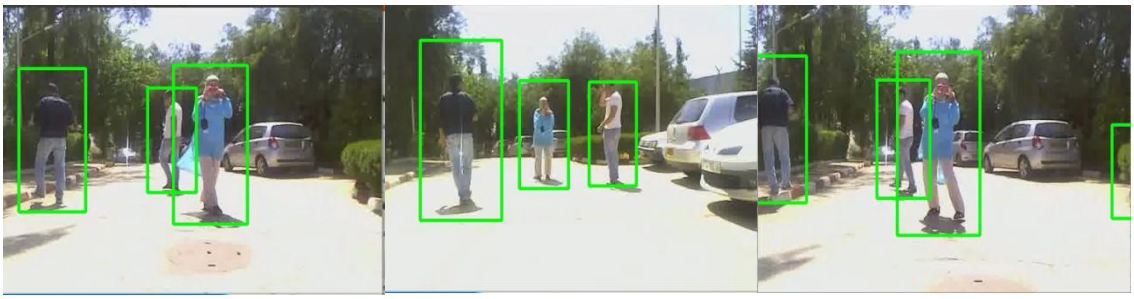


Figure III-44 : Application du détecteur de HOG sur les exemples de base Vid_ext 3

Figure III-45 : Application du détecteur de HOG sur les exemples de base Vid_ext 4

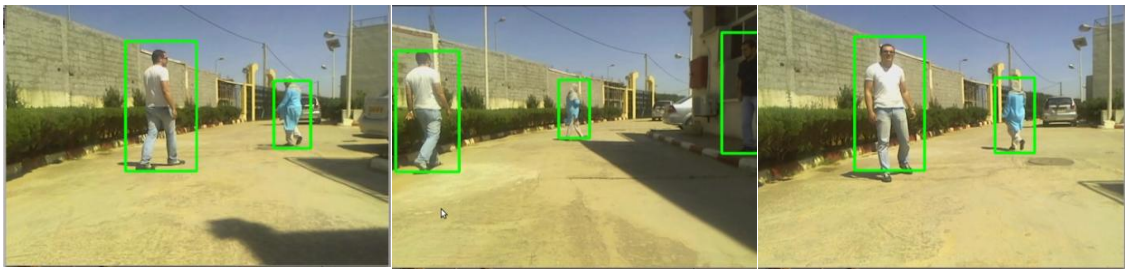


Figure III-46 : Application du détecteur de HOG sur les exemples de base Vid_ext 5

La base	Nombre d'images	Nombre de cibles	Cibles détectés	Nombre de fausses détections	Taux de détection
INRIA	900	1178	1012	170 (17%)	86%
Vid_int 1	70	188	148	51 (27%)	78%
Vid_int 2	439	964	827	108 (11%)	85,7%
Vid_ext 1	599	1029	998	115 (11%)	96 %
Vid_ext 2	80	181	147	35 (19%)	81%
Vid_ext 3	256	856	716	300 (35%)	84%
Vid_ext 4	100	300	168	3 (1%)	56%
Vid_ext 5	245	513	487	40 (8%)	94.93%
TOTAL	**2589**	**5209**	**4503**	**822 (16%)**	**86%**

Tableau III-10 : Résultats du détecteur de Piétons « HOG »

Remarque : Toutes nos bases sont construites à partir du robot mobile (vehicule), et les cibles à détecter étaient en mouvement aussi.

On résume dans ce tableau nos statistiques sur les taux de détection et de fausses alarmes sur les bases citées. le taux de détection est calculé en divisant le nombre de cibles encadrés par le détecteur, sur le nombre de cibles estimé par un humain.

Le taux de fausses détection est calculé en divisant le nobmre de fausses détection (des vignettes qui n'encadrent pas une cible) sur le nombre de cibles de cette base.

b - Détection de visages

- **Détecteur de viola et Jones**

On teste ici les taux de détection des apprentissages en fonction de l'orientation du visage. La courbe verte correspond au détecteur des visages de face tandis que la bleue montre les performances du détecteur de visages de profil. Les courbes ont été construites à partir de tests portant sur 5 sujets présents chacun selon 9 orientations différentes s'étalant uniformément de 0° à 90°.

