PDF-XChange Viewer Click to buy NOW! www.docu-track.com

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE MOULOUD MAMMERI, TIZI-OUZOU

FACULTE DE GENIE ELECTRIQUE ET DE L'INFORMATIQUE
DEPARTEMENT D'INFORMATIQUE

MEMOIRE DE MASTER

Présenté par :

M^r^ DJEBARRI Hacene et M^elle^ KHERBACHE Ghania

En vue de l'obtention du diplôme de Master en Informatique

Option Système informatique

Intitulé :

Détection de piétons pour la navigation en sureté des véhicules intelligents

Encadré par : Mr. Lotfi HOCINI

Soutenu le : 17/07/2014

PDF-XChange Viewer Click to buy NOW! www.docu-track.com

Remerciement

C'est avec un grand plaisir que nous réservons cette page, en signe de gratitude et de reconnaissance à tous ceux qui nous ont aidé à la réalisation de ce travail.

Mais avant toute personne, nous devrons remercier Dieu qui nous a donné la fois, la santé, le courage et la volonté pour terminer ce travail.

Nous tenons à présenter nos remerciements A notre promoteur Mr Hocini pour nous a fait confiance et pour nous avoir encouragés tout au long de ce projet.

Nous tenons aussi à exprimer toutes notre gratitude aux membres du jury pour avoir accepté d'évaluer et de juger notre travail.

Et aussi à toutes les personnes qui ont contribué de loin ou de prés au bon déroulement de notre travail.

PDF-XChange Viewer Click to buy NOW! www.docu-track.com

DEDICACE

Je dédie ce modeste travail :

A la mémoire de mon cher père que Dieu puisse l'accueillir dans son vaste paradis.

A la mémoire de la mère de mon promoteur, que dieu bénisse son âme.

*Nous tenons à exprimer notre profonde gratitude à notre promoteur M^r^* ***L.HOCINI***

A ma très chère mère pour son amour, son soutien et son sacrifice et tous les efforts qu'elle ma donnée le long de mon parcours et je la souhaite bonne santé et une longue vie.

A mon très cher frère Ahmed qui je l'aime énormément.

A mes chères sœurs : Yamna, Zahia, Kenza, Nadira celles qui m'ont aidé et soutenir à tous moment.

A mes grands parents : Zahra et Said.

A mes chers oncles : Arezki, Ahcen, Nadir, Seddik, Hamid.

Et surtout à mon chèr oncle Hakim et tata Taous qui sont toujours là pour me consoler et soutenir.

Et à ma chère tante Nadira.

Et à mon binôme Hacene.

Et bien sur à tous mes amis (es) que j'aime.

Et à rafik.

Et à sofiane.

Et à tous ceux et celles qui m'ont aidé à réaliser ce travail.

******Ghania******

PDF-XChange Viewer Click to buy NOW! www.docu-track.com

DEDICACE

Je dédie ce modeste travail :

A la mémoire de mon cher ami Malik et la mère de mon promoteur, que dieu bénisse leur âmes.

Nous tenons à exprimer notre profonde gratitude à notre promoteur M^r ***L.HOCINI***

A ma très chère mère pour son amour, son soutien et son sacrifice et tous les efforts qu'elle ma donnée le long de mon parcours et je la souhaite bonne santé et une longue vie.

A mes très chers frères.

A toute les familles DJEBARRI et MAMERI et DOMENICA et CHERFI.

Et à mon binôme GHANIA.

Et bien sur à tous mes amis (es).

Et à tous ceux et celles qui m'ont aidé à réaliser ce travail.

******Hacene******

SOMMAIRE

SOMMAIRE

Liste des figures

PDF-XChange Viewer
Click to buy NOW!
www.docu-track.com

Conclusion générale

Introduction :

Tous les ans, dans le monde, plus de 1,2 millions de personnes meurent sur la route [26], notamment en Algérie qui se place dans le podium pour le nombre d'accidents routiers annuel. Selon le « rapport mondial sur la prévention des traumatismes dus aux accidents de la route » [26], paru en 2004, la plupart des piétons blessés et tués oscille entre 10 et 50% des victimes de l'insécurité routière.

Le but de ce mémoire est de contribuer à la réalisation d'un système de protection de piétons qui se concentre principalement sur l'utilisation des informations acquises des différents dispositifs électroniques installés sur les véhicules. Ce système fait partie des systèmes d'aide à la conduite qui informent ou assistent le conducteur afin d'éviter l'apparition de situations dangereuses. Les exemples de ces systèmes sont nombreux : le système anti-blocage de freins ABS, le contrôle de trajectoire ESP, les systèmes de navigation GPS, l'allumage automatique des feux de croisement, le radar de recul, un système reconnaissant automatiquement la signalisation routière permettrait par exemple d'informer le conducteur s'il dépasse la vitesse autorisée ou s'il se trouve à proximité d'une école et la détection d'un piéton en utilisant la vision par ordinateur,…etc. Un système reconnaissant automatiquement la signalisation routière permettrait par exemple d'informer le conducteur s'il dépasse la vitesse autorisée ou s'il se trouve à proximité d'une école et la détection d'un piéton avec vision par ordinateur.

La vision par ordinateur est une branche de l'intelligence artificielle dont le but est de permettre à une machine de comprendre ce qu'elle «voit » lorsqu'on la connecte à une ou plusieurs caméras. Elle peut servir entre autre à la reconnaissance de formes, qui consiste à reconnaitre une forme dans une image après l'avoir enregistrée.

A l'heure actuelle, les systèmes de vision sont de plus en plus répandus (webcam, caméscope...) et les caméras se sont installées partout dans notre quotidien. Elles sont utilisées pour réaliser de la vidéosurveillance (dans les magasins, rues ou aéroports), de l'aide à la conduite (aide au guidage ou détection d'obstacles), et bien d'autres applications encore... Pour l'être humain, voir est une tâche innée et nous ne mesurons souvent pas la difficulté pour obtenir les mêmes performances artificiellement. Malgré les avancées de la vision par ordinateur, les systèmes développés sont très loin d'égaler les performances de l'œil et du cerveau humains.

Notre travail se déroule dans ce cadre complexe mais stimulant. Il s'agit de mettre au point un système capable de détecter des piétons à partir d'images provenant d'une simple caméra. Le problème est d'autant plus difficile que le système de vision est embarqué dans un véhicule se déplaçant en milieu urbain : l'environnement visuel contient de nombreux éléments, est sujet à de multiples variations, et de plus, le processus doit prendre place en temps réel, pour éviter une éventuelle collision.

La problématique principale est alors de reconnaître les piétons présents dans les images, c'est-à-dire être capable d'indiquer si les objets présents dans l'environnement devant le véhicule appartiennent ou non à la classe piétons.

Pour cela nous présentons les différentes étapes permettant de réaliser ce système de détection et de reconnaissance de piétons. Il ne s'agit pas de fournir ici une description exhaustive de l'état de l'art, ce qui ne saurait être réalisable dans ce mémoire tant les méthodes sont diverses

PDF-XChange Viewer
Click to buy NOW!
www.docu-track.com

Chapitre I :

Etat de l'Art sur l'apprentissage et la classification

I)- Introduction :

Les méthodes de classification ont pour but d'identifier les classes auxquelles appartiennent des objets à partir de certains paramètres descriptifs. Elles s'appliquent à un grand nombre d'activités humaines et conviennent en particulier au problème de la prise de décision automatisée. La procédure de classification sera extraite automatiquement à partir d'un ensemble d'exemples. Un exemple consiste en la description d'un cas avec la classification correspondante. Un système d'apprentissage doit alors, à partir de cet ensemble d'exemples, extraire une procédure de classification, il s'agit en effet d'extraire une règle générale à partir des données observées. La procédure générée devra classifier correctement les exemples de l'échantillon et avoir un bon pouvoir prédictif pour classifier correctement de nouvelles descriptions.
Les méthodes utilisées pour la classification et l'apprentissage sont nombreuses, citons : la méthode des Séparateurs à Vastes Marges (SVM), les Réseaux de Neurones, le K-Plus Proche Voisins, … etc. Nous présentons dans la suite de ce chapitre une étude détaillée de ces méthodes. Qui ont montrés leurs efficacités dans de nombreux domaines d'applications tels que le traitement d'image, la détection d'objets, la catégorisation de textes et le diagnostique médicale.

II)- Apprentissage automatique:

C'est un domaine sur lequel beaucoup de chercheurs se concentrent, l'apprentissage a pendant longtemps opposé les chercheurs où chacun tentait de prouver la suprématie de sa méthode. Aujourd'hui une opinion s'est établie autour de l'idée qu'il n'y a pas de meilleure méthode. Chacune est plus ou moins adaptée à un problème posé.

II.1)- Les différents types d'apprentissage :

Les méthodes d'apprentissage peuvent être classées en plusieurs catégories : apprentissage supervisé, apprentissage semi-supervisé, apprentissage non-supervisé et apprentissage par renforcement.

II.1. 1)- Apprentissage supervisé :

L'apprentissage supervisé a pour but d'apprendre par l'exemple. Il faut fournir au préalable une liste d'objets avec leurs étiquettes de classe appelée « ensemble d'apprentissage » afin que le système soit capable d'expliquer et ensuite de prédire l'appartenance d'un nouvel objet à une classe connue a priori. Beaucoup d'algorithmes d'apprentissage supervisé sont utilisés pour faire de la reconnaissance d'objets : caractères, visages, personnes...

Il existe plusieurs méthodes d'apprentissage supervisé comme SVM, Adaboost, réseau de neurons…etc.

II.1.1.1)- Le Boosting :

Le principe a été mis en place par Schapire **[1].** Il a montré qu'un algorithme d'apprentissage qui produit un classifieur faible peut être amélioré s'il est entraîné sur trois échantillons choisis parmi l'ensemble d'apprentissage initial. La combinaison des trois règles permet d'obtenir une règle de décision forte.

Soit S l'ensemble d'apprentissage comprenant N exemples. Nous nous plaçons dans le cas d'un problème de classification en deux classes. L'algorithme fournit à chaque apprentissage une hypothèse h, c'est-à-dire un scalaire indiquant à quelle classe appartient l'objet.

- l'algorithme d'apprentissage est entraîné sur un échantillon S_1 comprenant n_1 exemples sélectionnés dans S_1 (avec n 1 < N) ; un premier classifieur faible C_1 est obtenu ; il fournit une première règle de décision h_1 ;

- 2. un échantillon de n_2 exemples dans l'ensemble $(S - S_1)$ est sélectionné de telle sorte que parmi ces échantillons la moitié soit correctement classée par C_1, l'autre non. Un deuxième classifieur faible C_2 est alors obtenu ; il produit une règle de classification h_2 ;

- 3. enfin un classifieur faible C_3 est créé en l'entraînant sur un échantillon S_3 comprenant n_3 exemples sélectionnés dans $(S - S_1 - S_2)$ pour lesquels C_1 et C_2 ne fournissent pas la même réponse ;

- 4. le classifieur final C a pour règle de décision H telle que : H = vote majoritaire (h_1, h_2, h_3).

Un exemple simple de cette méthode est tiré du livre d'A. Cornuéjols et L. Miclet **[2],** il est fourni dans les figures suivantes. Bien sûr, les trois échantillons S_1, S_2 et S_3 tendent à recouvrir entièrement S afin de tirer le maximum d'informations de tout l'ensemble d'apprentissage. En pratique, il arrive souvent que le premier classifieur utilisé classe correctement un maximum d'exemples. Le découpage en sous-échantillons se fait généralement de façon empirique, en fonction du classifieur et des données présentes. Le processus présenté ci-dessus peut être réalisé de façon récursive pour chaque classifieur faible et ainsi traiter de 9 voire 27 sous-ensembles.
Toutefois, avec ce type d'apprentissage, il est envisageable de remplacer la fonction de test par une distribution de probabilités sur tout l'ensemble d'apprentissage afin d'obtenir une meilleure généralisation. La méthode qui suit reprend précisément cette idée.

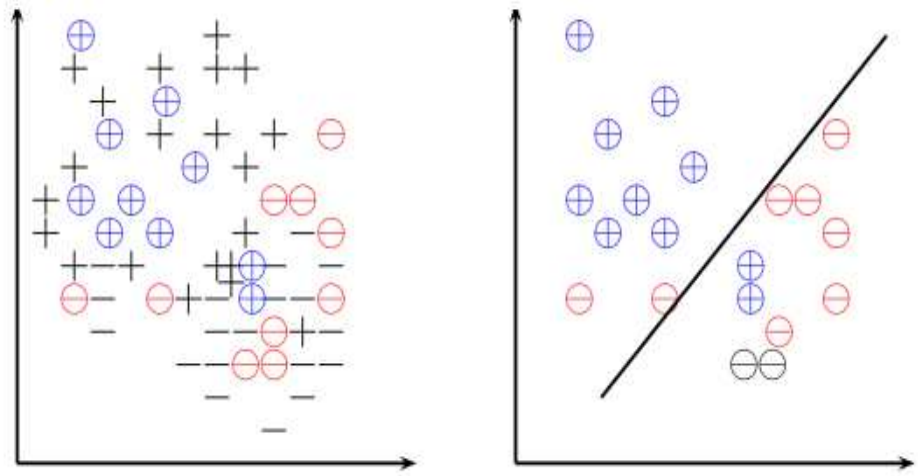


Figure I.1: Exemple de Boosting sur un exemple simple - première étape **[2].**

A gauche, l'ensemble d'apprentissage S comprend tous les exemples positifs (+) et négatifs (-). Le sous-ensemble S_1 est constitué d'un échantillon de S (points cerclés en bleu pour les exemples positifs et en rouge pour les négatifs).
A droite, l'ensemble S et la droite C_1 apprise sur celui-ci.

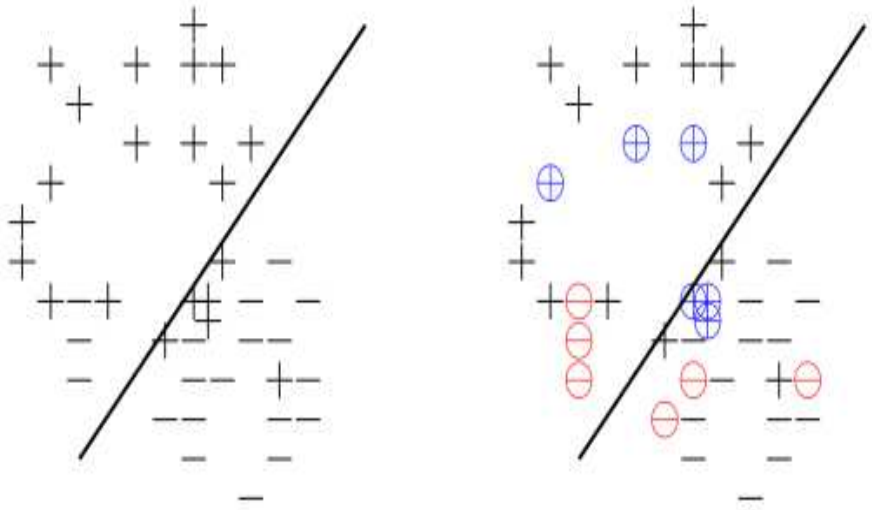


Figure I.2: Exemple de Boosting sur un exemple simple - deuxième étape **[2]**

A gauche, l'ensemble $S - S_1$ et la droite C_1 apprise sur S_1.
A droite, un ensemble S_2 (points cerclés) inclus dans $S - S_1$, comprenant les exemples les plus informatifs pour C_1.

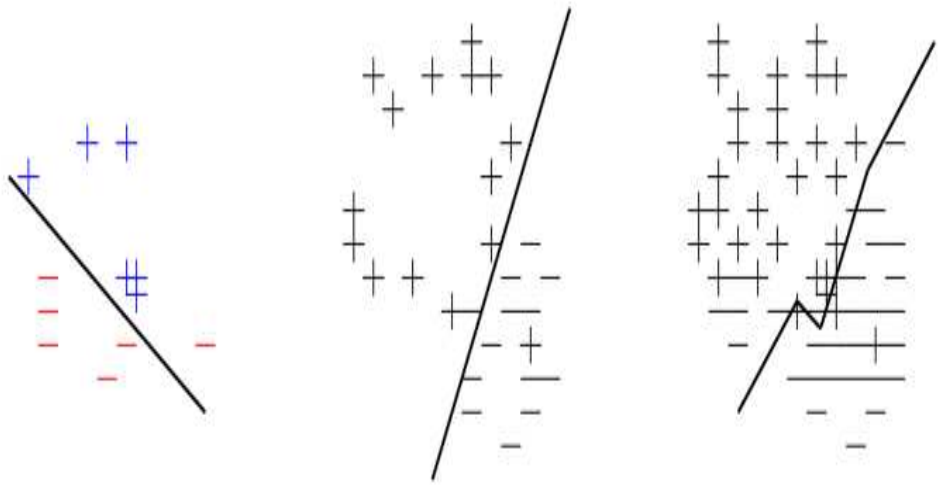


Figure I.3: Exemple de Boosting sur un exemple simple - dernière étape **[2]**

A gauche, l'ensemble S_2 et la droite C_2 apprise sur celui-ci.
Au centre, l'ensemble $S_3 = S - S_1 - S_2$ et la droite C_3 apprise sur S_3.
A droite, l'ensemble d'apprentissage S et la combinaison des 3 droites, qui permet de classifier correctement tous les exemples.

