# Cascade de classifieurs génératifs et discriminants pour la détection de véhicules

# A cascade of generative and discriminative classifiers for vehicle detection

Pablo Augusto Negri Lionel Prévost Xavier Clady
Université Pierre et Marie Curie-Paris 6, CNRS FRE 2507
Institut des Systèmes Intelligents et Robotique
3 rue Galilée, 94200 IVRY SUR SEINE

pablo . negri [ at ] lisif . jussieu . fr

## Résumé

*Dans cette communication, nous présentons un algorithme de détection de véhicules par vision monoculaire embarquée. Celui-ci utilise un schéma en cascade similaire à celui proposé par Viola et Jones en 2001[23]. Trois types de descripteurs sont présentés : les filtres rectangulaires (descripteurs de Haar), les histogrammes de gradient orienté (Histograms of Gradients, HoG) et, finalement la fusion, une concaténation des deux descripteurs précédents. Une étude comparative des résultats des détecteurs génératifs (descripteurs de HoG), discriminants (descripteurs de Haar), et de leur fusion, est exposée. Ces résultats montrent que le détecteur du type fusion combine les avantages des détecteurs de Haar et de HoG.*

## Mots Clef

Détection de véhicule, cascade de classifieurs, dopage, descripteurs de Haar, histogrammes de gradient orienté, fusion de descripteurs, classifieurs discriminants et génératifs.

## Abstract

*In this communication, we present an algorithm for the onboard vision vehicle detection problem, using the attentional cascade architecture proposed by Viola and Jones in 2001[23]. Three families of features are compared : the rectangular filters (Haar's features), the histograms of oriented gradient (Histograms of Gradients, HoG) and their combination (a concatenation of the two preceding features). A comparative study of the results of the generative (HoG features), discriminative (Haar's features) detectors, and of their fusion, is presented. These results show that the last detector combines the advantages of the two others.*

## Keywords

Vehicle detection, Cascade Detector, AdaBoost, Haar features, Histogram of Oriented Gradient, feature fusion, discriminative and generative classifiers.

## 1 Introduction

Dans cette communication, nous présentons un algorithme de détection de véhicules par vision monoculaire embarquée. Celui-ci utilise un schéma en cascade similaire à celui proposé par Viola et Jones en 2001[23]. L'originalité de notre approche consiste en l'étude comparée de classifieurs génératifs et discriminants, et en leur fusion.

**Détection de véhicules par vision embarquée.** L'augmentation du parc automobile est accompagnée par une demande croissante de systèmes d'aide à la conduite, promettant une conduite plus sécurisée et confortable [10]. Dans cette optique, diverses recherches ont été menées par la communauté dite **Transports Intelligents**. Ceci se traduit par l'installation des dispositifs de hautes technologies et autres systèmes de commande dans les véhicules ainsi que sur la route. Parmi les fonctions à intégrer dans les véhicules dits intelligents, il faut distinguer celles liées aux perceptions proprioceptive (dédiée à l'état interne du véhicule) et extéroceptive (dédiée à l'environnement externe au véhicule).
Dans ce cadre de nombreux capteurs sont couramment utilisés, dont la vision. En effet, un système de vision installé dans le véhicule porteur, peut fournir une description de la localisation et de la taille des autres véhicules dans l'environnement, ainsi que de la route, des panneaux de signalisation et des autres usagers de la voirie.
Une détection des véhicules obstacles en temps réel est exigée pour ce type d'application, de manière à laisser un temps de réaction au conducteur. La recherche exhaustive des positions potentielles des véhicules sur l'image complète est prohibitive pour les applications en temps réel. Pour résoudre ce problème, la plupart des méthodes dans la littérature se décomposent en deux étapes selon un processus attentionnel [21] :

– Génération des Hypothèses (GH) : le système fournit de façon très simple et rapide des positions potentielles de véhicules afin de restreindre le champ de recherche.
– Validation des Hypothèses (VH) : les hypothèses issues de l'étape antérieure sont validées en utilisant des algo-

rithmes plus complexes et les fausses détections sont éliminées.

Les approches mises en oeuvre pour valider les hypothèses emploient soit des primitives, soit l'apparence. Les méthodes par primitives utilisent des gabarits prédéfinis pour la classe **véhicule** et réalisent une corrélation entre ceux-ci et l'image d'entrée pour la validation : les primitives peuvent être des contours horizontaux et verticaux [19], des régions ou des *templates* déformables [3, 25, 5] ou rigides [24, 22]. Les méthodes basées sur l'apparence apprennent les caractéristiques de la classe **véhicule** à partir d'un ensemble d'images. Chaque image utilisée pour l'apprentissage est représentée par un vecteur de descripteurs locaux [1] ou globaux. Puis, l'apprentissage d'un classifieur (Réseaux de neurones, *Support Vector Machine* (SVM), loi de Bayes, etc.) permet d'estimer la frontière de décision entre la classe **véhicule** et la classe **non-véhicule**.