Figure III-47 : Exemples de détection de visages : vert Viola & Jones, orange LBP

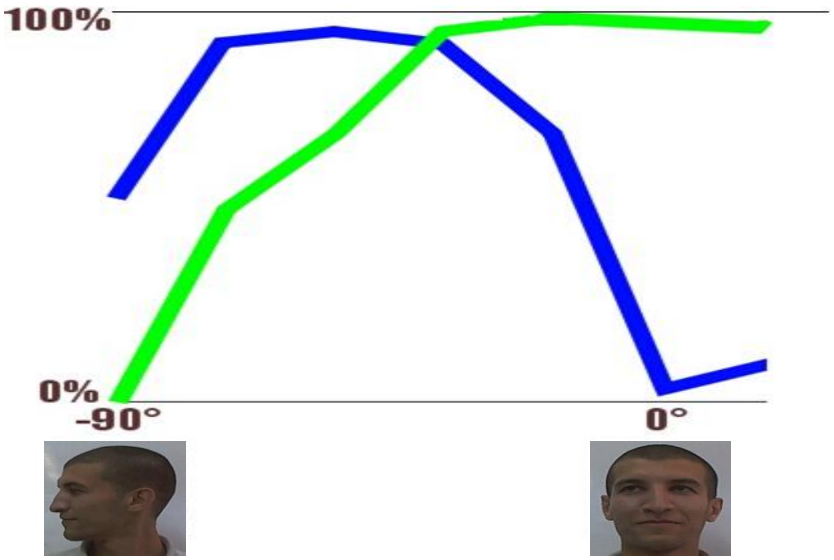


Figure III-48 : Angle de détection pour les détecteurs de visage de Viola et Jones

On remarque que le détecteur de visages de face, a un taux de détection élevée entre -30° et 30°. Tandis que pour celui du visage de profil, la détection est meilleure entre 30° et 70°.

- **Comparaison entre détecteurs de Viola et Jones et LBP :**

	Le détecteur	Nombre d'images	Nombre de personnes	Nombre de détections	Taux de détection	Fausses détections	Temps de détection (ms)
BASE 1	**LBP**	**100**	**240**	**181**	**75%**	**63 (26%)**	**35**
	Viola& Jones	**100**	**240**	**210**	**87.5%**	**35 (15%)**	**50**
BASE 2	**LBP**	**220**	**323**	**232**	**72%**	**70 (%)**	**32**
	Viola&Jones	**220**	**323**	**284**	**88%**	**54 (%)**	**48**

Tableau III-11 : comparaison des détecteurs de visages

On remarque que le détecteur LBP est plus rapide que celui de Viola et Jones, ce qui lui permet d'être implémenté en temps réel et en mouvement (35ms en moyenne <=> 30 images par secondes).

Le détecteur de Viola et Jones est plus robuste, avec une supériorité en plan d'efficacité et de fausses détection.

III.7 Conclusion

Le passage de l'algorithme en théorie à l'application réelle était la phase la plus importante. L'apprentissage nous a pris beaucoup de temps, le détecteur de personnes de Viola et Jones a pris plus d'un mois en un cluster, et malgré ça, il n'a pas donné de bons résultats. De plus, nous avons réalisé un deuxième passage : de l'application sur ordinateur, à l'implémentation sur le robot, ou on a utilisé trois modules séparés pour fermer la boucle : acquisition, décision et action.

On s'est contenté du détecteur de HOG, pour la détection de piétons, après une longue série de tests, nous avons réussi a trouvé la combinaison de paramètres qui nous convient. L'application fonctionne sur le robot en temps proche du temps réel et avec une performance assez satisfaisante (une détection supérieure à 86% en moyenne sur les bases d'essais utilisées, sur un total de 5209 vignettes ou personnes). Ce détecteur est assez performant, malgré le problème du nombre élevé de fausses détections.

Pour le visage, on a décidé d'implémenter les deux détecteurs, LBP pour sa rapidité de détection et Viola et Jones pour sa grande robustesse.

Enfin, l'intégration de notre module dans une application robotique.

Conclusion

Le travail présenté dans ce mémoire a été effectué au sein de l'équipe NCRM de la division Productique et Robotique au « CDTA ». Ce travail rentre dans le cadre d'un projet de véhicule autonome pour le transport urbain, et consiste à détecter les piétons et les suivre en utilisant une caméra embarquée.

Pour réaliser un tel travail nous avons été confrontés à des problèmes liés à l'aspect expérimental. L'environnement étant très complexe, les premiers problèmes concernaient les occultations, le changement de luminosité, la grande variation de texture et surtout la grande différence entre les personnes (âge, couleur de peau, taille, vêtements, position par rapport à la caméra, en mouvement ou statique, etc.).