II.1.1.2)- Adaptative Boosting (AdaBoost) :

Introduit en 1996 par Freund et Schapire **[3],** l'AdaBoost se base sur le principe fondamental du Boosting et énonce que l'avis de plusieurs experts est meilleur que celui d'un seul.
Le but est donc là encore d'associer plusieurs classifieurs faibles afin de créer un classifieur fort. La nouveauté introduite par l'AdaBoost est de proposer une distribution de probabilités apriori sur tout l'ensemble d'apprentissage en fonction de la réponse de l'algorithme à l'itération précédente. De cette façon, il est facile d'intégrer un grand nombre de classifieurs faibles dans la création du classifieur final.

Notons S l'ensemble d'apprentissage composé de N exemples tel que : $S= \{x_i, y_i\}_{i=1,...N}$.
$\mathbf{x_i}$ étant le vecteur de caractéristiques d'un exemple et y_i étant un scalaire indiquant la classe de l'objet. Dans le cas présent d'une classification en deux classes, $y_i = 1$ si l'objet appartient à la classe d'objet recherché (exemple positif), et $y_i = -1$ sinon (exemple négatif). Notons p_t un vecteur poids dont chaque composante est le poids associé à chaque exemple ; initialement $p_t = p_1$ et tous les éléments ont la même valeur. Un échantillon, noté S_1 et comprenant n_1 exemples (avec $n_1 < N$), est ensuite sélectionné parmi l'ensemble d'apprentissage. Un classifieur C_1 est entraîné sur cet échantillon, qui fournit une hypothèse h_1 correspondant directement à la classe de l'objet. Cette règle de décision est ensuite appliquée pour tous les exemples compris dans S afin de calculer l'erreur ε_1 du classifieur C_1 sur l'ensemble S. Le poids correspondant à chaque exemple est ensuite mis à jour en fonction du résultat de la

classification. Si un exemple est bien classé, son poids diminue et s'il est mal classé son poids augmente. Pour cela, un coefficient α 1 est calculé tel que :

$$\alpha 1 = \frac{1}{2}\ln\left(\frac{1-\epsilon 1}{\epsilon 1}\right)$$

Le poids des différents exemples vaut :

$p2(xi) = \frac{p1(xi)}{Z1} e^{-\alpha 1}$ Si $h_1(x_i) = y_i$

$p2(xi) = \frac{p1(xi)}{Z1} e^{+\alpha 1}$ Si $h_1(x_i) \neq y_i$

Z_1 est un terme de normalisation tel que $\sum_{i=1}^{m} p1(xi) = 1$.

Après cette étape de mise à jour des poids, un nouvel échantillon d'exemples est sélectionné. Les exemples choisis en priorité sont ceux dont les poids correspondants sont les plus élevés. Puis un nouveau classifieur est entraîné sur cet échantillon. De cette façon, ce nouveau classifieur créé se focalisera sur les exemples qui sont jusque là mal classés. Les poids sont alors remis à jour et le processus est réitéré t fois (avec t < T) tant que l'erreur globale sur tout l'ensemble d'apprentissage ϵ_T n'est pas nulle. Un ensemble de T classifieurs faibles est alors obtenu, qui constitue un classifieur fort C. A la fin de l'algorithme, chaque classifieur faible voit sa règle de décision pondérée par une valeur α_t calculée au cours de l'apprentissage ; pour la classification d'un nouvel exemple, le classifieur fort fournit ainsi l'hypothèse suivante:

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha\, t\, ht(x)\right)$$

Pour un objet ayant pour vecteur de caractéristiques x, si H(x) ≥ 0, l'objet est associé à la classe des positifs, et si H(x) < 0 il est associé à la classe des négatifs. L'algorithme ainsi créé est résumé dans l'algorithme 1. Un exemple de classification est présenté figures 4, 5 et 6 (exemple tiré de **[4]**).

En pratique, un seuil est fixé et lorsque l'erreur globale sera en deçà de celui-ci, l'algorithme arrête d'ajouter de nouveaux classifieurs faibles. En effet, lorsqu'il ne reste que quelques exemples mal classés, les nouveaux classifieurs ajoutés vont exclusivement se focaliser sur ces exemples et n'apporteront pas une réelle amélioration sur la règle de décision finale.

Le but étant de créer un classifieur fort capable de généraliser correctement pour de nouveaux exemples inconnus, ajouter des classifieurs faibles seulement pour quelques exemples n'est pas utile et risque même de détériorer les résultats du classifieur fort final (problème de surapprentissage).

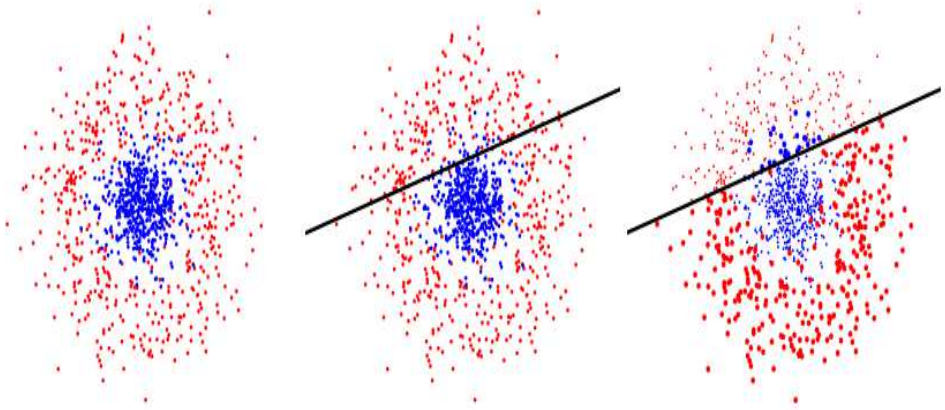


Figure I.4: Exemple d'apprentissage par un algorithme AdaBoost - ajout d'un premier classifieur faible **[4]**

A droite, l'ensemble de départ ; au milieu, ajout d'un premier classifieur faible ; à gauche, le poids des exemples mal classés augmentent.

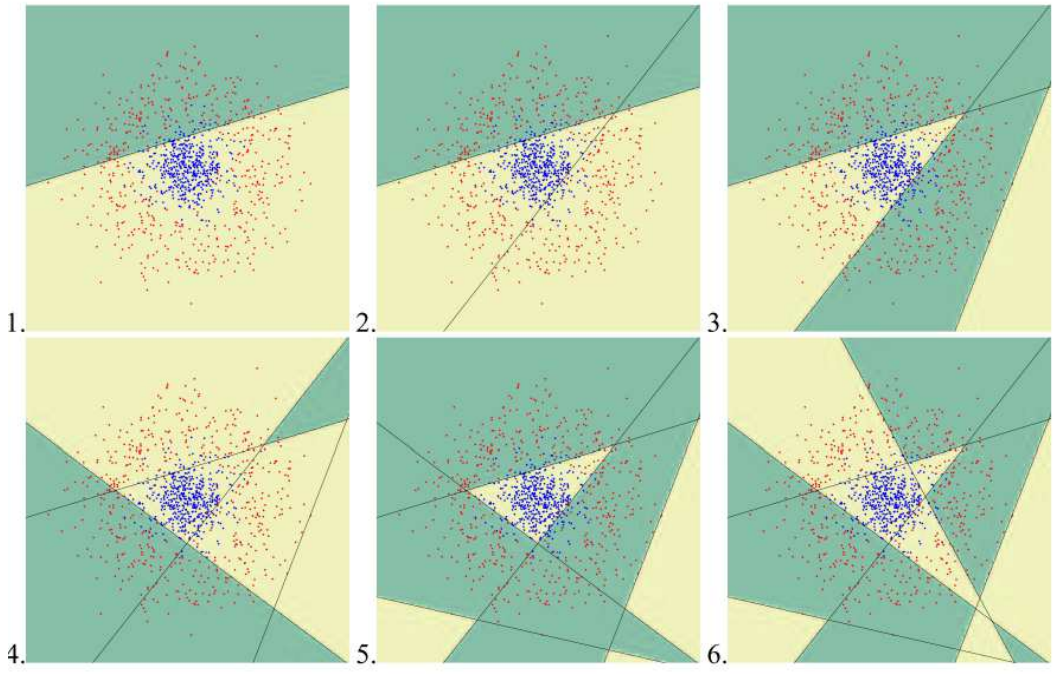


Figure I.5: Exemple d'apprentissage par un algorithme AdaBoost - combinaison de plusieurs classifieurs faibles **[4]**

A chaque itération, un nouveau classifieur faible est ajouté, et son avis est ajouté à la règle de décision de façon à séparer au mieux les deux classes d'objets.

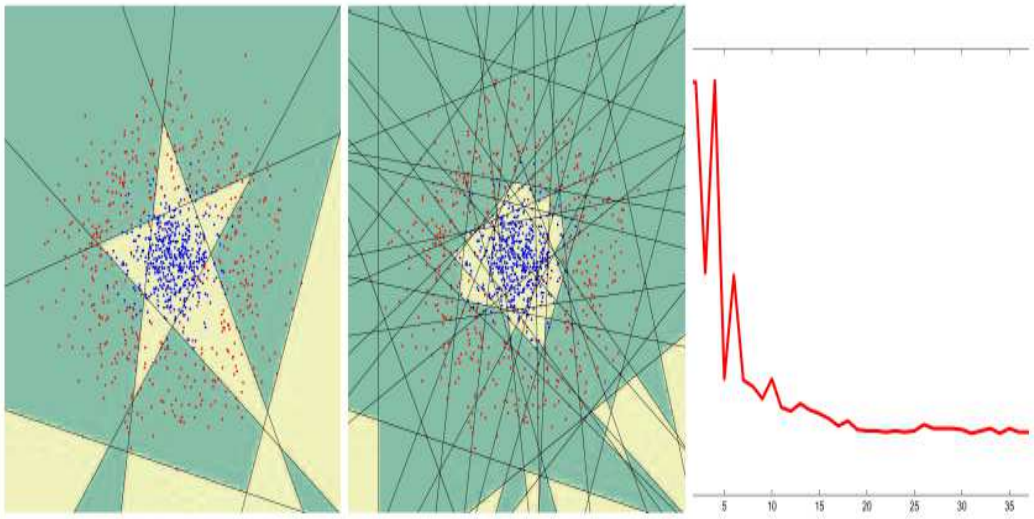


Figure I.6: Exemple d'apprentissage par un algorithme AdaBoost - fin **[4]**

Au fur et à mesure que les classifieurs faibles sont ajoutés, l'erreur de classification sur l'ensemble d'apprentissage converge vers une valeur (ici environ 5%). Au-delà de cette valeur, les nouveaux classifieurs doivent se concentrer à classer les derniers exemples «difficiles ».
L'algorithme d'Adaboost:

ENTREES :
Soit un ensemble d'apprentissage tel que
S= $\{(x_i, y_i)\}_{i=1\ldots N}$,
Avec,$x(i) \in R^M$,
Et $y_i \in +1, -1\}$, label classe.

Pour i= {1,…, N} **faire**
$P_i\ (X_i) \leftarrow \frac{1}{N}$
$i = i + 1$
Fin pour $i = N$

Pour $t = \{1, \ldots, N\}$**faire**
Tirer un échantillon d'apprentissage S_t dans S selon les probabilités p_t.
Entraîner un classifieur C_t sur ce sous-échantillon.
Soit ε_tl'erreur apparente de C_t sur S.
Calculer $\alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$

pour $i = \{1, \dots, N\}$ **faire**

-si $h_t(x_i) = y_i$ (bien classé par h_t):

$$p_{t+1}(\boldsymbol{x_i}) \leftarrow \frac{p_t(\boldsymbol{x_i})}{z_i} \boldsymbol{e^{-\alpha_t}}$$

-si $h_t(x_i) \neq y_i$ (mal classé par h_t):

$$p_{t+1}(\boldsymbol{x_i}) \leftarrow \frac{p_t(\boldsymbol{x_i})}{z_i} \boldsymbol{e^{+\alpha_t}}$$

-i=i+1

fin pour *i=N*

-T=t+1 ;

fin pour *t=T*

NB : Z_t est une valeur de normalisation telle que : $\sum_{i=1}^{m} p_t(x_i) = 1$.

SORTIES : un classifieur fort C constitué d'un ensemble de classifieurs faibles $C_1 \dots \dots, C_T$ tel que $\epsilon_T \approx 0$ fournissant l'hypothèse finale : $H(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$ pour un nouvel exemple x. La règle de décision pour attribuer la classe *y* de cet exemple est: $y = sign(H(x) \geq 0)$, avec la convention $sign(0) = 1$.

II.1.1.3)- Séparateur à Vastes Marges (SVM) :

C'est en 1979 que Vladimir Vapnik introduit le Séparateur à Vastes Marges(SVM). Cependant, c'est depuis le début des années 90 que le SVM est beaucoup utilisé par la communauté scientifique. L'acronyme SVM remplace Support Vectors Machine (Machine à Vecteurs de Support) ou Séparateur à Vastes Marges. Dans ce cas, la classification est faite en recherchant un hyperplan optimal qui maximise les marges entre les exemples de la classe positive et de la classe négative. La séparation entre les classes est réalisée dans un espace de redescription des données de très grande dimension.

Papageorgiou et al. **[5]** ont utilisé avec succès les SVM dans le cas de la détection d'humains. Dans les premières versions de leur système de détection, ils utilisent les SVM avec une représentation des humains basée sur les filtres de Haar. Dans les versions suivantes, ils utilisent plusieurs SVM, un pour chaque partie du corps humain **[6]** et ils combinent ensuite les résultats de chaque classifieur avec des contraintes géométriques. Dalal et Triggs [7] utilisent aussi un SVM linéaire pour détecter la présence humaine en combinaison avec les histogrammes des gradients orientés. Sidenbladh **[8]** utilise les SVM pour apprendre un modèle de la marche humaine réalisé avec le flot optique. Yoon et Kim **[9]** utilisent aussi les SVM en combinaison avec une carte des distances de Mahalanobis.

Pour s'en tenir au principe de base de cette technique, les SVM effectuent la classification de données entre deux classes en déterminant un hyperplan séparateur maximisant la distance (« marge ») entre ce plan et les points de l'ensemble d'apprentissage, tout en minimisant les erreurs. La Figure 7 illustre des exemples d'hyperplans séparant les données de deux classes. On voit dans la Figure7 (a) trois exemples d'hyperplans séparant les deux classes parmi une infinité de solutions possibles. La Figure7 (b) illustre le cas de l'hyperplan optimal au sens de la maximisation de la marge entre les données. Comme on le verra, la définition de cet hyperplan optimal fait intervenir certains des points (les « points supports » ou « support vectors ») de l'ensemble d'apprentissage. Sur l'illustration ci-dessous, ceux-ci apparaissent en rouge.

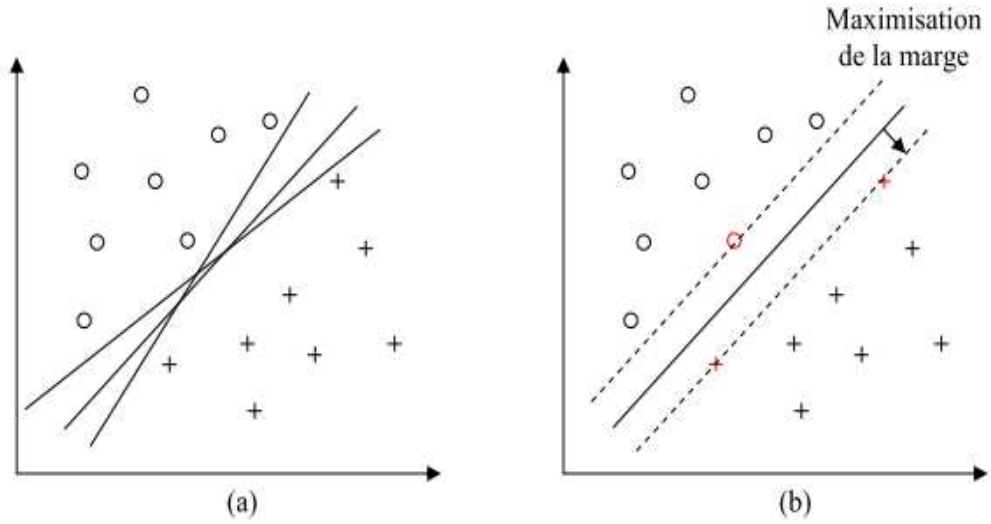


Figure I.7: Exemples d'hyperplans séparant les données de deux classes.

(a) exemples d'hyperplans parmi une infinité de solutions possibles.
(b) l'hyperplan optimal au sens de la maximisation de la marge entre des données.
Les points rouges sont les points supports.

II.1.1.4)- Relevance Vector Machine (RVM) :

La méthode des RVM pour « Relevance Vector Machine » c'est une méthode qui permet de traiter des problèmes de régression. Elle utilise le modèle classique linéaire des machines à noyau des SVM ($h(x) = \sum_{j=1}^{M} wjxi + b = (w.x) + b$) mais fait appel à une formulation bayésienne pour déterminer les paramètres et sélectionner les exemples pertinents qui permettront de réaliser le modèle discriminant final.

Les RVM se basent sur le même modèle que les SVM. Ces deux méthodes construisent un modèle linéaire épars - ou parcimonieux - dans le sens où seul un certain nombre de vecteurs d'apprentissage est conservé : les vecteurs supports des SVM et les vecteurs « relevant » des RVM.

Même si l'apprentissage est hors-ligne, il faut toutefois une puissance de calcul plus conséquente pour les RVM qui requièrent d'inverser au moins à la première itération une matrice de taille N × N, N étant le nombre de données comprises dans l'ensemble d'apprentissage.