**Détecteur de Viola et Jones.** En marge de cette classification des méthodes, Viola et Jones [23] proposent un schéma attentionel original appliqué à la détection de visages. Celui-ci repose sur une cascade de classifieurs de complexité croissante : chaque étage restreint de plus en plus la zone de recherche en rejetant une portion importante de zones ne contenant pas de visages. Cette méthode utilise des descripteurs dits de Haar appelés aussi filtres rectangulaires, déjà éprouvés par Papageorgiou *et al.* [15, 16], et l'apprentissage Adaboost proposé par Freund et Schapire [6]. Ce dernier leur permet notamment de sélectionner à chaque étage un nombre limité de descripteurs[1]. L'utilisation d'une image intégrale pour le calcul des descripteurs de Haar, combinée à la cascade, permet un fonctionnement de l'ensemble en temps réel.

**Les deux descripteurs utilisés.** Dans les travaux exposés, deux types de descripteurs de paramètres sont comparés : les filtres rectangulaires (par abus de langage, nous les désignerons par descripteurs de Haar) et les histogrammes de gradient orienté (*Histograms of Gradients*, HoG). Ces deux descripteurs sont actuellement utilisés couramment par la communauté de reconnaissance de formes pour la détections ou la reconnaissance d'objets.

Les premiers descripteurs ont été initialement introduits par [15, 16] pour la détection de piétons et de véhicules en s'inspirant d'une décomposition via des ondelettes de Haar. Cette base de filtres a été élargie par la suite par différents travaux [23, 11], qui ne correspondent plus strictement à la théorie des ondelettes. Ils sont alors nommés filtres rectangulaires ou *Haar-like filters*.

Les histogrammes de gradients orientés consistent en une organisation en histogramme des pixels d'un voisinage selon leur orientation et pondérés par leur magnitude. Dans des travaux récents [4, 26, 9], les HoG, organisés sous la forme de descripteurs *Scale Invariant Feature Transform* (SIFT) [12], ont été utilisés avec succès pour la détection de piétons. [4] regroupe les SIFT collectés sur une fenêtre puis utilise un SVM pour la classification. [26] utilise une stratégie proche de Viola et Jones en remplaçant les descripteurs de Haar par les SIFT ; un algorithme Adaboost (un classifieur SVM linéaire est associé à chaque SIFT sélectionné) est alors utilisé en cascade .

**Originalité de notre approche.** Dans toutes les approches précédentes, les HoG sont associés à des classifieurs discriminants. Soit chaque *bin* est considéré comme une dimension du vecteur de paramètres [4, 2, 9] associé à un algorithme de classification discriminant. Soit l'ensemble des *bins* de chaque descripteur SIFT [26] est associé à un classifieur discriminant, dans une approche de type *boosting*.

Notre approche propose d'intégrer aussi des classifieurs génératifs au niveau des HoG. Nous avons donc étudié le comportement de deux détecteurs similaires à celui de Viola et Jones, mais construits :

- soit avec des descripteurs de Haar associés à des classifieurs discriminants.
- soit avec des descripteurs de HoG associés à des classifieurs génératifs.
- soit avec une concaténation des deux descripteurs précédents.

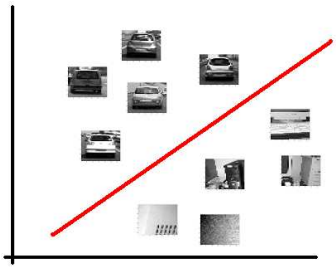
Le dernier détecteur réalise donc une fusion des descripteurs de Haar (discriminants) et de HoG (génératifs). D'autres combinaisons auraient bien entendu pu être envisagées. Certains papiers proposent aussi la fusion entre différents descripteurs (Haar+HoG [9] ou Haar+Gabor [20]), mais ils l'abordent selon un point de vue strictement discriminant dans les deux cas.

Le choix des descripteurs peut être discuté mais l'objectif principal de cette étude est de montrer la complémentarité des classifieurs génératifs et discriminants, complémentarité déjà constatée - théoriquement et expérimentalement - dans la littérature [14, 13]. Dans la suite de cette communication, nous allons développer cette idée, ainsi que présenter plus précisément la manière dont nous avons défini les descripteurs (section 2), ainsi que les classifieurs *faibles* associés utilisés dans l'algorithme de *boosting* (section 3). Ensuite nous expliciterons notre implémentation avec ou sans cascade (section 4) puis présenterons plusieurs résultats obtenus sur une base d'images embarquées pour ces trois détecteurs (section 5). Ces résultats nous permettront d'analyser et de conclure sur les comportements particuliers des descripteurs discriminants et génératifs, et de leur fusion.