Nous avons présenté en premier lieu les robots mobiles autonomes en donnant une définition, l'utilité de ce type de systèmes, les domaines d'applications et ses principaux éléments. Ensuite nous avons abordé la vision assistée par ordinateur, sa définition, domaine d'utilisation puis on a détaillé quelques outils de traitement d'image qu'on a utilisé dans nos programmes.

En deuxième lieu, nous avons réalisé une étude sur les méthodes de détection de cibles, on se focalisant surtout sur celles qui détectent les humains. Vu la difficulté de cette tâches, peu de techniques récentes ont pu donner d'assez bons résultats. Après cela, nous les avons évalué et choisis quelques unes pour l'implémentation.

En troisième étape, on a conçu des bases d'apprentissage pour chaque méthode, une étape qui nous a pris du temps, puis développé les programmes pour les évaluer. La conception des bases de données était une tache assez importante, il fallait prendre en considération les différences entre les personnes, puis faire un apprentissage fiable avec un minimum de fausses alarmes. Ensuite, nous avons tenté d'améliorer notre application en testant ses performances pour différentes valeurs de paramètres, avant d'implémenter les meilleures sur le robot.

Une étape importante est l'évaluation des performances, ou on a testé cette application sur des bases de tests téléchargé depuis internet et d'autres conçus en temps réel depuis notre caméra embarquée sur le robot. Les résultats étaient assez satisfaisant, le temps de détection aussi.

Au cours de ce projet et à travers le travail fait, nous avons pu acquérir beaucoup de connaissances. Premièrement, nous avons utilisé les outils appris au cours de notre formation, et enrichi nos connaissances dans le domaine du traitement d'image en apprenant de nouveau outils(OpenCV). Deuxièmement, nous avons pu acquérir des notions de base sur les robots mobiles, en manipulant un robot de type véhicule autonome. On a aussi appris à utiliser le système d'exploitation LINUX, avec un logiciel conçu pour les systèmes embarqués appelé « GenoM » ; celui-ci utilise une architecture modulaire (parallèle) ainsi que le langage de programmation C/C++.

Cette expérience a été très riche pour nous, la collaboration avec d'autres stagiaires et des membres de l'équipe NCRM nous a appris à travailler en groupe, surtout avec l'architecture modulaire où chaque projet dépend des autres.

On est satisfait de notre contribution pour le projet, surtout qu'on a un bon taux de détection. Nous estimons que notre module pourra être exploité pour différentes tâches, parmi elles le suivi de cibles, l'évitement de piétons, etc. Il pourra aussi être utilisé en dehors de la robotique, comme pour la surveillance ou encore l'observation et la gestion du trafic.

Pour améliorer les performances de notre module, nous citons quelques recommandations :

- L'ajout d'une deuxième caméra (stéréo vision) pour le calcul de la profondeur (une $3^{ème}$ dimension).
- Diminuer le temps de détection, en combinant la robustesse du détecteur de HOG et la rapidité du classifieur de Viola & Jones : AdaBoost, la cascade de classsifieur peut réduire d'avantage le temps d'exécution du module.
- On pourrait aussi améliorer le taux de performance des HOG si on y intègre le concept de l'image intégrale.