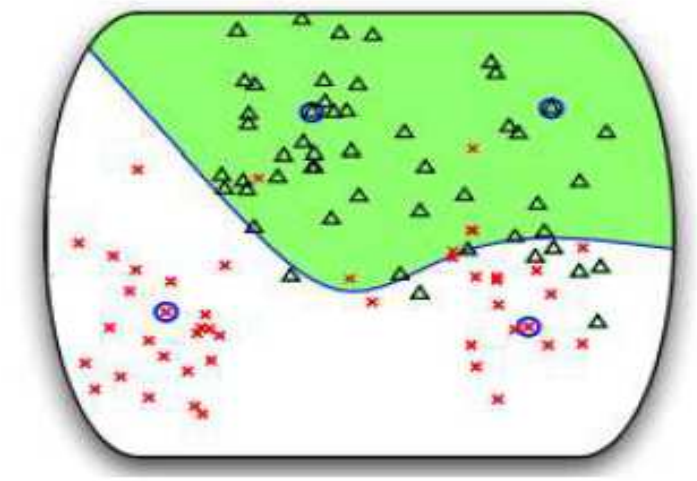


Figure I.8: Exemple de classification par les RVM.

Les croix rouges et les triangles noirs représentent les points d'apprentissage de deux classes différentes. Les points cerclés en bleu représentent les vecteurs « relevant » des RVM. La surface verte représente la région de l'espace des paramètres à laquelle appartiennent les points de la classe des triangles noirs et la surface blanche représente la région à laquelle appartiennent les points de la classe des croix rouges.

II.1.1.5)- K-Plus Proches Voisins:

L'algorithme des « K-Plus Proches Voisins » est la méthode la plus classique (KPPV) **[10].** Le principe est simple et consiste à calculer la distance d'un nouvel objet par rapport à ceux dont la classe est déjà connue, par exemple avec une distance euclidienne ; le nouvel objet appartiendra à la classe dont il est le plus proche. Cette technique donne d'assez bons résultats dans des cas simples et est facile à mettre en œuvre car elle est non-paramétrique.

Toutefois le temps de calcul de la prédiction est assez long car il nécessite un calcul de distance à tous les éléments de la base d'apprentissage. La **figure 9** montre un exemple simple d'une classification par un algorithme des KPPV avec un calcul de distance euclidienne : le nouvel exemple est associé à la classe B car la majorité de ses plus proches voisins appartiennent à la classe B (au sens de la norme euclidienne).

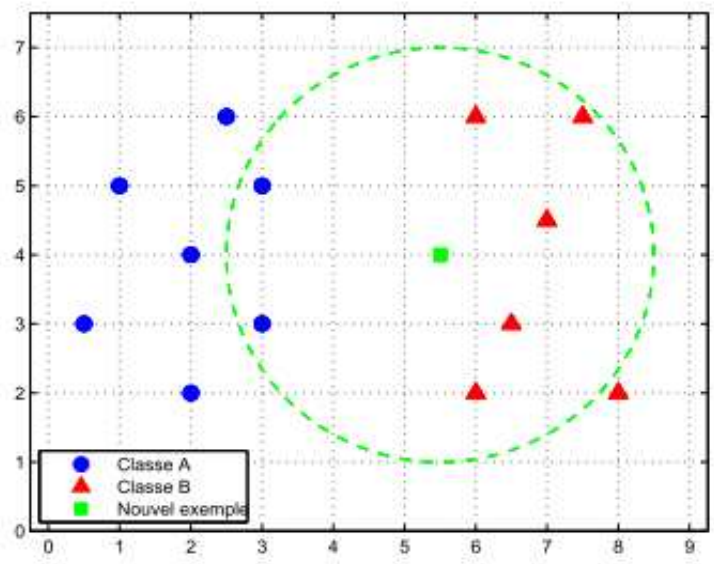


Figure I.9: Exemple simple de classification par un algorithme K-Plus Proches Voisins avec un calcul de distance par la norme euclidienne

Les distances entre le nouvel exemple et les différents points des deux classes sont calculées. Comme ses plus proches voisins appartiennent à la classe B, il est alors étiqueté comme appartenant à cette classe.

II.1.1.6)- Réseau de neurones :

Un réseau de neurones est un système de traitement de l'information qui a été développé comme généralisations des modèles mathématiques assortissant la connaissance humaine. Ils se composent d'un grand nombre d'unités de traitement hautement reliées appelées neurones, travaillant ensemble pour exécuter une tâche de classification donnée.

Il est un processeur parallèle distribué, ayant une prospérité naturelle pour stocker une connaissance expérimentale. Il ressemble au cerveau humain en trois aspects : la connaissance est acquise par le réseau par un processus d'apprentissage, des forces de connexion reliées ensemble, connues sous le nom de poids synaptiques, sont employées pour stocker la connaissance, chaque neurone a un état interne appelé seuil ou fonction d'activation (ou fonction de transfert) utilisée pour classifier les vecteurs.
Malheureusement, ils sont souvent difficiles à construire puisque leur structures (nombre de couches cachées et nombre de neurones par couche pour les perceptrons) influe beaucoup sur les résultats et il n'existe pas de méthode pour déterminer automatiquement cette structure.

On trouve plusieurs types de réseau de neurones : le VQ (apprentissage non-supervisé), le LVQ (apprentissage supervisé), les cartes auto organisatrices, le perceptron multicouches avec apprentissage par rétro propagation, les réseaux probabilistes, les réseaux à fonction radiales de base (FRB), et les réseaux récurrents.

Une classification par réseau de neurones comporte les étapes suivantes :
Tout d'abord une phase de prétraitement des images d'apprentissage et l'association à chaque image d'apprentissage (entrée de réseau) un vecteur de sortie, puis vient l'étape d'initialisation (création des couches du réseau). On fait l'apprentissage (supervisé) du réseau, jusqu'à atteindre une certaine erreur minimale (le réseau apprend à bien classifier les images d'apprentissage). On présente ensuite au réseau une nouvelle image à identifier (phase de reconnaissance ou de simulation ou d'activation du réseau) qui sera finalement affectée à une classe donnée **[11].**

Par exemple dans l'état de détection de visage :
Le réseau de neurones est utilisé pour classifier les pixels de l'image, en tant que visage ou non-visage. Dans toute utilisation de réseaux de neurones, il faut définir une topologie du réseau. La topologie de réseau est déterminée par des testes successifs et il n'y a aucune méthode standard à suivre pour définir la meilleure.
Un visage se distingue surtout par des yeux, un nez et une bouche. La topologie de base sera donc d'une unité finale fournissant une réponse binaire ou probabiliste.
On mettra derrière cette unité les couches cachées du réseau, on appelle notamment cela une topologie de base car le nombre d'unités, leur taille et leur position restent non empiriques et ne peuvent jamais être exactement fixés.
Les réseaux de neurones les plus répandus et les plus simples à la fois restent les perceptrons multicouches (PMC) qui consistent à une succession de 9 couches, interconnectées totalement ou partiellement.
L'algorithme d'apprentissage total reviendra à transmettre à tous les PMC, traitant chacun une unité, le résultat attendu. Si l'exemple à apprendre est un visage, la première étape est de transmettre aux PMC les yeux, la bouche et le nez. De là, chaque PMC applique son algorithme d'apprentissage.
L'inconvénient de cette approche réside dans le temps de calcul qui ne permet pas souvent de faire des traitements en temps réel. Le schéma suivant explique bien cette approche.

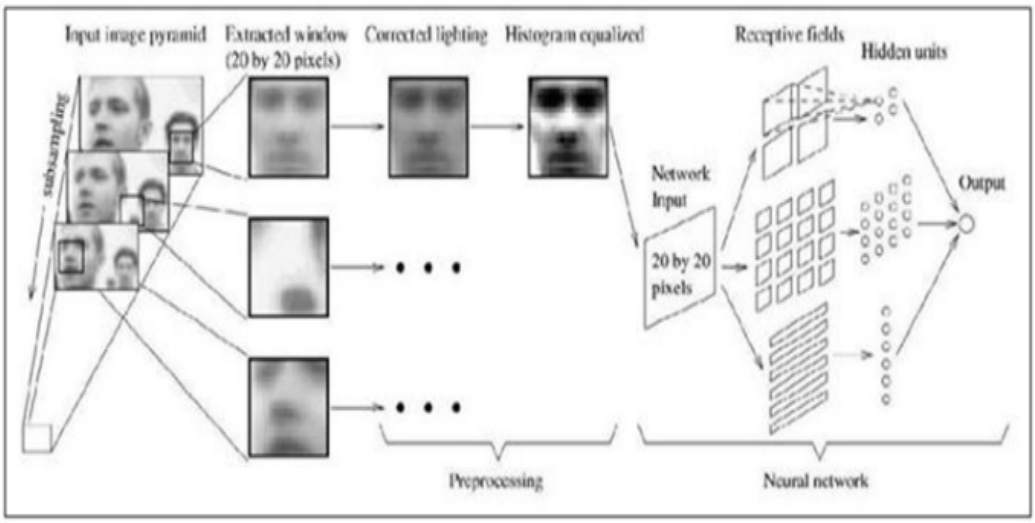


Figure I.10: Principe du réseau de neurone pour la détection des visages.

II.1.1.7)- Arbres de décision :

Le formalisme des arbres de décision permet de classifier un nouvel objet en testant ses caractéristiques les unes à la suite des autres. La classification se fait à travers une séquence de questions dans laquelle chaque question dépend de la réponse à la question précédente.

Cette séquence de questions est représentée par un arbre de décision dont les feuilles terminales représentent les classes.
La figure 3.2 illustre un exemple de classification sur des données continues en deux dimensions en utilisant les arbres de décision.
Dans la phase de construction de ce classificateur, les exemples de l'ensemble d'apprentissage sont divisés récursivement par des tests définis sur les caractéristiques pour obtenir des sous-ensembles d'exemples ne contenant que des exemples appartenant tous à une même classe. Les algorithmes existants (CART **[12]**, ID3 **[13],** C4.5**[14]** ...) différent essentiellement par leur façon de choisir, à une étape donnée, et parmi les caractéristiques disponibles, la caractéristique de segmentation et par le critère d'arrêt.

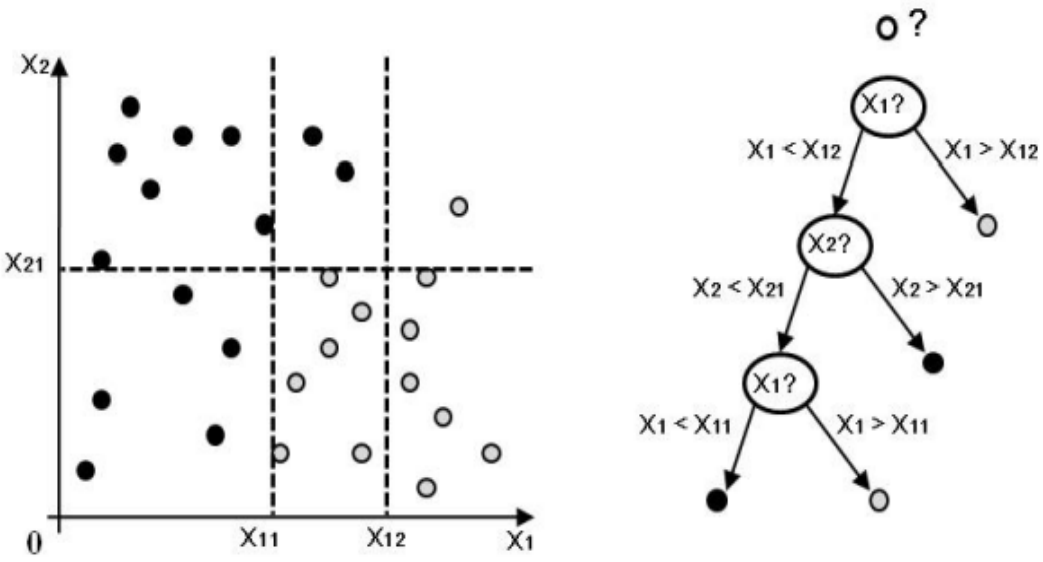


Figure I.11 : Exemple de classification avec les arbres de décision.

II.1.2)- Apprentissage non-supervisé :

Pour l'apprentissage non-supervisé, l'ensemble d'apprentissage est seulement composé d'exemples, sans aucune étiquette de classe. C'est à l'algorithme de trouver des dépendances, des structures entre les différents exemples. Le « clustering » ou partitionnement de données regroupe un ensemble de méthodes d'apprentissage non-supervisé, comme l'algorithme des K-means **[10]** ou l'Isodata ou bien C-means … etc **[15]**. Les classes (ou « clusters » en anglais) sont créées par l'algorithme qui regroupe dans une même classe des objets ayant des

caractéristiques communes entre elles et différentes avec les objets n'appartenant pas aux mêmes classes.
Le but des algorithmes de clustering est donc de minimiser la distance intra-classe (grappes d'éléments homogènes) et de maximiser la distance interclasse afin d'obtenir des sous-ensembles le plus distincts possible.
La mesure des distances est un élément prépondérant pour la qualité de l'algorithme de clustering.

II.1.2.1)- k-means:

K-means définit par McQueen **[16]** est un des plus simples algorithmes de classification automatique des données. L'idée principale et de choisir aléatoirement un ensemble de centres fixé a priori et de chercher itérativement la partition optimale.

Chaque individu est affecté au centre le plus proche, après l'affectation de toutes les données la moyenne de chaque groupe est calculé, elle constitue les nouveaux représentants des groupes, lorsqu'ont abouti à un état stationnaire (aucune donnée ne change de groupe) l'algorithme est arrêté.

Algorithme : K-means
Entrée
Ensemble de N données, noté par x
Nombre de groupes souhaité, noté par k
Sortie
1) Initialisation aléatoire des centres C_k;
Répéter
2) Affectation : générer une nouvelle partition en assignant chaque objet au groupe dont le centre est le plus proche ;

$$x_i \in C_k \; si \; \forall j |x_i - \mu_k| = \min_j |x_i - \mu_j| \qquad (1)$$

3) Représentation : Calculer les centres associe à la nouvelle partition ;

$$\mu_k = \frac{1}{N} \sum_{x_i \in C_k} x_i \qquad (2)$$

Jusqu'à convergence de l'algorithme vers une partition stable ;
Fin.

Prenons un exemple de classification par les K-means. L'algorithme est initialisé aléatoirement avec un certain nombre de clusters pour lesquels un point moyen, appelé centroïde, est évalué.
A chaque itération, la distance entre chaque exemple aux différents centroïde est calculée ; chaque exemple est alors associé au cluster dont la distance au centroïde est la plus proche.

Puis les centroïdes sont réévalués. L'algorithme se termine lorsqu'il n'y a plus aucun changement.
Un exemple simple de fonctionnement est montré sur la **figure12**.

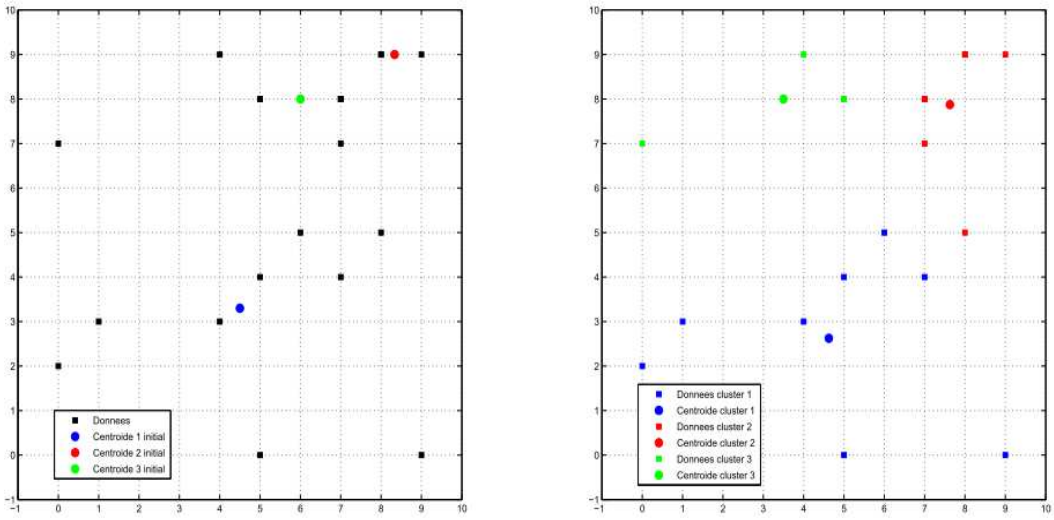


Figure I.12: Exemple simple de classification non-supervisé par un algorithme des K-means a- Données de départ et initialisation de 3 centroïdes b- Résultat final.

II.1.2.2)- L'algorithme génétique :

L'algorithme génétique (AG) est un algorithme de recherche basé sur les mécanismes de la sélection naturelle et de la génétique. Il combine une stratégie de "survie des plus forts" avec un échange d'information aléatoire mais structuré. Pour un problème pour lequel une solution est inconnue, un ensemble de solutions possibles est créé aléatoirement.

Les AGs sont des algorithmes itératifs basés sur la reproduction et l'évolution naturelle des individus en utilisant les principes de la survie des individus considérés comme les plus forts ou les mieux adaptés à l'environnement. Il s'agit alors de combiner les points forts de chaque individu pour en créer de nouveaux de manière à ce que leur efficacité soit meilleure.
Avec les AGs on cherche à optimiser une fonction (objectif) donnée dans un espace de recherche, celui des individus. Pour l'optimiser, on définit une fonction d'évaluation (fitness)n reliée à cette fonction objectif et appliquée sur chaque individu ou chromosome.
En général, le fonctionnement d'un AG est baée sur les phases suivantes (Figure 13) :
1. Initialisation : Générer aléatoirement une population initiale de taille N chromosomes.
2. Evaluation : Evaluer chaque individu de la population par la fonction d'évaluation appropriée au problème (fonction de fitness).
3. Reproduction : Créer une nouvelle population de N chromosomes par l'utilisation d'une méthode de sélection appropriée et l'application d'opérateurs génétiques (croisement et mutation) sur certains chromosomes au sein de la population courante.