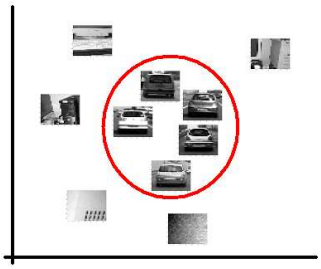
# 2 Espace de paramètres

Les exemples positifs (vignettes avec véhicules) ou négatifs (vignettes sans véhicules), se distribuent dans un espace à $N$ dimensions qui dépend des paramètres choisis pour représenter l'information. Dans l'espace initial (défini par les niveaux de gris de l'image), ces exemples peuvent être mélangés. En sélectionnant l'espace de représentation et le classifieur adéquats, on peut espérer les séparer. Dans

[1] En réalité, Adaboost ne sélectionne pas les descripteurs, mais leurs classifieurs faibles associés (cf. section 3).



Espace de paramètres discriminant



Espace de paramètres génératif

FIG. 1 – Espaces de paramètres 2D.

ce travail, deux types de paramètres ont été évalués : les filtres de Haar et les histogrammes de gradient orienté. Le premier ensemble de paramètres définit un modèle de véhicules discriminant qui sépare les deux classes à l'aide d'une frontière (hyperplan). Les exemples de test vont être classifiés selon leur position dans l'espace par rapport à l'hyperplan. Le modèle de véhicules estimé par les histogrammes est de type génératif : il est établi *a priori* et les exemples de test sont classifiés d'après leur similarité (dissimilarité) avec le modèle.
Ces espaces permettent d'extraire une information de haut niveau (contours, texture, etc) des niveaux de gris de l'image. Dans cette section, nous allons décrire les deux espaces de paramètres sélectionnés pour réaliser la détection des véhicules.

## 2.1 Filtres rectangulaires dits de Haar

Les ondelettes de Haar ou filtres rectangulaires, donnent une information sur la distribution des niveaux de gris entre deux régions voisines dans l'image.
La figure 2 montre l'ensemble des filtres de Haar utilisé dans nos travaux. Les filtres choisis sont ceux à *deux* et à *trois* rectangles. Pour obtenir la valeur (sortie) d'un filtre appliqué à une région de l'image, la somme des pixels dans le rectangle blanc est soustraite de la somme des pixels dans le rectangle gris (multipliée par un coefficient, dans le cas du filtre à trois rectangles).

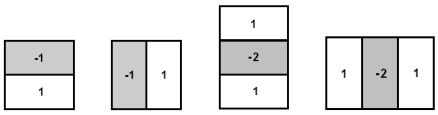



FIG. 2 – Ensemble de filtres de Haar.

Viola et Jones [23] ont introduit l'image intégrale, qui est une représentation intermédiaire de l'image d'entrée permettant de réduire le temps de calcul lié à l'application de ces filtres (lorsque ceux-ci se recouvrent). La somme d'une région rectangulaire peut être évaluée à partir de quatre références à l'image intégrale. Par conséquent, la différence entre deux rectangles adjacents est obtenue via six références à l'image intégrale, et pour le cas d'un filtre à trois rectangles, nous n'en avons besoin que de huit.
La figure 3 illustre le filtrage d'une image avec deux types de filtres rectangulaires à 2 échelles différentes : 2x2 et 4x4 pixels. Les figures montrent que les filtres choisis mettent en valeur les contours verticaux et horizontaux (pixels clairs) de l'image. Nous pouvons aussi observer qu'en doublant la taille des filtres, nous filtrons les détails de la figure originale, tout en conservant les contours principaux.

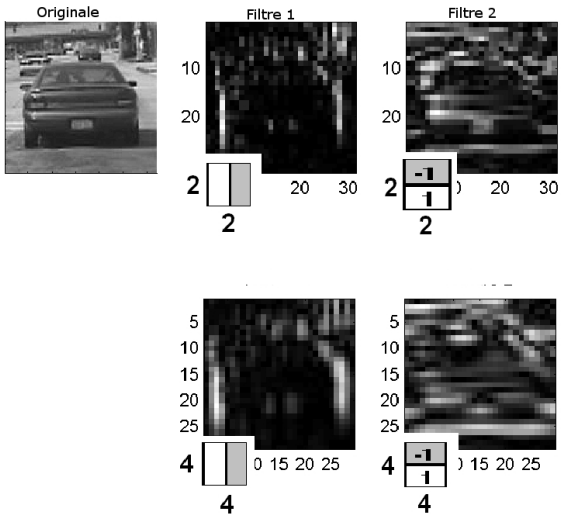



FIG. 3 – Image originale et imagettes issues de l'application de filtres de Haar verticaux et horizontaux.