Annexes

1. Description du Robucar :

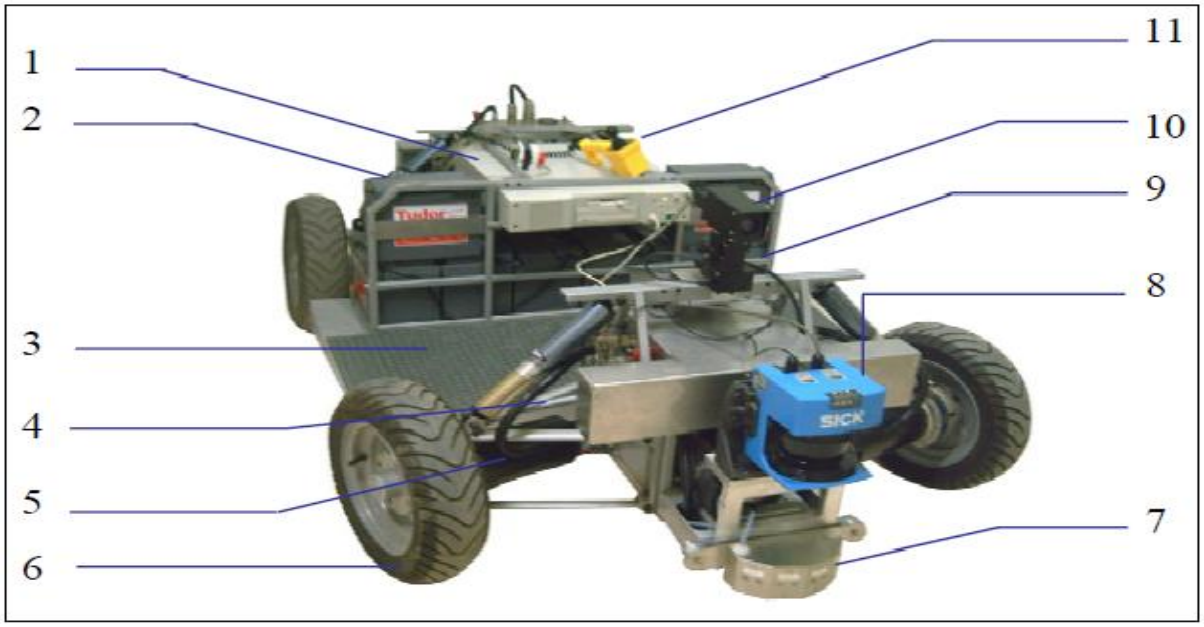


Figure A-1 robot mobile Robucar

1 : PC embarqué

2 : 8 batteries

3 : Châssis nid d'abeille

4 : Vérin électrique de direction

5 : Un moteur de traction par roue

6 : Roue motrice et directrice (avec un frein à commande électrique)

7 : Ceinture de capteurs à ultrasons (deux ceinture de 4 capteurs, avant et arrière)

8 : Capteur laser

9 : Ecran tactile

10 : Caméra CCD (N/B)

11 : joystick pour la conduite manuelle sécurisée

2. Caractéristiques technique du Robucar :

Le robot mobile Robucar se déplace grâce à ses quatre roues motrices et directrices réparties en deux trains ; avant et arrière. Ces roues sont reliées à quatre moteurs qui assurent la traction du robot et deux vérins qui assurent la rotation des roues avant et arrière. Les caractéristiques générales du robot mobile Robucar sont les suivantes :

- longueur totale 1836mm.
- largeur totale 1306mm.
- hauteur : 616 mm.
- poids totale : 310 kg.
- Motorisation : 4 moteurs électriques de 1200 Watts.
- 4 roues motrices et directrices.
- Vitesse maximale : 18 km/h (5 m/s).
- autonomie : 2 heures d'utilisation continue.
- capacité d'accueil : 2 personnes avec baguages.
- conduite automatique ou manuelle.

Le Robucar est constitué d'un ordinateur embarqué et de deux cartes RSMPC555 équipées d'un microcontrôleur MOTOROLA MPC555. Ces cartes contrôlent respectivement les moteurs de traction avant et arrière ainsi que ceux de direction. L'ensemble PC embarqué et cartes MPC555 communique via un bus de terrain contrôlé area network (CAN).

Le robot est muni d'un capteur laser, d'une caméra CCD N/B placés sur la face avant du robot, de huit capteurs à ultrasons ; quatre placés à l'avant et quatre à l'arrière. Il comporte aussi des encodeurs (liés aux quatre moteurs de traction et aux deux vérins) qui servent à mesurer le déplacement et l'orientation effectués par le robot. Les données provenant de ces capteurs sont présentées de façon à donner la vitesse linéaire et l'angle de braquage du robot à partir desquelles peut être calculée la position du robot.

3. Environnement logiciels :

L'interface homme/machine à été implémentée sur la machine distante, comportant le système d'exploitation libre Linux Ubuntu, alors que le robot logique est implémenté sur la machine embarquée avec le système d'exploitation Linux Red-Hat.

3.1. Préparation de l'environnement GenoM

La préparation de l'environnement GenoM est divisée en deux phases : la phase de configuration et la phase d'installation.

Mais avant d'entamer ces deux opérations, il est indispensable de rassembler un ensemble d'outils nécessaires à sa construction:

Les outils externes : GenoM a besoin de certains outils qui sont généralement disponibles sur la majorité des systèmes d'exploitation récents :

- *autoconf :* version 2.59 ou plus
- *automake :* version 1.8 ou plus
- *GNU make :*verssion 3.79 ou plus
- *pkgconfig :*verssion 0.15 ou plus
- *groff :* verssion 1.10 ou plus
- *Tcl/tk : Tool Command Language* verssion 8.0 ou plus

Les outils et librairies OpenRobots : GenoM est un outil d'open-source, pour pouvoir l'installer et l'utiliser il faut télécharger et installer les différant librairies et outils suivant :