4. Retour à la phase 2 tant que la condition d'arrêt du problème n'est pas satisfaite.

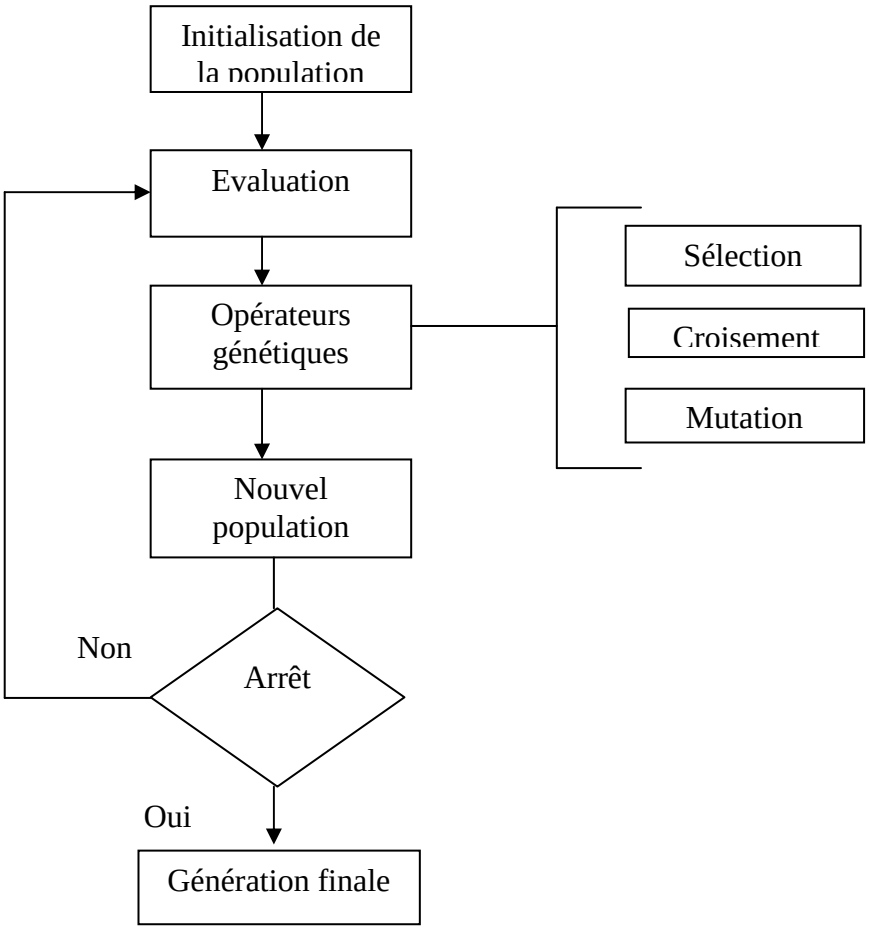


Figure I.13 : Architecture générale d'un algorithme génétique

Opérateurs génétiques :

La reproduction joue un rôle fondamental pour le passage d'une génération à une autre.
Elle représente un cycle de l'algorithme génétique. Elle se fait par l'application d'opérations génétiques : la sélection, le croisement et la mutation.

1)- Sélection :

La sélection consiste à choisir les individus à partir desquels on va créer, la génération suivante, les individus qui survivront, les parents qui par croisement génèreront des enfants, ou les individus qui subiront une mutation. La sélection des individus s'effectue le plus souvent sur la base de leur fonction d'évaluation. Plusieurs opérateurs de sélection existent parmi lesquels les méthodes par probabilité **[17]**, par rang de classement dans la population **[18]** ou par tournoi **[19]**, … etc.

2)- Croisement :

Le croisement est chargé de construire un individu (ou une solution) qui soit le mélange de plusieurs solutions. Dans le croisement, les chromosomes échangent des séquences de gènes entre eux.

Ce processus est appliqué à chaque paire de chromosomes sélectionnés selon une des méthodes décrites précédemment avec une certaine probabilité P_{crois}. Les paires de chromosomes sont recopiées sans modification dans la génération suivante avec la probabilité $1 - P_{crois}$. Plus P_{crois} est élevée, plus il y a de nouveaux individus qui apparaissent dans la population.
Parmi les méthodes de croisement les plus utilisées, le croisement à un point et le croisement multi-points.

3)- Mutation :

La mutation est définie comme étant la modification aléatoire de la valeur d'un gène dans un chromosome. La figure 14 illustre un exemple de mutation appliquée à la quatrième position d'un chromosome binaire. La mutation joue le rôle de bruit qui empêche l'évolution de se figer. Elle permet d'étendre l'exploration de l'espace et garantit que l'optimum global puisse être atteint. Cet opérateur évite donc une convergence vers les optima locaux.

Figure I.14 : Exemple d'une opération de mutation.

II.1.3)- Apprentissage semi-supervisé :

Ce terme regroupe des méthodes qui se situent entre l'apprentissage non-supervisé et l'apprentissage supervisé. Ce type de méthodes est utilisé quand un grand nombre de données est disponible mais sans qu'elles soient toutes étiquetées. L'initialisation de la méthode est faite à partir d'un petit jeu de données correctement étiquetées. Puis l'algorithme doit lui-même étiqueter les exemples suivants et construire son propre modèle.

Les algorithmes d'apprentissage non-supervisé et semi-supervisé sont beaucoup utilisés pour la recherche d'informations sur internet notamment. Ils permettent de traiter ainsi une grande quantité de données.

Plusieurs techniques d'apprentissage semi-supervisé existent parmi lesquels l'auto-apprentissage, Co-apprentissage, S3VM, T-SVM et le Co-training

II.1.3.1)- Auto-Apprentissage

1 L'auto-apprentissage (self-training) **[20]** consiste à entraîner un classifieur avec les données étiquetées (DL).

2 Le classifieur est, ensuite, utilisé pour étiqueter les données incomplètes (DU).
3 Les données étiquetées avec un haut degré de confiance sont ajoutées aux données d'apprentissage (DL).
4 Le classifieur est ré-entraîné sur les données de DL et la procédure est répétée jusqu'à satisfaire un critère d'arrêt.

II.1.3.2)- Séparateur Semi-Supervisé à Vaste Marge (S3VM) :

Dans cette approche, deux contraintes sont ajoutées au problème quadratique des SVM
Ces contraintes sont définies pour maintenir les données non-étiquetées à l'extérieur de la marge tout en minimisant l'erreur de classification supposée :

- $\frac{1}{2}\|w\|^2 + C\left(\sum_{i=1}^{l} \varepsilon_i + \sum_{j=l+1}^{N} min\left(\varepsilon_j^u, \varepsilon_j^{u^*}\right)\right)$
- $y_i(w^*.x_i + b) \geq 1 - \varepsilon_i$
- $\left(w^*.x_j + b\right) \geq 1 - \varepsilon_j^u$
- $\left(w^*.x_j + b\right) \leq -1 + \varepsilon_j^{u^*}$
- $\varepsilon_j^u / \varepsilon_j^{u^*}$ Coefficient de Lagrange lié à une erreur de classification dans le cas où l'échantillon est classé +1/-1**[20]**

II.1.3.3)- T-SVM :

L'idée du T-SVM est d'induire une fonction globale à l'aide des données annotées et des données de test **[20]**.
Le problème peut alors se poser de la manière suivante :

- $\frac{1}{2}\|w\|^2 + C \sum_{i=1}^{l} \varepsilon_i + C^* \sum_{j=l+1}^{N} \varepsilon_j^*$
- $y_j^*(w^*.x_j + b) \geq 1 - \varepsilon_j$
- $y_j^*(w^*.x_j + b) \geq 1 - \varepsilon_j^*$

Cette minimisation doit s'effectuer pour tous les cas de catégorisation possibles !

<u>Approche itérative pour réduire la complexité :</u>

L'idée de [Joa99a, Joa02] est de partir d'une fonction locale et d'étiqueter les données $\left(x_j, y_j^*\right)$ puis de généraliser petit à petit.

Pour ce faire, le paramètre C^* est progressivement incrémenté :

Dans le cas $C^* = 0$, l'apprentissage est purement inductif
Dans le cas $C^* = 1$, l'apprentissage est complètement transductif.
A chaque itération, on cherche à minimiser la fonction quadratique puis on réapprend le SVM en augmentant C^*.

Algorithme

- La fonction objectif est minimisée en cherchant un couple de données non étiquetées (x_m,x_l) se situant dans la marge (ou dans la zone où $\xi_m + \xi_l > 2$) et tels que $y_m^* \neq y_l^*$.

- Les étiquettes sont alors permutées pour replacer les données dans une zone plus acceptable, minimisant ($\xi_m + \xi_l$).

Si un couple d'étiquettes a été permuté, un nouveau SVM est appris et la procédure est répétée.
Si, par contre, aucun couple n'a été trouvé, C^* est augmenté et l'algorithme passe à la prochaine itération.

- Les itérations s'arrêtent lorsque C^* a atteint un seuil fixé a priori.

II.1.3.4)- Co-training

Le Co-training consiste à entraîner plusieurs classifieurs, chacun basé sur une vue du corpus, puis à les améliorer entre eux à l'aide d'une masse importante de données non annotées ; les classifieurs plus à même de classer un exemple donné jouant le rôle de "professeurs" pour les autres**[20].**

Soit un ensemble de données d'apprentissage A, de données de test T et de données non étiquetées U ; et un ensemble de paramètres (features) P.
Un exemple $e \in A$ est un couple e=(v, l)
Où v est un vecteur de dimension |P | représentant les différentes valeurs de chaque paramètre, et l est la classe de cet exemple.
On découpe U en N partitions de taille égale notées U_1, U_2, ...U_N.
On entraîne deux classifieurs C_1 et C_2 sur A. Un exemple e = (v, l) classé par un classifieur C_i donne C_i (e) = (l', s) où l':est la classe attribuée par C_i avec un score de confiance $s \in [0..1]$.
On définit un seuil $seuil_i \in [0..1]$ de confiance minimale pour chaque classifieur C_i.
On constitue deux ensembles $G_1 = G_2 = A$.

- Pour k allant de 1 à N
  - Chaque exemple $e = (v, l) \in U_k$ est classé par C_1 et C_2 :

C_1 (e) = (l_1, s_1) et C_2 (e) = (l_2, s_2)
Si $s_1 > seuil_1$, $G_2 := G_2 \cup \{(v, l_1)\}$
Si $s_2 > seuil_2$, $G_1 := G_1 \cup \{(v, l_2)\}$

- On entraîne C_i sur G_i
- On teste C_i sur T

PDF-XChange Viewer Click to buy NOW! www.docu-track.com

II.1.4)- Apprentissage par renforcement :

Ce type de méthode est un apprentissage interactif. A chaque décision que l'algorithme prend, il reçoit en retour des réponses de l'environnent appelées signaux de renforcement. C'est un processus adaptatif qui améliore la solution en fonction des réponses qu'il reçoit. L'algorithme de « Q-learning » **[21]** réalise un apprentissage par renforcement. Il produit une matrice Q dont chaque élément Q(s,a) est une mesure de l'intérêt d'effectuer l'action a lorsque le système se trouve dans l'état s. Par ailleurs, des résultats théoriques garantissent, dans des cas précis, la convergence de l'algorithme vers des valeurs optimales de Q. Ces algorithmes par renforcement permettent d'établir des processus plus complexes, comme le guidage de robots. Ce dernier a un objectif final à atteindre et en fonction des réponses de ses différents capteurs, il va pouvoir affiner ses actions.

Conclusion :

Dans ce chapitre, nous avons présenté le concept d'apprentissage automatique et de la classification ainsi qu'un état de l'art des méthodes et algorithmes usuels en apprentissage automatique (supervisé, non-supervisé,... etc).

Et dans le deuxième chapitre on présentera les concepts de la détection d'objets, en détaillant les différentes méthodes existantes sur la détection.

PDF-XChange Viewer
Click to buy NOW!
www.docu-track.com

Chapitre II :

Détection de la présence humaine

I)- Introduction :

Il est facile pour un humain de reconnaître un autre humain autour de lui ou sur une image bien que chaque individu soit différent et même unique, nous avons tous en commun notre morphologie. Une personne normalement constituée a deux bras, deux jambes, une tête, ...etc. Ce qui implique qu'une silhouette humaine pourra être différenciée d'une silhouette d'un animal. Mais c'est un problème très complexe pour un système automatisé. Pourtant, beaucoup de systèmes ont besoin d'avoir des informations sur la présence ou l'absence de personnes dans leur environnement. Les applications potentielles d'un système de détection automatique de la présence humaine sont nombreuses. Nous pouvons par exemple citer les systèmes de transport intelligent, la robotique, la surveillance . . .

Beaucoup de méthodes de détection existent déjà, comme les histogrammes de gradient orienté, particulièrement performants. Les méthodes les plus efficaces construisent des modèles statistiques par apprentissage supervisé, à partir des caractéristiques de forme ou d'apparence, calculées sur de nombreux exemples d'images de personnes.

Dans ce chapitre on présente quelques méthodes pour détecter la présence humaine dans une séquence vidéo. De telles méthodes peuvent être employées par des systèmes de navigation en sûreté des véhicules intelligents. Et nous détaillerons une approche précise, robuste et rapide de détection de personnes, l'algorithme de détection d'objet AdaBoost proposé par Viola Jones. Nous exposerons aussi l'estimation de distances et la calibration de la caméra. Et à la fin nous terminerons le chapitre par une conclusion.

II)- Détection d'objets :

La détection de l'objet est de déterminer si l'objet spécifié est présent ou pas, et c'est une méthode qui permet de détecter la présence d'une instance (reconnaissance d'objet) ou d'une classe d'objets dans une image numérique, et encore de déterminer les emplacements et les tailles de chaque objet.
Une attention particulière est portée à la détection de visage et la détection de personne. Ces méthodes font souvent appel à l'apprentissage supervisé et ont des applications dans de multiples domaines, tels la recherche d'image par le contenu ou la vidéo surveillance.

Des méthodes spécifiques ont été développées pour certains types d'objets, par exemple pour la détection de visage ou la détection de personne. Ces méthodes peuvent prendre en compte des caractéristiques spécifiques de l'objet comme par exemple le rapport largeur/hauteur, la présence des yeux et de la bouche dans le cas des visages, …etc.

La détection de la présence humaine est un cas particulier de 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.

II.1)- Détection de la présence humaine :

La détection humaine est la première étape pour un certain nombre d'applications telles que la surveillance vidéo intelligente, route des systèmes d'assistance et de gestion de contenu numérique intelligent. Détection d'objets est l'une des étapes de prétraitement les plus importants dans la reconnaissance d'objets et de systèmes d'identification. C'est un problème difficile en raison de la variance de l'éclairage, la couleur, l'échelle, la pose et ainsi de suite. Cela peut être fait par la recherche et l'indexation d'images fixes contenant objet dans divers taille, la position et le fond. Ce document passe en revue les différents aspects de la détection humaine dans des images statiques et de se concentrer sur méthode basée sur l'apprentissage qui s'appuie classificateurs.

II.1.1)- Les méthodes de détection de la présence humaine :

Il existe de nombreuses techniques de détection de piétons pour l'aide à la conduite. Depuis quelques années la détection des formes, des objets ainsi que celle des humains sont prises comme domaines de recherche par plusieurs personnes et sociétés ce qui amène à l'existence de plusieurs approches pour la détection des humains dans une image, parmi ces approches on cite les suivantes :

II.1.1.1)- Méthode de Dalal et Triggs (Histogrammes de gradients orientés) :

Les histogrammes de gradients orientés (Histograms of Oriented Gradients) ont été introduits par Dalal et Triggs**[22]** pour faire de la reconnaissance des formes sur des piétons dans des images fixes. Ils permettent de calculer les occurrences des orientations du gradient dans une portion localisée de l'image. Ce descripteur est aussi utilisé dans SIFT (Scale-Invariant Feature Transform) présenté par Lowe dans **[23]**. Ici, l'idée est d'en exploiter la qualité mais de façon épurée afin de l'utiliser pour des applications temps réel.

Les histogrammes de gradients se basent sur le calcul de gradient qui peut se faire en tout point d'une image. De façon générique, le gradient permet de calculer les variations d'une fonction par rapport aux variations de ses différents paramètres. En ce qui concerne le calcul de gradients dans les images de luminance, il s'agit de calculer la variation de l'intensité des pixels dans différentes directions. Cela revient à un calcul de gradient 1D dans les directions intéressantes (classiquement en horizontal, vertical ou diagonal). Un gradient 1D horizontal ou vertical est tout simplement un calcul de la dérivée partielle de la fonction image I(x,y) (voir figure 1).

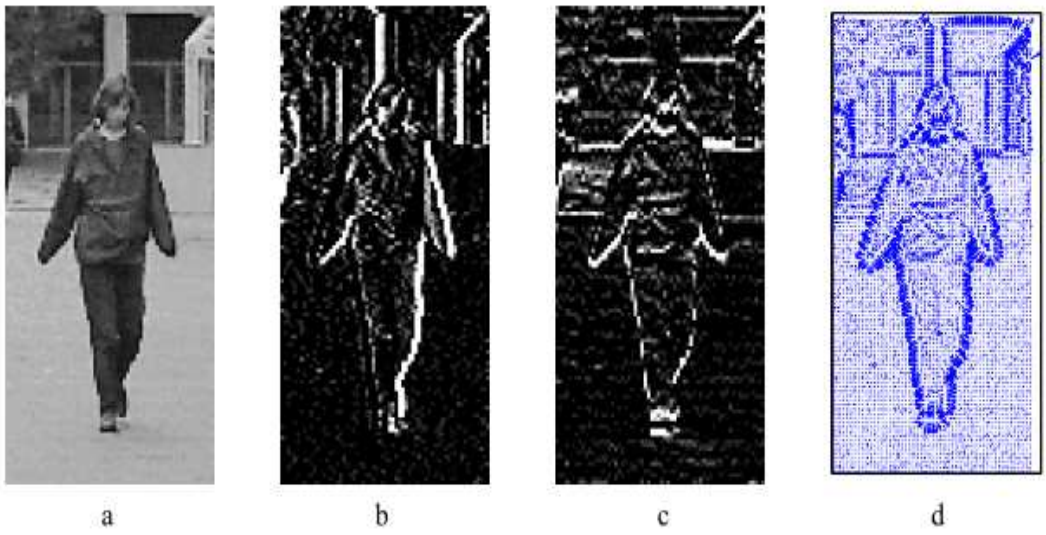


Figure II.1: Calcul du module et de l'orientation du gradient

a) image initiale.
b) calcul du gradient suivant x.
c) calcul du gradient suivant y.
d) orientation du gradient.

II.1.1.1.1)- Calcul des HOG:

Le calcul des histogrammes de gradient orientés se concentre sur trois étapes :

1)- calcul du gradient:

Il s'agit de calculer le gradient pour tous les points de l'image ; usuellement deux masques de dérivation sont appliqués sur l'image :

Un horizontal :=> $M_1 = [-1, 0, 1]$
Et un vertical :=> $M_2 = [-1, 0, 1]^T$

Pour chaque point de l'image, une approximation de la composante horizontale du gradient notée $\mathbf{G_x}$ et de la composante verticale notée $\mathbf{G_y}$ est ainsi obtenue. La plupart du temps la valeur absolue du gradient est prise car c'est le contraste entre deux régions qui importe le plus : un objet noir sur un fond blanc aura donc la même réponse qu'un objet blanc sur un fond noir.

2)- calcul de l'orientation :

Une fois que le calcul du gradient a été effectué pour tous les pixels de l'image, il faut en calculer l'orientation ; celle-ci peut être définie comme étant l'angle $\theta = \arctan\left(\frac{Gy}{Gx}\right)$. Classiquement $\theta \in [0, \Pi]$ (**[24]**).

3)- construction de l'histogramme :

L'image est divisée en plusieurs cellules. Pour chacune d'entre elles, un histogramme de gradients orientés est construit en comptabilisant les occurrences du gradient dans une barre correspondant à un intervalle d'orientation spécifique. Un exemple est montré sur la figure 2.

Différents types de masques ont été testés dans **[22]**. Il existe les R-HOG (pour Rectangular-HOG) des C-HOG (pour Circular-HOG). Pour les R-HOG, les gradients autour du pixel sont comptabilisés avec une fenêtre rectangulaire ; les C-HOG correspondent à l'utilisation d'une fenêtre circulaire autour du point considéré pour le calcul du gradient. Cette approche est toutefois peu utilisée pour des applications de reconnaissances d'objets car il requiert un temps de calcul plus conséquent.

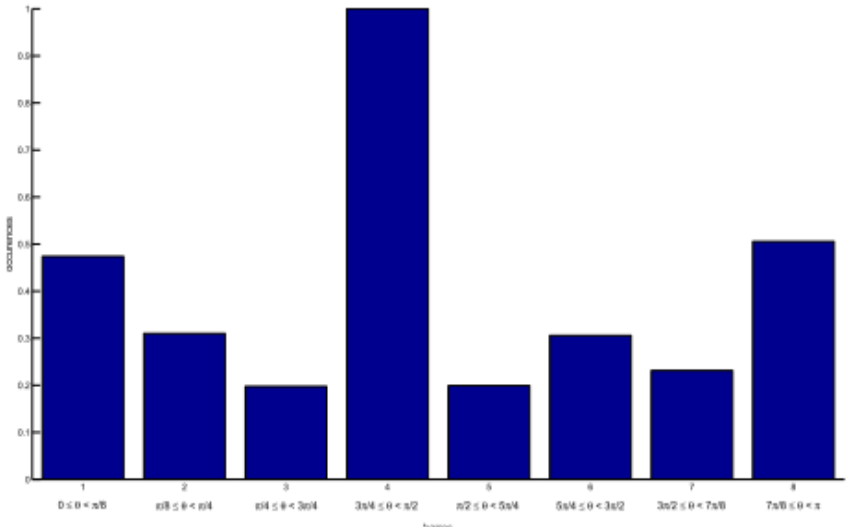


Figure II.2: Exemple d'un histogramme de gradients à 8 barres correspondant à la figure 1.

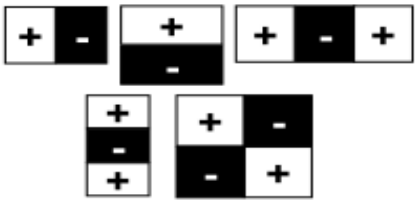
II.1.1.2)- la méthode de Viola & Jones (filtres de Haar):

La méthode proposée par Paul Viola et Michael Jones en 2003 dans leur papier, "Robuste Découverte du Visage du temps réel" était en avant un pas considérable dans le champ de la découverte du visage.
Semblable aux autres méthodes antérieures, ils ont utilisé des algorithmes de la machine-érudition pour sélectionner un ensemble de caractéristiques de Haar simples qu'ils ont combiné dans un classifieur mesurable effectif.
Cependant, leur papier a introduit trois innovations de la clef à qui ont permis leur détecteur accomplissez des augmentations de la performance sur systèmes antérieurs. Le premier était l'usage d'une nouvelle image "image intégrante" pour calcul du trait plus rapide. La seconde était l'usage de l'AdaBoost algorithme de type Boosting largement utilisé comme un moyen pour sélectionner un classifieurs simples et effectif. Le dernier était une méthode de combiner des classifieurs dans un "cascade" que rapidement éliminent des régions de l'origine et concentre compulsionnel sur régions plus prometteuses de l'image.

II.1.1.2.1)- les caractéristiques de Haar

C'est un détecteur fondé sur des caractéristiques plus globales de l'objet **[25]** et sont des fonctions permettant de connaître la différence de contraste entre plusieurs régions rectangulaires contiguës dans une image et de l'utiliser en tant que descripteur d'images pour la reconnaissance d'objets.

En effet, Ces descripteurs sont calculés dans une fenêtre de taille fixe (ex. 24x24 pixels) et permettent de calculer la différence entre la somme des pixels dans les zones blanches et la somme des zones noires et généralement, ils sont classifiés en 3 sortes : 2 rectangles, 3 rectangles et 4 rectangles descripteurs et pour les régions blanches ont des poids positifs et les régions noires ont des poids négatifs :



Et la valeur de descripteur est calculée par :

$$f_i = \text{Sum}(r_{i,\text{blanche}}) - \text{Sum}(r_{i,\text{noire}})$$

On code ainsi les contrastes existants pour piéton et les relations spatiales sont comme suite :

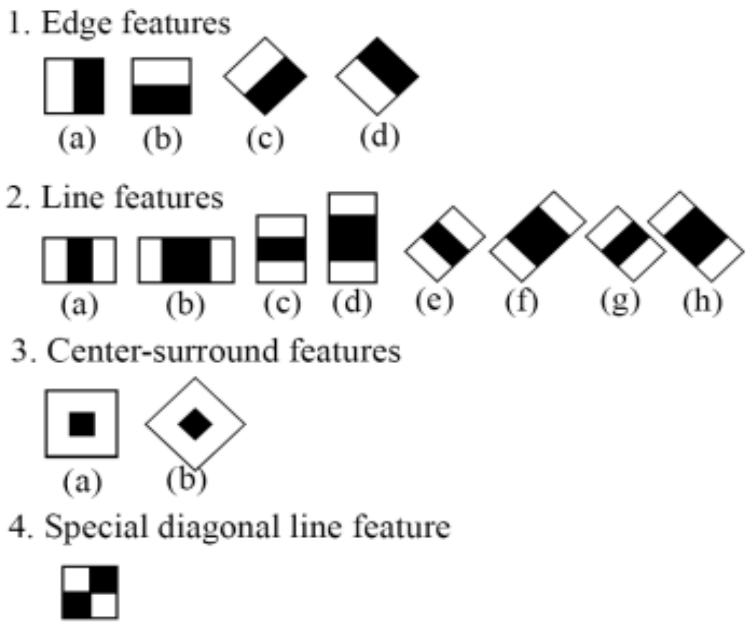


Figure 3 : Descripteurs de Haar.

Un descripteur de Haar est caractérisé par:

- le nombre de rectangles (2,3 ou 4)
- la position (le sommet supérieur gauche) (x,y) de chaque rectangle

- la largeur w et la hauteur h de chaque rectangle avec 0<x,x+w<W ; 0<y,y+h< H
- les poids positifs ou négatifs de chaque rectangle

II.1.1.2.2)- L'image integrale

L'image intégrale est une nouvelle représentation qui va permettre de calculer plus rapidement les attributs du descripteur. L'idée est de calculer seulement une fois la somme de tous les pixels de l'image **[2]**. Le pixel à la position (u, v) de l'image intégrale contient la somme de tous les pixels, de l'image initiale, supérieurs et à gauche de la position (u, v)

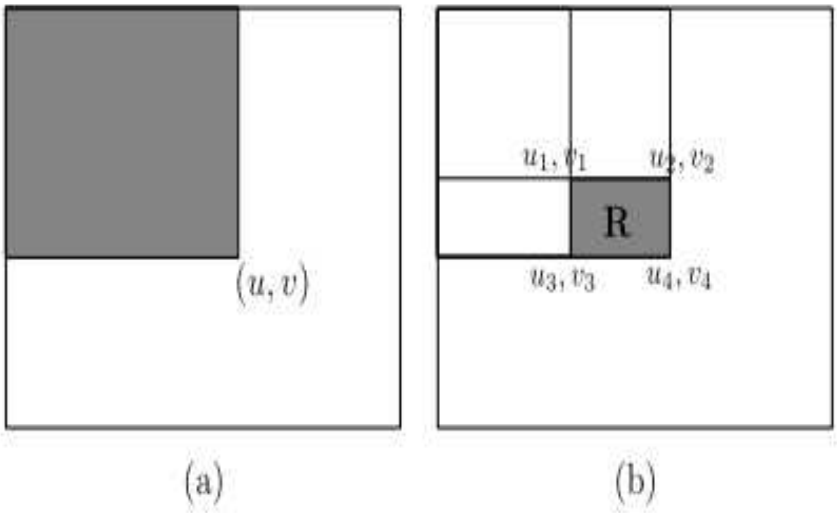


Figure II.3 : Image intégrale et calcul de la somme des niveaux de gris dans un rectangle.

(a)La valeur de l'image intégrale à la position (u, v), **(b)** quatre références à l'Image Intégrale permet le calcul de la somme des niveaux de gris dans R.
L'image intégrale est une représentation intermédiaire de l'image d'entrée permettant de réduire le temps de calcul des filtres. Soit une image en niveaux de gris i(u, v) (où (u, v) sont les coordonnées dans l'image) l'image intégrale est définie comme :

$$ii(u,v) = \sum_{\substack{u'<u \\ v'<v}} i(u',v')$$

La somme des niveaux de gris des pixels dans une région rectangulaire de i peut être calculée rapidement à partir des 4 références à l'image intégrale (figure 3):

$$R = \big(ii(u_4,v_4) + ii(u_1,v_1)\big) - \big(ii(u_2,v_2) + ii(u_3,v_3)\big)$$

Par conséquent, la différence entre deux rectangles adjacents est obtenue à travers six références à l'image intégrale ii(u, v). Pour un filtre à trois rectangles, on a besoin de huit références.

II.1.1.2.3)- L'algorithme d'Adaboost :

Comme nous l'avons présenté dans le chapitre précédent.

L'apprentissage consiste à analyser un très grand nombre d'images positives et négatives.

L'apprentissage d'un modèle perceptron consiste à déterminer automatiquement ses paramètres à partir d'une base étiquetée d'exemples d'apprentissage. L'algorithme est le suivant (pour simplifier on considère que W est un vecteur ligne et que les x sont des vecteurs colonnes) :

- D=(x_i ,y_i) i=1..N ensemble d'apprentissage, où y_i =C(x_i)
- initialiser w, on note ce vecteur initial w(0)
- t=0
- à t, choisir aléatoirement un exemple x dans D, on le nomme x_t

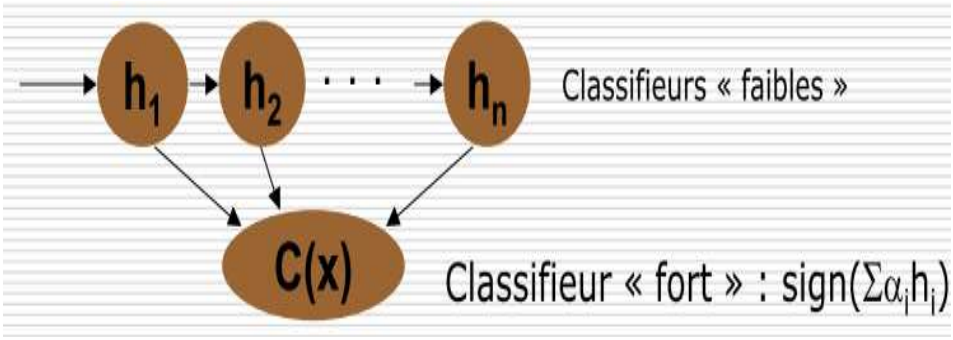
- **si** x(t) est correctement classé, i.e y_t (w(t).x_t)>0 **alors**
  w(t+1)=w(t)
- **si** x(t) est mal classé, i.e. y_t (w(t).x_t) < 0 **alors**
  w(t+1)=w(t)+ ε y_t x_t
  t=t+1

- jusqu'à (critère d'arrêt satisfait).

Où ε est le pas de gradient, c'est une valeur positive non nulle, en général relativement faible (par exemple 1/N). On utilise différents critères d'arrêts (plus d'erreur de classification depuis un certain temps, t a atteint une valeur maximale, …etc.)

Adaboost est une méthode d'apprentissage permettant de "booster" les performances d'un classifieur quelconque nommé "classifieur faible". L'idée est de faire passer les candidats à classifier à travers plusieurs classifieurs faibles, chacun étant entraîné en portant plus d'attention sur les candidats mal classifiés par le classifieur précédent.

Donc l'Algorithme adaboost construit un classifieur «fort» C(x) en sélectionnant et en combinant les classifieurs «faibles» h_n(x) les plus performants



Critère de sélection:
$h_t = arg_j min\ E_j$
E_j= l'erreur de classification
h_t= classifieur sélectionné

II.1.1.2.4)- Le cascade :

Le troisième apport du travail de Viola et Jones **[25]** consiste en l'introduction d'un apprentissage basé sur la méthode de dopage (Boosting) qui permet d'obtenir une fonction de classification avec une architecture en cascade.
Pour réaliser un traitement efficace, il est nécessaire d'avoir l'avis de plusieurs classifieurs forts.
Une cascade de classifieurs est un arbre de décision dégénéré dans laquelle, chaque étape est entraînée pour détecter un maximum d'objets intéressants tout en rejetant une certaine fraction des objets non-intéressants.
La structure de la cascade reflète le fait que l'image est constituée majoritairement de sous-fenêtres négatives. Une sous-image doit passer tous les classifieurs afin d'être acceptée comme visage. Le déclenchement de tous les classifieurs par un résultat positif devient ainsi un événement rare. Il faut que le nombre de sous-images éliminées dès les premières étapes de la cascade soit très élevé.
La cascade des classifieurs commence par un taux de détection de presque 100% mais avec un taux de faux positifs (fenêtres négatives déclarées comme positives) élevé. Il diminue rapidement avec quelques itérations. Ainsi on rejette immédiatement un grand nombre de régions négatives, donc cela accélère l'apprentissage et la détection.
La présence d'un visage dans les images est très rare et la cascade complète est ainsi très rarement appliquée à une sous-image. La grande majorité des sous-images sont rejetées en échouant dès les premières étapes.

La cascade est la combinaison successive des classifieurs forts, dont la complexité est croissante tout au long de la structure (voir figure). Au départ, les classifieurs simples, éliminent la plupart des fausses alarmes, avant que les classifieurs plus complexes 1 soient utilisés pour éliminer des fenêtres plus difficiles. Ainsi, l'attention du système se concentre, au fur et à mesure des étages de la cascade, sur des zones de plus en plus réduites et pertinentes.

Cette façon d'organiser la détection permet d'incrémenter la performance totale du système (taux de détections correctes et taux de fausses alarmes), en même temps qu'elle augmente la vitesse en se focalisant sur des régions susceptibles de contenir un véhicule.