- pocolibs : système de communication et primitive temps réel (version utilisée..)
- mkdep : outils du LAAS pour déterminer les dépendances (version utilisée..)
- libedit : editline, optionnelle (version utilisée..)
- eltclsh : interactive TCL shell avec editline , optionnelle mais très pratique (version utilisée ..)
- GenoM : générateur de modules (version utilisée..), ceci peut se faire en installant robotpkg, ainsi on peut installer tout les packages manquants car celui-ci nous informe au cours de son installation, de tout ce dont GenoM a besoin pour bien fonctionner.
- Viam : module d'acquisition polyvalent (The Versatile Image Acquisition Module).
- Vidéo4Linux : un pilote spécifique pour les caméras.

a-La phase de configuration :

La configuration de l'environnement consiste à :

- ✓ créer le répertoire qui va contenir l'installation des outils et librairies nécessaires pour la construction de l'environnement lors de la phase d'installation (openrobots).
- ✓ Créer et introduire les différentes variables d'environnement (rajoutées au root path), qui vont décrire les chemins menant vers le répertoire Openrobots et ses sous répertoires : le **bin** (répertoire des fichiers binaires qui seront considérés

comme des commandes ou des exécutables), le **lib** (répertoire des librairies), le **include** (répertoire contenant les includes) créés automatiquement lors de l'installation des outils.

L'introduction de ces variables d'environnement va permettre au système de reconnaitre toutes nouvelle commande ou librairie installer dans l'Openrobots à partir de n'importe quel répertoire sans être obligé de rentrer dedans. On a créé les variables suivantes :

- ✓`export LD_LIBRARY_PATH=/usr/local/lib`
- ✓`OPENROBOTS/lib/openprs:/usr/lib`
- ✓`export LD_LIBRARY_PATH=$OPENROBOTS/lib`
- ✓`export PKG_CONFIG_PATH=$OPENROBOTS/lib/pkgconfig:/usr/include`
- ✓`export PATH=$PATH:.:$OPENROBOTS/bin:$OPENROBOTS/sbin`
- ✓`export PATH=$PATH:.:$OPENROBOTS/bin:$OPENROBOTS/sbin`

b-La phase d'installation :

Dans cette phase on va installer les différents outils et librairies déjà cités, ces derniers doivent être d'abord téléchargés ensuite décompressés (si sont sous format « tar.gz »).

3.2. Manuel d'utilisation du module :

Le module sur le quel nous avons travaillé est 'surfvs', celui-ci était distingué à l'asservissement visuel (suivie de points dans une image ou flux vidéo). Les étapes à suivre pour faire fonctionner le module sont, la configuration et l'installation (make et make install).

Pour le lancement du module, on a besoin d'ouvrir trois fenêtres du terminal et exécuter les commandes suivantes :

1ère fenêtre:

- h2 init (puis 'yes')
- Viam –b (celui-ci doit être en matche pour que le surfvs puisse être exécuté)

2ème fenètre :

- Surfvs –b

3ème fenètre :

- Tclserv

- Eltclsh –package genom
- Connect
- Source vs2.tcl
- Viam (viam est lancé)
- face (détection de visage activée)
- hog (détection de personne activée)