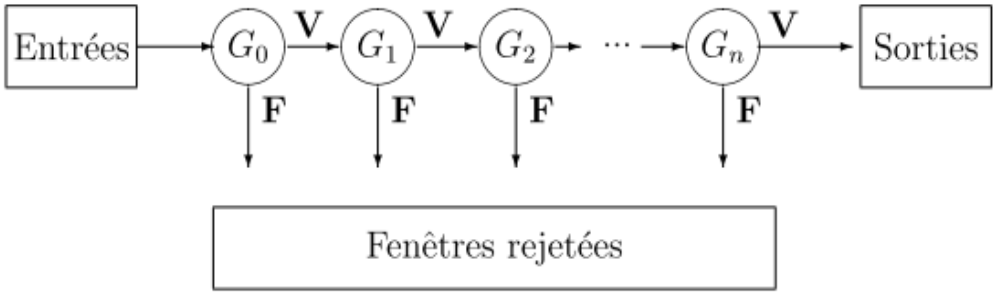


Figure II.4: Cascade de classifieurs forts.

Cela peut se démontrer intuitivement si nous prenons un premier classifieur G_0 composé d'un faible nombre des fonctions g_t (trois, quatre ou cinq au total) et l'appliquons sur plusieurs régions d'une image. Il est entraîné pour éliminer la moitié des régions négatives.
Les régions qui ont subsisté subissent l'application du deuxième classifieur G_1, composé lui aussi d'un faible nombre de fonctions faibles. G_1 est spécialisé dans le traitement des échantillons plus proches de la classe véhicule, qui ont été validées par G_0, et élimine, à son tour, la moitié des fausses alarmes.
Nous pouvons considérer que les régions restantes ont de plus forte probabilité, encore, d'appartenir à la classe véhicule. Pour éliminer la moitié des fausses alarmes, les prochains classifieurs seront composés d'un nombre de plus en plus important de g_t, avec pour conséquence l'augmentation du temps de calcul par région. Mais, étant donné que les étapes antérieures ont éliminé successivement un grand nombre des régions négatives, cette augmentation n'a pas d'incidence, du fait du très faible nombre de régions qui restent à évaluer.

III)- Estimation de la distance :

On parle ici de la distance entre le véhicule et le piéton

La **distance d'arrêt** est la distance parcourue par un véhicule entre le moment ou son conducteur perçoit un problème, et le moment où le véhicule est à l'arrêt complet.

La **distance de réaction** est la distance que parcourt le véhicule entre le moment où son conducteur perçoit un problème, et le moment où il pose son pied sur la pédale de frein.

La **distance de freinage** est la distance parcourue par le véhicule entre le moment où son conducteur commence à appuyer sur la pédale de frein, et le moment où le véhicule est à l'arrêt complet.

III.1)- La distance de réaction :

Il s'agit de la *distance parcourue pendant le temps de réaction.*

Le temps de réaction :

C'est le temps qui s'écoule entre le moment où le conducteur voit les feux rouges du véhicule devant lui s'allumer, et le moment où effectivement, son pied sera posé sur la pédale de frein (mais avant qu'il commence à appuyer).
Pour parler clairement, c'est le temps qui s'écoule pendant lequel le cerveau du conducteur détecte le problème, l'analyse, choisit une réponse, envoie l'ordre d'éxecution de cette réponse (ici par exemple, de freiner, c'est-à-dire pour l'impulsion de parcourir dans les nerfs, la distance entre le cerveau et la jambe).

Enfin il faut encore ajouter le temps pour le pied de lacher la pédale d'accélérateur pour aller se placer sur la pédale de frein. Tous ces « micros-temps » mis bout-à-bout représentent selon les scientifiques environ une seconde, pouvant passer à deux secondes simplement à cause de la fatigue d'une journée de travail, ou d'un rhume !

Ce concept de temps de réaction est souvent totalement sous-estimé voire oublié par beaucoup.
Pour vous donner un ordre de comparaison, Usain Bolt et ses petits camarades de jeu, mettent en gros 0,3 seconde pour « réagir » au coup de feu du starter sachant que :

- c'est une situation simple, connue et répétée de multiples fois à l'entrainement : bang = départ;
- toute l'attention et la concentration des coureurs est focalisée sur cet instant;
- ce sont des athlètes surentraînés… et en forme.

III.2)- La distance de freinage:

On parle donc de la distance que va parcourir le véhicule entre le moment où le conducteur commence à appuyer sur la pédale de frein et le moment où le véhicule va finalement s'arrêter. L'état du conducteur est ici bien moins déterminant que les intempéries, l'état de ses pneus ou de la chaussée. La diversité des paramètres pris en compte ne permet pas un calcul simple de cette valeur, autrement qu'en « manipulant » l'équation de base :

- Distance d'arrêt = distance de réaction + distance de freinage

Donc :

Distance de freinage = distance d'arrêt – distance de réaction

III.3)- La distance d'arrêt :

La *distance d'arrêt* – sur sol sec – s'obtient en multipliant le chiffre des dizaines de la vitesse (nous avons déjà vu ce principe dans l'article sur la distance de sécurité) par lui-même.

Ainsi, à 50 km/h entre la perception du problème et l'arrêt du véhicule, la distance parcourue aura été de 5(0) x 5(0) = 25 m
ou encore à 110 km/h, 11 x 11 = 121 m.

Le calcul se corse (un peu) si on considère les conditions d'adhérence d'une route mouillée. Il faut alors reprendre le calcul initial, puis rajouter la moitié du résultat obtenu.

IV)- Calibration de la caméra :

Nous savons quelle est la zone de travail où chercher des piétons en 3D. A partir du positionnement de la caméra et de ses paramètres de calibration, il est possible de déterminer sa position 2D dans les images. La caméra est représentée par un modèle sténopé comme montré sur la figure 6.

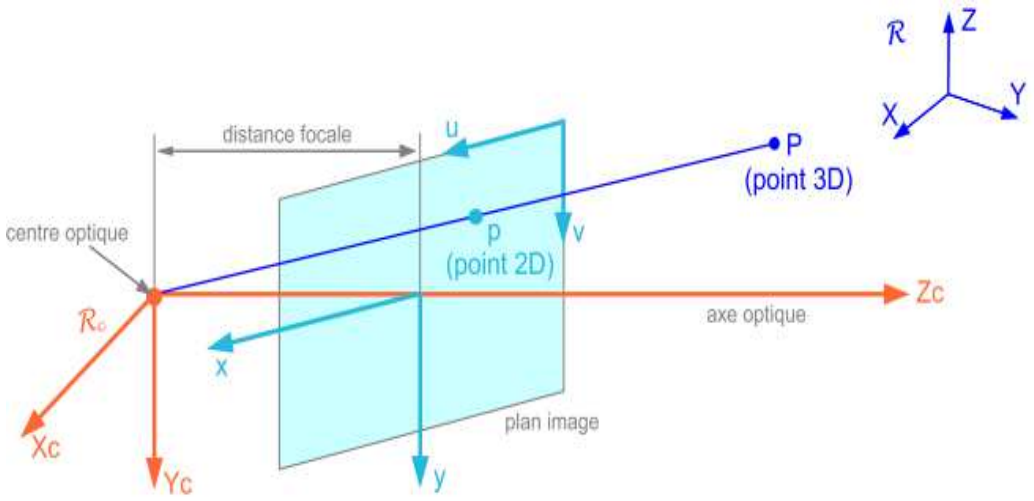


Figure II.5: Représentation d'une caméra par un modèle sténopé

Passage d'un point réel 3D « P » vers le point image 2D « p ».

A cause les difficultés que nous avons par rapport à la calibration de la caméra, nous avons optés la fonction d'interpolation polynomiale pour calculer la distance réelle en fonction de la distance en pixels.

V)- La fonction polynomiale :

Soit ***f*** une fonction numérique de classe ***n*** avec ***n*** ϵ N^* (c'est-à-dire que les n premières dérivées sont continues et la $(n+1)^{ème}$ dérivée existe).
Et soit P_n un polynôme de degré ***n***.
On dit que P_n interpole f aux points : (x_0, y_0) (x_1, y_1) $(x_2, y_2)\ldots(x_n, y_n)$
si et seulement si $f(x_i)=P_n(x_i)=y_i \quad i=0..n.$

L'interpolation polynomiale sert à approcher une fonction inconnue (ou méconnue le cas échéant) pour un polynôme simple à calculer et à implémenter.

Méthode de Lagrange :

On définit le polynôme de Lagrange (d'ordre i) par :

$L_i(x) = \prod_{\substack{j=0 \\ i\neq j}}^{n} \frac{x-x_j}{x_i-x_j}$ pour $i=0\ldots n.$

On montre que ces polynômes forment une base dans l'espace des polynômes de degré n et par conséquent tout polynôme peut s'écrire comme une comme une combinaison linéaire de ces derniers. En particulier, le polynôme d'interpolation qui est donné par :

$$P_n(x) = \sum_{i=0}^{n} y_i L_i(x)$$

Exemple d'application :

Soit les vecteurs d'observation : $X = [-1 \;\; 0 \;\; 1 \;\; 3]^T$ représentant l'entrée d'un système donné et $Y = [-2 \;\; -1 \;\; 2 \;\; 14]^T$ représentant la sortie. Le système est donné par le polynôme $g(x) = x^2 + 2x - 1$.

On cherche l'image de l'entrée $x=4$ par le polynôme d'interpolation est $n=3$

$$L_0(x) = \frac{4-0}{-1-0}\cdot\frac{4-1}{-1-1}\cdot\frac{4-3}{-1-3} = -4\cdot\frac{3}{-2}\cdot\frac{1}{-4} = \frac{-3}{2}$$

$$L_1(x) = \frac{4-(-1)}{0-(-1)}\cdot\frac{4-1}{0-1}\cdot\frac{4-3}{0-3} = 5\cdot(-3)\cdot\frac{1}{-3} = 5$$

$$L_2(x) = \frac{4-(-1)}{1-(-1)}\cdot\frac{4-0}{1-0}\cdot\frac{4-3}{1-3} = \frac{5}{2}\cdot 4\cdot\frac{1}{-2} = -5$$

$$L_3(x) = \frac{4-(-1)}{3-(-1)}\cdot\frac{4-0}{3-0}\cdot\frac{4-1}{3-1} = \frac{5}{4}\cdot\frac{4}{3}\cdot\frac{3}{2} = \frac{5}{2}$$

Ainsi,

$P_3(4) = -2 \cdot \frac{-3}{2} + (-1) \cdot 5 + 2 \cdot (-5) + 14 \cdot \frac{5}{2} = 23.$

Vérification : $g(4) = 4^2 + 2 \cdot 4 - 1 = 23 = P_3(4)$.

VI)- Conclusion :

Nous avons présenté dans ce présent chapitre quelques méthodes de détection de la présence humaine dans des séquences d'images, ainsi l'estimation de la distance et enfin la fonction d'interpolation polynomiale.

Dans le chapitre suivant nous passons à l'analyse et la conception où nous schématisons notre travaille de programmation.

PDF-XChange Viewer
Click to buy NOW!
www.docu-track.com

Chapitre III :

Implémentation et réalisation

I)- Introduction :

Dans ce chapitre nous allons présenter en premier lieu l'environnement de développement de notre application, ainsi le langage de programmation utilisé, en donnant les raisons pour les quelles on a choisi le langage de programmation c++, puis on va présenter OpenCV, ensuite on va évaluer

II)- Environnement de développement

Nous avons choisi pour la réalisation de notre travail le Microsoft visual studio et le langage c++ comme environnement de développement de l'application.

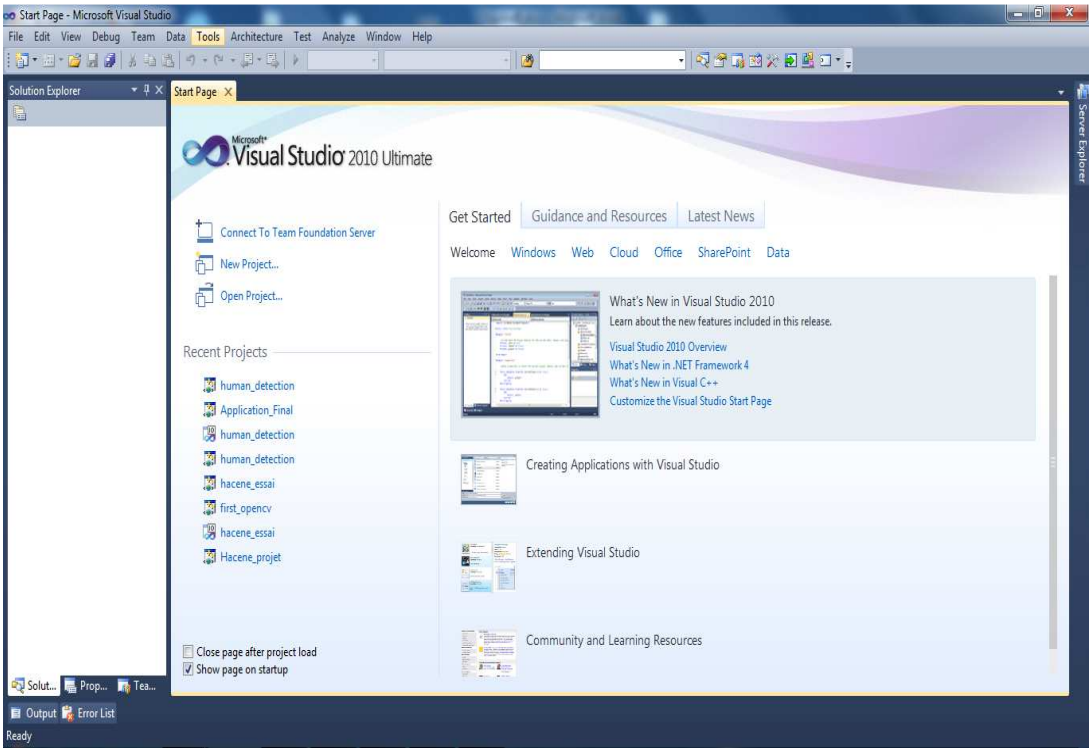


Figure III.1: interface de l'environnement de développement Visual studio c++

II.1)- Microsoft Visual Studio 2010:

Visual Studio est une suite de logiciels et d'outils de développement permettant aux développeurs de faciliter la manipulation des données en établissant la liaison entre les données et l'application et de générer des applications Web ASP.NET, des Services Web XML, des applications bureautiques et des applications mobiles. Visual Basic, Visual C++, Visual C# et Visual J# utilisent tous le même environnement de développement intégré (IDE, Integrated Development Environment), qui leur permet de partager des outils et facilite la création de solutions faisant appel à plusieurs langages. Par ailleurs, ces langages permettent de mieux tirer parti des fonctionnalités du Framework .NET.

II.2)- Matlab :

Est un langage de programmation de quatrième génération et un environnement de développement, il est utilisé à des fins de calcul numérique. Développé par la société The MathWorks, MATLAB permet de manipuler des matrices, d'afficher des courbes et des données, de mettre en œuvre des algorithmes, de créer des interfaces utilisateurs, et peut s'interfacer avec d'autres langages comme le C, C++, Java, et Fortran. Les utilisateurs de MATLAB (environ un million en 2004) sont de milieux très différents comme l'ingénierie, les sciences et l'économie dans un contexte aussi bien industriel que pour la recherche.

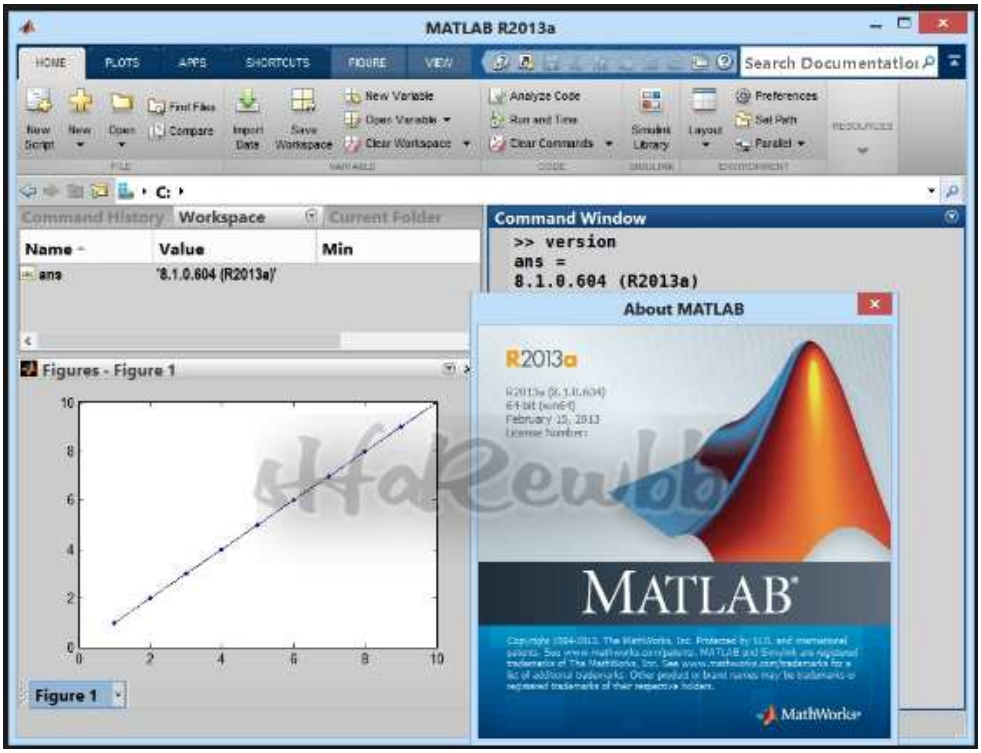


Figure III.2 : interface de l'environnement de développement Matlab

II.3)- Description du langage c++ :

Le langage C++ a été développé par « Bjarne Stroustrup » au cours des années 1980, alors qu'il travaillait dans le laboratoire de recherche Bell d'AT&T. Il s'agissait en l'occurrence d'améliorer le langage C. Il l'avait d'ailleurs nommé *C with classes* (C avec des classes). Les premières améliorations se concrétisèrent donc par la prise en charge des classes, ainsi que par de nombreuses autres fonctionnalités, En 1989, c'est la sortie de la version 2.0 de C++. Parmi les nouvelles fonctionnalités, il y avait l'héritage multiple, les classes abstraites, les fonctions membres statiques, les fonctions membres constantes, etc….