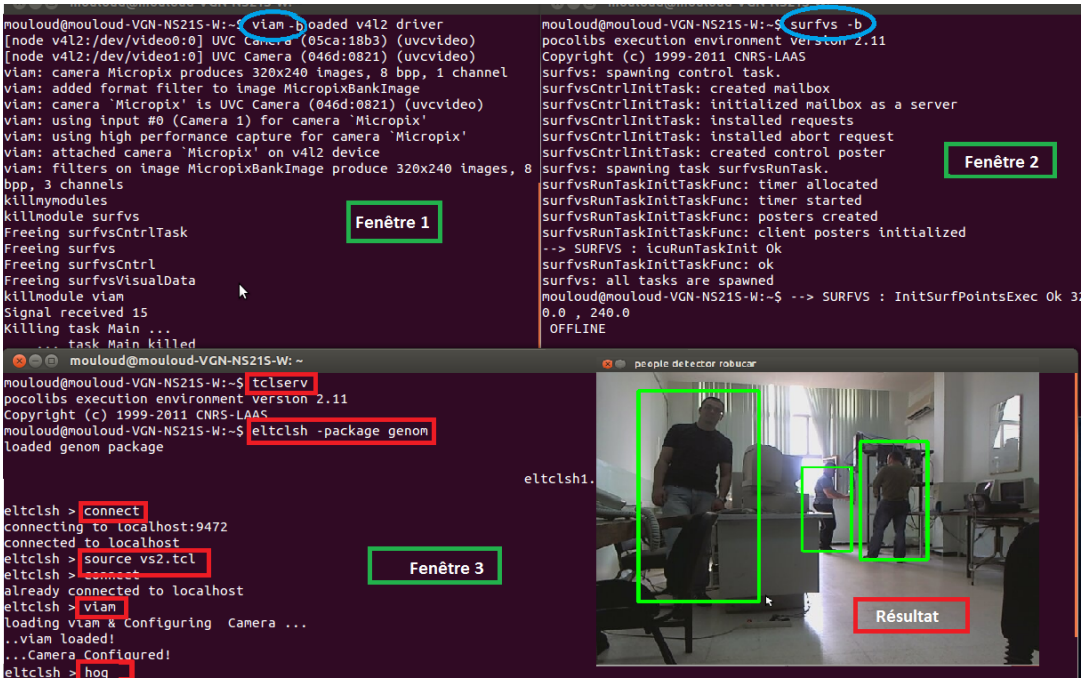


Figure A-2 : Utilisation du module de détection

Remarque : un détecteur est activé à la fois

3.3. Scripte tcl :

```
proc viam { } {
        puts "loading viam & Configuring  Camera ..."
        lm viam
        puts "..viam loaded!"
        ::viam::DriverLoad v4l2
        ::viam::BusPrint
        viam::CameraCreate Micropix v4l2:/dev/video1:0
        viam::BankCreate MicropixBank VIAM_DOUBLE_BUFFERING VIAM_DISABLE
        viam::BankAddCamera MicropixBank Micropix MicropixBankImage
        viam::CameraSetHWMode Micropix  VIAM_HWSZ_320x240
VIAM_HWFMT_YUV422_YUYV  VIAM_HW_FIXED VIAM_HWFPS_FREE
VIAM_HWTRIGGER_INTERNAL
        viam::PushFormatFilter  MicroPix-bayer MicropixBankImage
VIAM_FMTFILTER_FORMAT  VIAM_FILTER_SOFTWARE VIAM_FILTER_MANUAL
VIAM_FILTER_YUV422_YUYV_TO_BGR
        viam::Init
```

```
        viam::Configure MicropixBank
        viam::Acquire -ack MicropixBank 0
        #viam::Display -ack MicropixBank MicropixBankImage VIAM_ON VIAM_OFF 0 0
        (activation du flux camera)
        puts "...Camera Configured!"
}
proc face { } {
        puts "loading surfvs & Configuring   ..."
        lm surfvs
        puts "..surfvs loaded!"
        surfvs::InitSurfPoints 320 240 SURFVS_REF_OFFLINE 0.0001 4 4 2 0
                SURFVS_REF_FINAL
        surfvs::FaceDetection
        puts "...surfvs Configured!"
}
proc hog { } {
        puts "loading surfvs & Configuring   ..."
        lm surfvs
        puts "..surfvs loaded!"
        surfvs::InitSurfPoints 320 240 SURFVS_REF_OFFLINE 0.0001 4 4 2 0
SURFVS_REF_FINAL
        surfvs::PedestrianDetection
        puts "...surfvs Configured!"
```

Bibliographie

[1] B. Zana and C. LeMoine. ‘‘*Autant d'espèces, autant de monde. La perception visuelle: un système de haute technologie*’’, p 817, juin 2001.

[2] http://atilf.atilf.fr/, ‘‘*Définition du robot*’’, ALTIF le dictionnaire de la langue française informatisé,

[3] http://atilf.atilf.fr/, ‘‘*définition de la robotique’’*, ALTIF le dictionnaire de la langue française informatisé.

[4] fr.wikipedia.org, ‘‘*Robot*’’, Wikipédia l’encyclopédie libre.

[5] D. Filliat, Cours ‘‘*Robotique mobile*’’, Octobre 2005.

[6] Alexandre Lampe, ‘‘*Méthodologie d’évaluation du degré d’autonomie d’un robot mobile terrestre*’’, Décembre 2006.

[7] David Filliat, ‘‘*Robotique-Mobile*’’, ENSTA Paris Tech, Octobre 2011.

[8] B. Bayle, Support de cours ‘‘*Robotique mobile*’’, Ecole nationale Supérieure de Physique de Strasbourg, Université de Strasbourg, 2008.

[9] P. Aknin, ‘‘*Capteurs et traitement du signal dans les transports guidés*’’, éditions Hermès/Lavoisier,volume 2, 2002.