Comme le langage C++ évoluait, la bibliothèque standard évoluait de concert. La première addition à la bibliothèque standard de C++ concernait les flux d'entrées/sorties qui apportaient les fonctionnalités nécessaires au remplacement des fonctions C traditionnelles telles que

printf et *scanf*. Ensuite, parmi les additions les plus importantes, il y avait la Standard Template Library.
En langage C, ++ est l'opérateur d'incrémentation, c'est-à-dire l'augmentation de la valeur d'une variable de 1. C'est pourquoi C++ porte ce nom : cela signifie que C++ est un niveau au-dessus du C.

II.4)- OpenCV :

Nous avons utilisé dans notre application OpenCV (Open Source Computer Vision Library) qui est une librairie de traitement d'images et de vision par ordinateur en langage C/C++, proposée par Intel pour Windows et Linux. Cette bibliothèque propose un grand nombre d'opérateurs « classique » : création, lecture et écriture d'images, accès aux pixels, traitement d'images, apprentissage, détection de visages, suivi d'objet vidéo, etc.

Avec OpenCV, nous pouvons entraîner une cascade en format xml pour la détection d'humains

Un des buts d'OpenCV est d'aider les gens à construire rapidement des applications sophistiquées de vision à l'aide d'infrastructure simple de vision par ordinateur. La bibliothèque d'OpenCV contient près de 500 fonctions.
Il est possible grâce a la « licence de code ouvert » de réaliser un produit commercial en utilisant tout ou partie d'OpenCV. Il n'est pas obligatoire de montrer le code du produit et les améliorations réalisées au domaine public.

III)- Les fonctions utilisées.

– cvCaptureFromCAM :
C'est une fonction prédéfinie qui permet de capturée l'image.

– cvQueryFrame :
C'est une fonction prédéfinie qui Permet de faire implémente une image dans une matrice.

– cvtColor(frame, frame_gray, CV_BGR2GRAY) :
C'est une fonction prédéfinie elle transforme la matrice frame (image) en une matrice frame_gray (image) intensité de gris.

– ellipse(Image, Centre, Rayon, Couleur, Epaisseur, Type ligne, Décalage) :
C'est une fonction prédéfinie, La structure de cette fonction est définie dans le programme implémenté et elle ne s'exécute qu'après avoir déterminé ses paramètres qui se déduisent à partir du traitement effectué sur les images car les rayons des cercles sont proportionnels aux tailles du corps humain trouvés.

– DetecterEtAfficher(Mat frame):

Elle correspond à la fonction principale du programme implémenté, permet d'effectuer tout les traitements nécessaires pour la détection des humains comme :(conversion de la couleur en nivaux de gris).

–Fullbody_cascade.detectMultiScale() :

C'est la ou sa passe le traitement de détection d'un humain.

– putText() :

C'est une fonction prédéfinie qui permet d'afficher un texte sur l'image.

Pour trouver la distance en mètres, on la calcule avec la formule suivante :

Dist=p.cal (480-(body[i].y+body[i].height)) ;

La formule ***(480-(body[i].y+body[i].height)*** permet de retourner la distance de la position de l'humain en pixels dans l'image.

– p.cal () :

Calcule la distance réelle (en mètre) en fonction de la distance en pixels, pour cela nous avons utilisé la fonction polynomiale de Lagrange p, qui consiste a supposé quelques points en valeurs pixels sur notre image pour les calculer en mètres dans la réalité et ensuite généralisé notre fonction polynomiale

III.1)- Implémentation de la fonction polynomiale

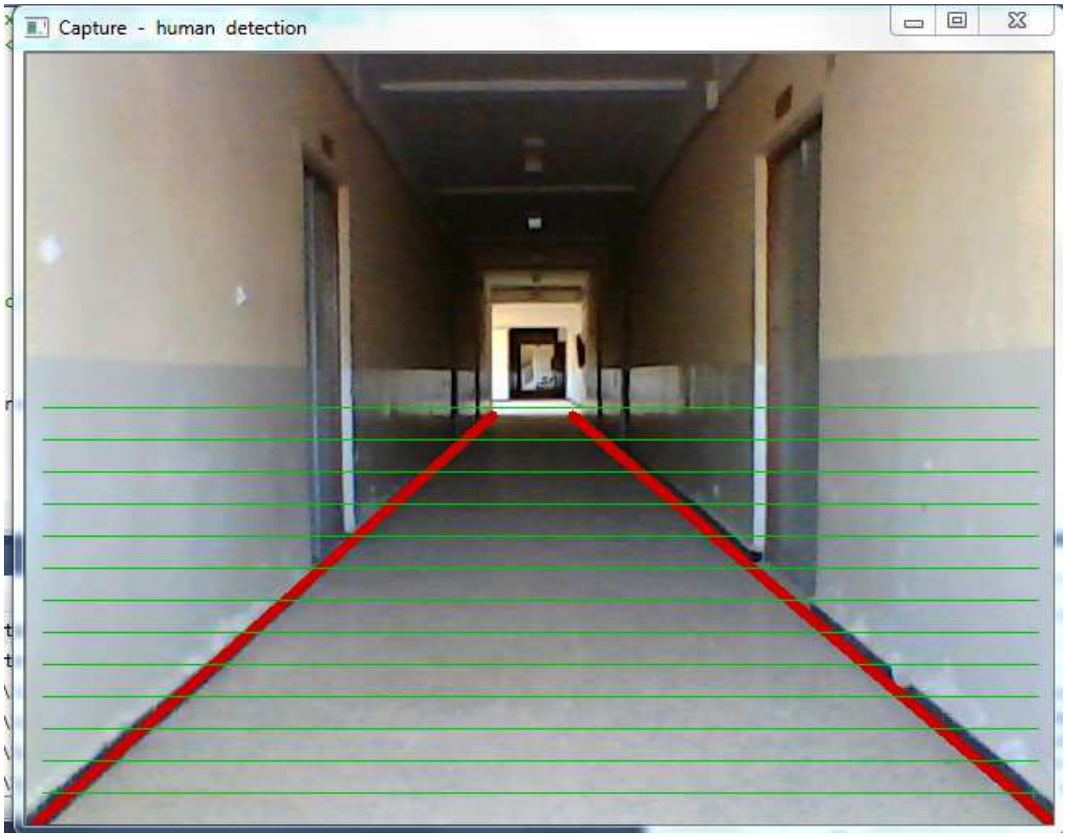


Figure III.3: image correspondante aux distances supposées en pixels.

Nous avons trace des lignes qui ce suivi a chaque fois avec 20 pixels et en calculant en même temps la distance en réel pour chaque point nous avons obtenu ces résultats comme suit :

Distance en Pixels	Distance en mètres
0	3.25
20	3.625
40	3.94
60	4.44
80	5.06
100	5.94
120	7
140	8.75
160	11.5
180	18.5
200	36

Tableau1 : Résultats des valeurs calculées manuellement

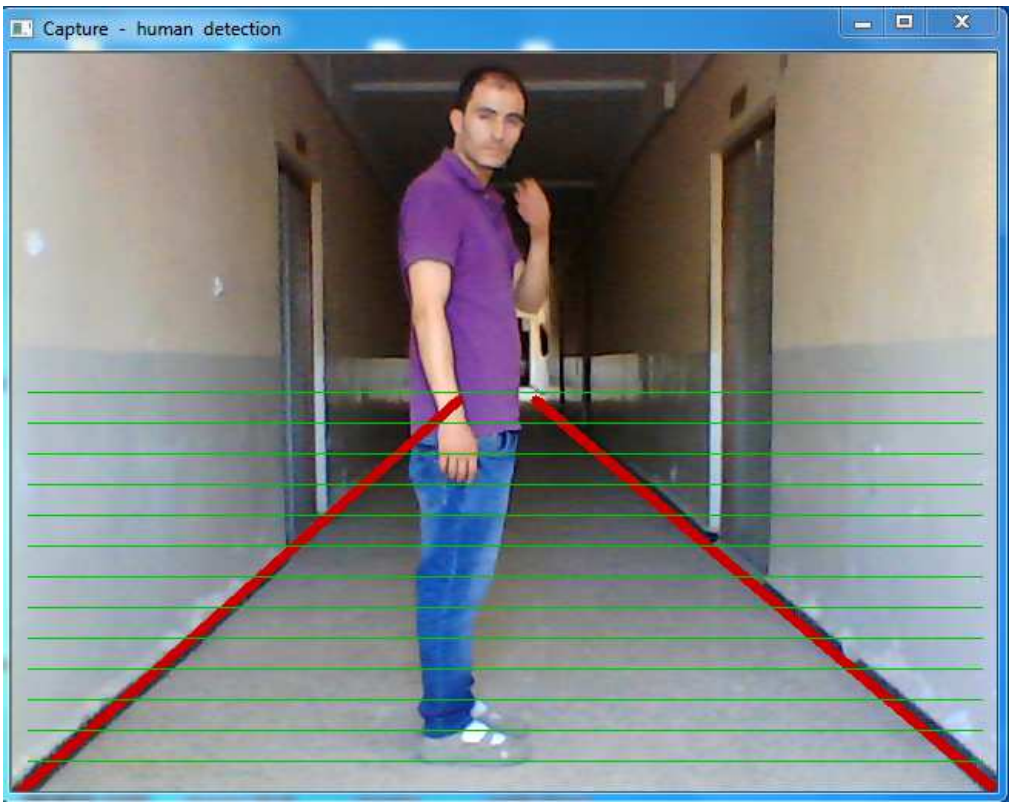


Figure III.4: image correspondante à la distance capturée à 20 pixels.

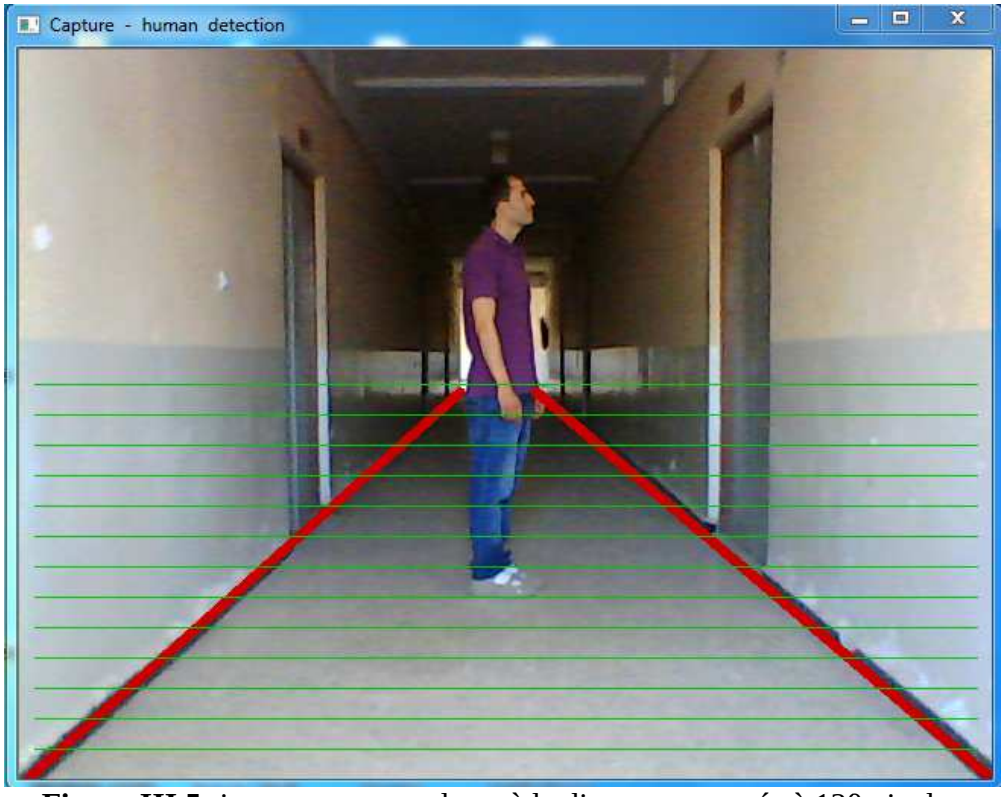


Figure III.5: image correspondante à la distance capturée à 120 pixels.

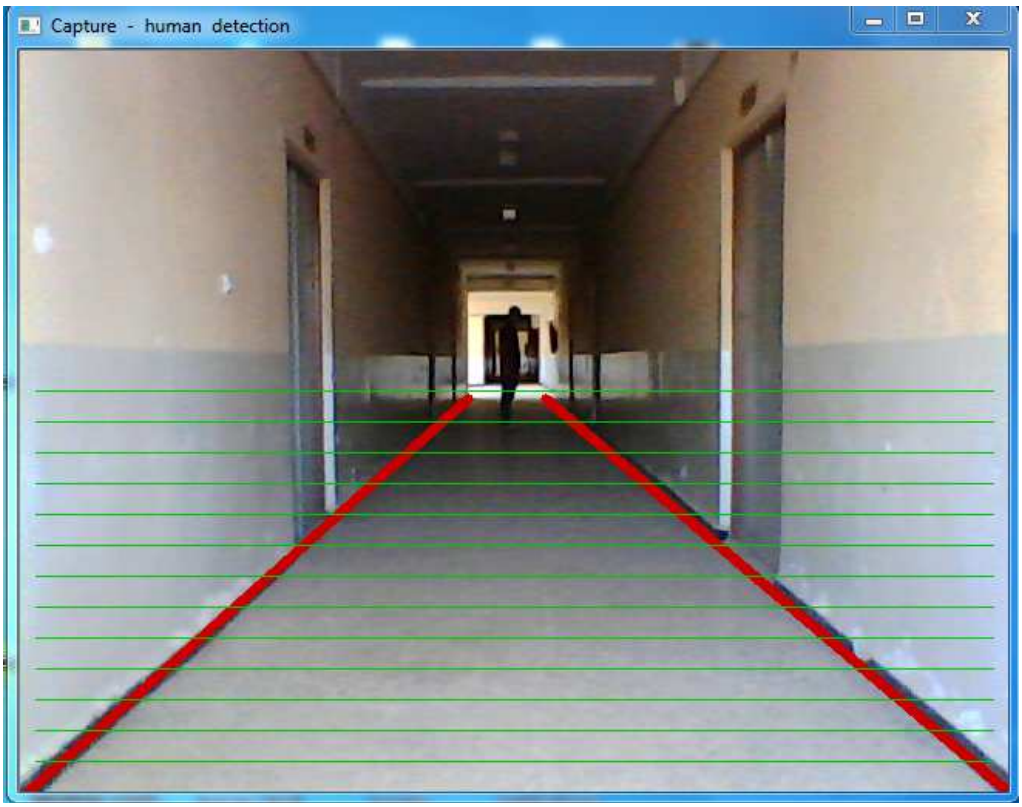


Figure III.6: image correspondante à la distance capturée à 180 pixels.

Avec ces valeurs obtenus et à l'aide de Matlab et on a obtenu le graphe suivant :

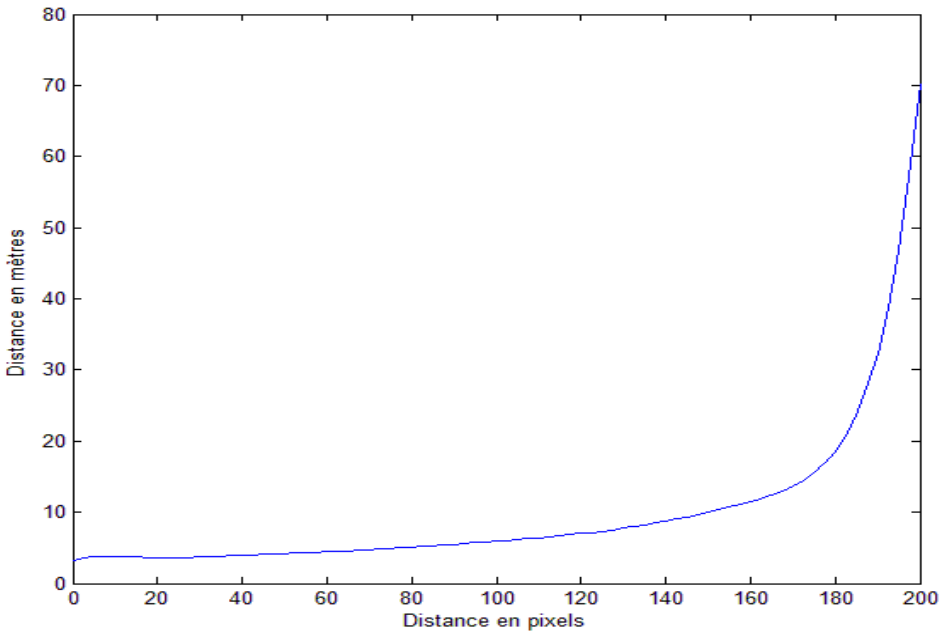


Figure III.7: graphe représente la distance réelle(m) en fonction des pixels.