[10] H. Bensakhri et M. Derboua, ‘‘*Détection des bords de la chaussée par vision monoculaire pour un robot mobile de type voiture*’’. Université Saad Dahleb, Blida, PFE 2009.

[11] M. Derradji, ‘‘*Capteurs de proximité d’un robot Bipède*’’, rapport de stage INRIA, Grenoble, Juillet 2006.

[12] B.Kazed, Support cours, master2, ‘‘*Intelligence Robotique*’’. Université Saad Dahleb, Blida, 2011-2012.

[13] David Filliat, cours C10-2 ‘‘*Matériels courants en robotique*’’, ENSTA, octobre 2004.

[14] M. Badeche, ‘‘*Suivi de modèle et Incrustation d’Objets Virtuels pour la Réalité Augmentée* ’’, Thèse de magistère, Université de Batna, 2006.

[15] http://www.map.toulouse.archi.fr, ‘‘*Support de cours*’’, Ecole nationale supérieure d’architecture de Toulouse, 2002.

[16] D. Loulha& M. Boutheldja, ‘‘*Extraction d’informations et modélisation d’une carte de contours par la transformée de hough*’’, PFE 2003.

[17] Daniel Pourcelot, ‘‘*Histogramme*’’, Club Besançon Déclic image, octobre 2010.

[18] Jeff Schewe, ‘‘*traitement d’images*’’, comment ça marche.net, octobre 2008.

[19] B. GUY, ‘‘*Analyse d’image*’’, Axe Génie des Procédés, centre SPIN, Ecole des Mines de Saint-Etienne, 2010.

[20] Vincent BARRA, Cours de ‘‘*traitement d’images*’’.

[21] Z. Hai-bo, Y. Kui, L. Jin-dong, ‘‘*A Fast and Robust Vision System for Autonomous*

Mobile Robots Proc’’, International conference on Robotics, Intelligent Systems and Signal Processing, China, volume1 p. 60-65, Octobre 2003.

[22] McLauchlan, P. F. and J. Malik, ‘‘*Vision for Longitudinal Vehicle Control*”, in Proceedings of the Eighth British Machine Vision Conference (BMVC’97), 1997.

[23] Blake, A. and A. Yuille, ‘‘*Active Vision*’’, MIT press, November, 1992.

[24] Blake, A. and M. Isard, ‘‘*Active Contours*’’, Springer-Verlag, London, 1998.

[25] D. Koller, J. Weber and J. Malik, ‘‘*Robust multiple car tracking with occlusion reasoning in Proc*’’. 3rd European Conf. on Computer Vision (ECCV’94), Stockholm, vol. 1, pp. 189-196, May, 1994.

[26] Smith S.M. and J.M. Brady, ‘‘*ASSET-2: Real-Time Motion Segmentation and Shape Tracking*’’, in IEEE. Trans. on Pattern Analysis and Machine Intelligence, vol. 17, No. 8, pp. 814-820, August 1995.

[27] B.Bascle, P. Bouthemy, R. Deriche, and F. Meyer, ‘‘*Tracking complex primitives in an image sequence*’’, Technical Report 2428, INRIA, Sophia-Antipolis, France, December 1994.

[28] T. Drummond, and R. Cipolla, ‘‘*Real-time tracking of complex structures with online camera calibration*’’, in Image and Vision Computing, vol. 20, No. 5-6, pp. 427-433, 2002.

[29] Dellaert F., C. Thorpe, and S. Thrun, ‘‘*Super-Resolved Texture Tracking of Planar Surface Patches”*, in Proc. of IEEE/RSJ International Conference on Intelligent Robotic Systems, October, 1998.

[30] N.X.Dao, BJ.You, SR.Oh and M. Hwangbo, ‘‘*Visual Self-Localization for Indoor Mobile Robots Using Natural Lines*’’, in Proc. of the 2003 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems IROS’2003, pp. 1252-1257, Las Vegas, Nevada, October, 2003.

[31] G. N. DeSouza, and Avinash C. Kak, ‘‘*Vision for Mobile Robot Navigation: A Survey*”, in IEEE. Trans. on Pattern Analysis and Machine Intelligence, vol. 24, No. 2, pp. 237-267, February, 2002.

[32] http://fr.wikipedia.org/wiki/, ‘‘*Détection de personne*’’, Wikipédia l’encyclopédie libre.

[33] Yannick Benezeth, ‘‘ *Détection de la présence humaine par vision*’’, Thèse Université d'Orléans, page 31, Octobre 2009.

[34] Paul Viola et Michael Jones, ‘‘*Robust Real-time Object Detection*’’, IJCV,workshop on SCTV, Vancouver, Canada, Juillet 2001.

[35] B. Heisele et C. Wöhler, ‘‘*Motion-based recognition of pedestrians*’’, Proceedings of the 14th International Conference on Pattern Recognition, vol. 2, p. 1325-1330,Brisbane, Australie, 16-20 août 1998.

[36] P. Viola, M. Jones et D. Snow, ‘‘*Detecting Pedestrians using Patterns of Motion and Appearance*’’, dans IJCV, vol. 63, no 2,p. 153-161,2005.

[37] Navneet Dalal et Bill Triggs, ‘‘*Histograms of Oriented Gradients for Human Detection*’’, Conference on Computer Vision and Pattern Recognition IEEE, p.886 - 893, San Diego, USA, Juin 2005.

[38] Markus Enzweiler, Dariu Gavrila, ‘‘*Monocular Pedestrian Detection:Survey and Experiments*’’, TPAMI, Vol 31, n° 12, p. 2179-2195, Décembre 2009.

[39] Oncel Tuzel, Fatih Porikli et Peter Meer, ‘‘*Pedestrian detection via classification on Riemannian manifolds*”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 30, no 10, p. 1713-1727, Octobre 2008.

[40] Y. Mu, S.Yan, , Y.Liu, T.Huang, and B.Zhou, “*Discriminative local binary patterns for human detection in personal album*”, CVPR,p 1–8, 2008.

[41] J. Zhou, and J.Hoang, “*Real time robust human detection and tracking system*”, CVPR,p.149,Canada, Juin 2005.

[42] Jae-Yeong Lee and Wonpil Yu, “*Moving Object Tracking in Driving Environment*”, URAI, p.139-141,Korea 23-26 Nov 2011.

[43] D.M. Gavrila, J. Giebel and S. Munder, “*Vision-Based Pedestrian Detection The PROTECTOR System*”,conference ECCV,volume 3952,p.428-441,Parma Italy 2006.

[44] C.Papageorgiou, T .Poggio, "*Trainable pedestrian detection*", ICIP, vol. 4, p. 35-39, 1999.

[45] O.Tuzel, F.Porikli, P.Meer "*Region Covariance: A Fast Descriptor for Detection and Classification*", European Conference on Computer Vision (ECCV), May 2006.

[46] Xiaoyu Wang, Tony X. Han, Shuicheng Yan, "*An HOG-LBP Human Detector with Partial Occlusion Handling*", p.32-39, ICCV 2009.

[47] C. Wojek and B. Schiele. "*A performance evaluation of single and multi-feature people detection*" .Pattern recognition, In DAGM, 2008

[48] P.Dollar, C. Wojek, B.Schiele, P. Perona, "*Pedestrian Detection: A Benchmark*", CVPR, p. 304-311, Canada, Juin 2009.

[49] Bo Wu and Ram Nevatia, "*Detection of Multiple, Partially Occluded Humans in a Single Image by Bayesian Combination of Edgelet Part Detectors*", IEEE International Conference on Computer Vision, pages 90-97, 2005

[50] K. Mikolajczyk, and C. Schmid, and A. Zisserman, "*Human detection based on a probabilistic assembly of robust part detectors*", The European Conference on Computer Vision (ECCV), vol. 3021/2004, p. 69-82, 2005.

[51] Q. Zhu, S. Avidan, M. Yeh, and K. Cheng, "*Fast Human Detection Using a Cascade of Histograms of Oriented Gradients*", Proc. IEEE International Conference on Computer Vision and Pattern Recognition, p. 1491-1498, 2006

[52] http://pascal.inrialpes.fr/data/human/, ''*Base d'images de personnes de l'INRIA*''.

[53] http://www.science.uva.nl/research/isla/downloads/pedestrians/index.html, ''*Bases d'images et de vidéos Daimler pour la détection de piétons*''.

[54] Philip Kelly, Noel E. O'Connor, Alan F. ''*Smeaton, A Framework for Evaluating Stereo-Based Pedestrian Detection Techniques IEEE Transactions on Circuits and Systems for Video Technology*'', Volume 8, p. 1163-1167 , 2008.

[55] Sara Fleury, Matthieu Herrb, ''*manuel GenoM*'', 4 décembre 2003.

[56] Claire MAHEO, ''*Méthodes de suivi d'un objet en mouvement sur une vidéo* '', Université Espagne du 15/06/09.