-Lors de l'exécution de notre programme (l'application)

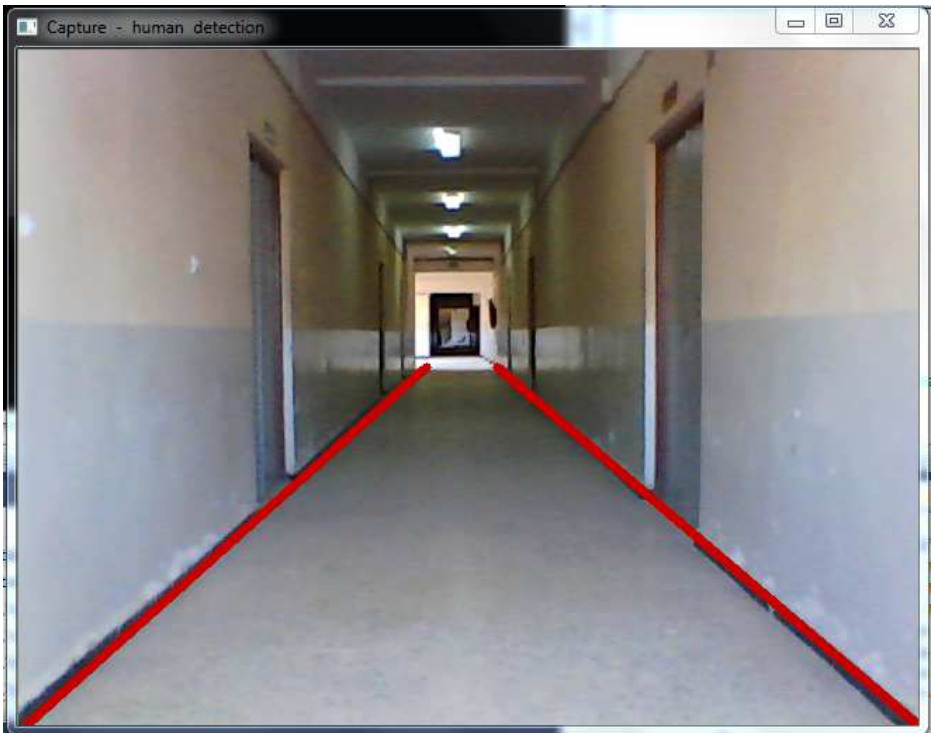


Figure III.8: le champ de vue de la caméra

Discussion :

Nous lance notre programme la ou nous avons trace notre champ de travail après avoir calibré notre camera nous obtenons notre route comme suit sur la figure III.8

-Détection de piéton : Un exemple d'une détection d'un piéton dans notre champ de travail sans estimation de la distance.

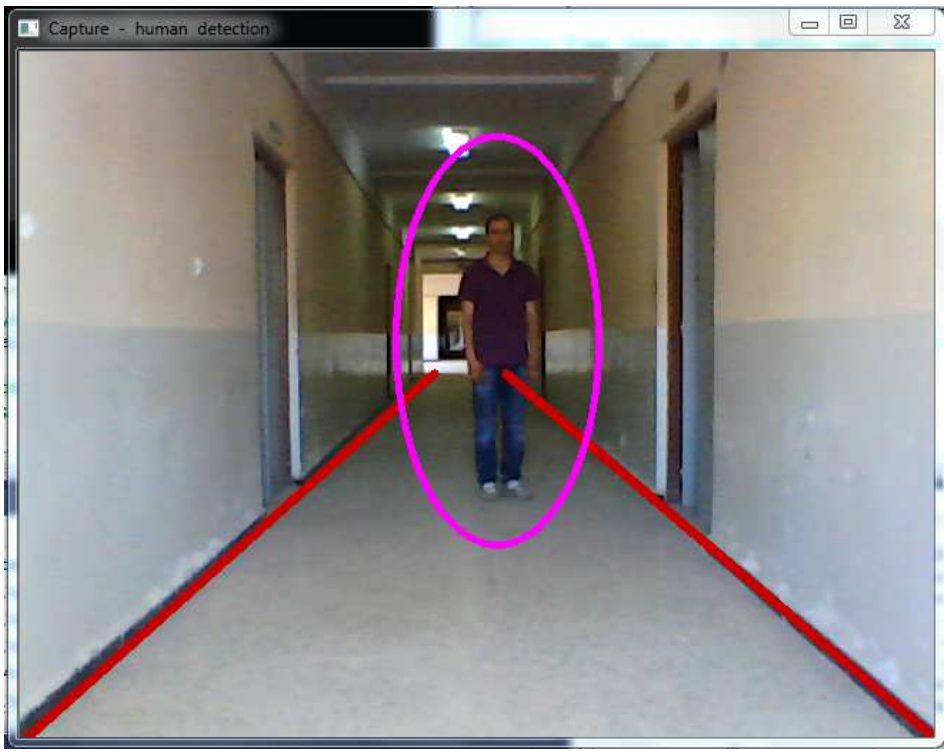


Figure III.9: détection d'un piéton

-Discussion :

Après exécution de notre programme nous remarquant dans la figure III.9 que l'humain est détecté et cerné avec l'ellipse rose donc notre programme c'exécute parfaitement.

-Calculer et afficher la distance où se situe le piéton : Un exemple d'une détection d'un piéton dans notre champ de travail avec estimation de la distance.

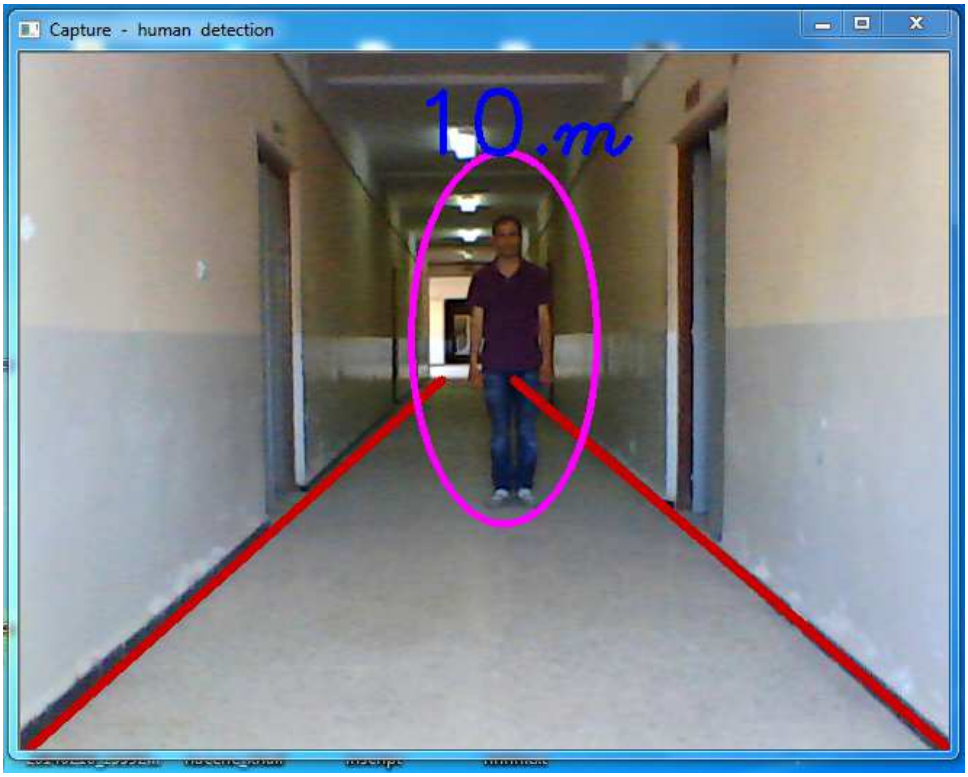


Figure III.10 : détection d'un piéton avec estimation de la distance

Discussion :

Dans cette étape nous exécutons notre programme après avoir implémenté la fonction polynomial de Lagrange pour l'estimation de la distance nous apercevrons dans la figure III.10 que l'humain est bien détecté avec l'estimation de sa distance par-rapport au véhicule, donc notre programme c'exécute parfaitement.

Figure III.11: détection de multiples piétons

Discussion :

Dans cette étape nous exécutons notre programme avec plusieurs personnes nous apercevrons encore une autre fois dans la figure III.11 que les humains sont parfaitement détecté avec estimation de leurs distance par-rapport au véhicule, donc notre programme c'exécute parfaitement.

-Détection rate, certainement dues à la faible résolution de piéton :

Figure III.12 : détection ratée

Discussion :

Après avoir exécuté notre programme avec cette prise de vue nous apercevrons que l'humain n'est pas détecté, certainement due certainement à la faible résolution et intensité du piéton.

-un emplacement qui n'est pas dans notre champ de travail :

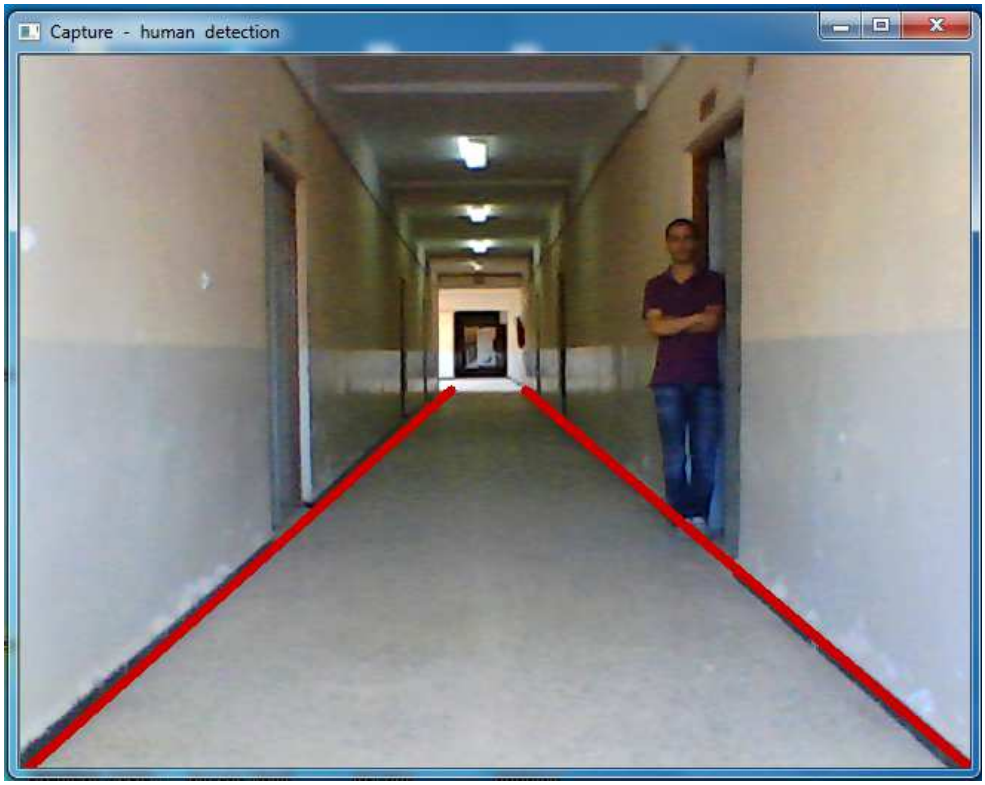


Figure III.13: tentative d'une détection qui n'est pas dans notre champ de travail

Discussion :

Après avoir exécuté notre programme avec cette prise de vue nous apercevrons que l'humain n'est pas détecté, car l'humain n'est pas dans le champ de détection.

III)-Conclusion :

Dans ce chapitre nous avons, au premier lieu, présenté les différents environnements d'implémentation et de développement, et les outils et langages de programmation que nous avons utilisé pour implémenter notre application.

Par la suite, nous avons présenté les différentes taches que notre application peut réalisé.

PDF-XChange Viewer
Click to buy NOW!
www.docu-track.com

Conclusion générale

Conclusion générale

PDF-XChange Viewer Click to buy NOW! www.docu-track.com

Comme annoncé en introduction, le but principal de ce travail était de contribuer à la réalisation d'un système de protection des piétons et du conducteur contre toute insécurité routière et nous avons également assimilé le suivi d'humain cible et la détermination de sa distance de manière continue et fiable tout au long du flux vidéo par rapport au véhicule traité en temps réel.

Nous nous sommes attachés à mettre au point une méthode de reconnaissance de piétons rapide et fiable. Pour cela nous avons dans un premier temps choisi un descripteur d'images, avec pour objectif, qu'il soit suffisamment informatif mais peu coûteux en temps de calcul. Nous avons tout d'abord utilisé un descripteur classiquement usité dans ce domaine : les ondelettes de Haar.
Vient ensuite la tâche d'apprentissage qui utilise les informations fournies par le descripteur en amont en sélectionnons les meilleurs descripteurs. Là encore nous avons mis en œuvre la méthode les plus populaire de cette dernière décennie : méthodes de Boosting avec un algorithme de type AdaBoost
En ce qui concerne les descripteurs d'images, nous en avons testé les ondelettes de Haar, malgré sa simplicité, il obtient de très bons résultats. De plus, le passage vers des applications temps réel est facilité car il est très peu coûteux en temps de calcul.
Nous espérons que notre mémoire suscitera un engouement et un intérêt scientifique chez les promotions futures afin qu'elles puissent améliorer le modeste travail que nous avons entamé, et ce en développant l'étude et la mise en œuvre des algorithmes de la détection d'humain autre que ceux basés sur les descripteurs de Haar et de l'algorithme du AdaBoost…etc.

Toute fois cette étude nous a permis d'entrevoir avec plus de motivations d'éventuels travaux de recherche dans le vaste domaine qu'est la vision par ordinateur et a suscité chez nous l'envie d'explorer les multiples facettes de cette science.

Notre programme, répond parfaitement à la problématique proposée mais pourrait également être amélioré dans le futur, il sera intéressant de terminer le projet, de reprendre le travail effectué et de le compléter afin d'estimer la distance de freinage et améliorer sa vitesse d'exécution et le résultat en le combinons avec plusieurs technique.

PDF-XChange Viewer
Click to buy NOW!
www.docu-track.com

Bibliographie

Bibliographie

[1] Robert E. Schapire. The strength of weak learnability. Machine Learning, 5(2) :197–227, June 1990.

[2] Antoine Cornuéjols and Laurent Miclet. Apprentissage artificiel, concepts et algorithmes. Editions Eyrolles, August 2002.

[3] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In Proceedings of the 30th International Conference on Machine Learning (ICML), pages 148–156, Bari, Italia, 3 June 1996.

[4] Jan Sochman. Learning for Sequential Classification. PhD thesis, Czech Technical University, Prague, Czech Republic, February 2009.

[5] C. Papageorgiou and T. Poggio. A trainable system for object detection. International Journal of Computer Vision, 38 :1533, 2000.

[6] A. Mohan, C. Papageorgiou, and T. Poggio. Example-based object detection in images by components. Pattern Analysis and Machine Intelligence, 23(4) :349_361, 2001.

[7] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection”, in IEEE Proc. of the conference on Computer Vision and Pattern Recognition, pp. 886–893, 2005.

[8] H. Sidenbladh. Detecting human motion with support vector machines. Conference on Computer Vision and Pattern Recognition, 2 :188191, 2004.

[9] S.M. Yoon and H. Kim. Real-time multiple people detection using skin color, motion and appearance information. International Workshop on Robot and Human Interactive Communication, pages 331334, 2004.

[10] Richard O. Duda, Peter E. Hart, and David G. Stork. Pattern classification. John Wiley & Sons, 2001.

[11] ‘‘Morphological Shared-Weight Neural Network For Face Recognition’’, A dissertation submitted to the University of Manchester Institute of Science and Technology for the degree of MSc, August 2004.

[12]Breiman , L. et al. (1984). Classification and Regression Trees. Chapman and Hall, New York.

[13]Quinlan , J. R. (1986). Induction of decision trees. Machine Learning, 1:81–106.

[14]Quinlan , J. R. (1993). C4.5 : programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[15] Anil K. Jain and Richard C. Dubes. Algorithms for clustering data. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[16] Celeux G,.Diday E., Govaert G., Lechevallier Y., Ralam-Bondrainy H. Classification Automatique des Données. Bordas, Paris, 1989.

Bibliographie

[17]Goldberg et David , E. (1989). Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st édition.

[18] Davis , L. (1991). The genetic Algorithm Handbook. Ed.New-York : Van Nostrand Reinhold,ch.17.

[19]Miller , B. L. et Goldberg , D. E. (1995). Genetic algorithms, tournament selection, and the effects of noise. Complex Systems, 9:193–212.

[20]: Semi-supervised learning literature survey. http ://www.cs.wisc.edu/ jerryzhu/pub/ssl survey.pdf.

[21] Christopher J.C.H. Watkins and Peter Dayan. Technical note : Q-learning. Machine Learning, 8(3-4) :279–292, May 1992.

[22] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In IEEE Conference on Computer Vision and Pattern Recognition, pages 886–893, San Diego, U.S.A., 20 June 2005.

[23] David G. Lowe. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 60(20) :91–110, November 2004.

[24] Frédéric Suard, Alain Rakotomamonjy, Abdelaziz Bensrhair, and Alberto Broggi. Pedestrian detection using infrared images and histograms of oriented gradients. In IEEE Intelligent Vehicles Symposium, pages 206–212, Tokyo, Japan, 13 June 2006.

[25] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features”, in IEEE Proc. of the conference on Computer Vision and Pattern Recognition, pp. 511–518, 2001.

[26] M. Peden, R. Scurfield, D. Sleet, D. Mohan, A. A. Hyder, E. Jarawan, and C. Mathers, “Rapport mondial sur la prévention des traumatismes dus aux accidents de la circulation,” Organisation Mondiale de la Santé et Banque Mondiale,” Publication hors série, Avril 2004.