Chaque descripteur $j$ est défini comme une fonction $f_j(x_j, y_j, s_j, r_j)$, où $(x_j, y_j)$ la position dans la vignette, $r_j$ est le type de filtre rectangulaire (voir fig. 2) et $s_j$ l'échelle (nous en avons 5 en total : 1x2, 2x4, 4x8, 8x16 et 16x32). Les filtres parcourent la vignette par pas d'un pixel. Pour une vignette de taille 32x32 pixels, l'espace de paramètres de Haar est donc de dimension 8151.

## 2.2 Histogrammes de gradient orienté

Les histogrammes de gradient orienté (HoG) définissent l'autre espace de paramètres présenté dans ce travail. Ce descripteur local utilise le module et l'orientation du gradient autour d'un point d'intérêt ou à l'intérieur d'une région de l'image pour construire des histogrammes.
Pour calculer le gradient de l'image d'entrée (niveaux de gris), nous appliquons un filtre de Sobel, de taille 3x3. Les orientations des pixels sont ensuite quantifiées afin qu'elles prennent des valeurs entières entre 0 et $N_{orr} - 1$ (ici $N_{orr}$ est égal à 4) en utilisant le module $\pi$ au lieu du module $2\pi$.

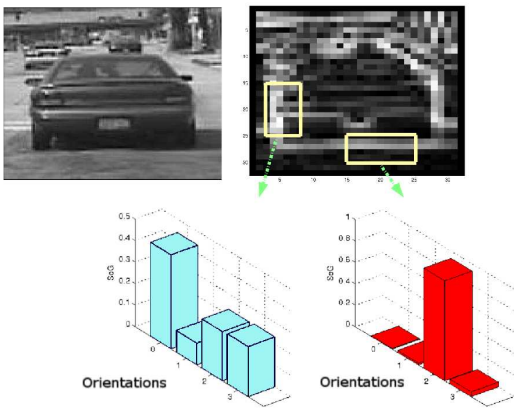



FIG. 4 – Application des HoG dans l'image d'un véhicule.

Chacun des histogrammes est construit de la façon suivante :

- nous parcourons tous les pixels de la région rectangulaire,
- pour chaque pixel d'orientation $o$, nous incrémentons la classe correspondante de l'histogramme de la valeur du module du gradient en ce point,
- une fois évalués tous les points, nous normalisons l'histogramme pour que la somme des intensités des classes soit égale à 1.

Chaque descripteur $j$ est défini comme une fonction : $h_j(x_j, y_j, s_j, r_j)$, où $(x_j, y_j)$ est la position dans la vignette, $r_j$ le type de rectangle et $s_j$ l'échelle. Les types de rectangles dépendent de la relation entre la largeur et la hauteur qui peut être (1x1), (1x2), (2x1). Nous considérons au total quatre échelles : $s : \{2, 4, 8, 16\}$. Les rectangles parcourent la vignette par pas d'un pixel. Pour une vignette de taille 32x32 pixels, l'espace de paramètres de HoG est donc de dimension 3917.

Nous observons, dans les exemples de la figure 4, que dans une des régions, les points de contours trouvés sont, en majorité, d'orientation horizontale (classe deux de l'histogramme). L'autre région contient des pixels de contours de toutes les classes, avec un plus grand nombre de points verticaux.

Nous allons utiliser une représentation intermédiaire de l'image d'entrée, inspirée de l'image intégrale, qui permet le calcul rapide des histogrammes : l'histogramme intégral [17]. Comme pour l'image intégrale, nous allons obtenir un tableau tri-dimensionnel (la troisième dimension correspondant à l'orientation) qui nous permet de calculer l'accumulation des valeurs du module du gradient pour une orientation donnée dans une région à partir de quatre références à l'histogramme intégral. Ensuite, l'histogramme complet est évalué avec $4 * N_{orr}$ références à l'histogramme intégral.

# 3 AdaBoost

La dimension des espaces de paramètres est très supérieure au nombre de pixels de l'image d'entrée. L'utilisation de ces ensembles complets pour réaliser la classification est inadéquat du point de vue du temps de calcul et de la robustesse, étant donné que certains de ces paramètres ne contiennent pas d'information pertinente (bruit).

**Adaboost.** Cet algorithme de dopage [6] s'est montré capable d'améliorer les performances de nombreux systèmes de classification et de détection. Il trouve des hypothèses précises en combinant plusieurs fonctions de classification *faibles* qui, en moyenne, ont une précision modérée. AdaBoost est un algorithme itératif qui recherche dans le vecteur de descripteurs, les fonctions de classification *faibles* les plus discriminantes pour les combiner en une fonction de classification *forte* :

$$G = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t g_t \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t = S \\ 0 & \text{sinon} \end{cases} \quad (1)$$

Où $G$ et $g$ sont les fonctions de classification respectivement $forte$ et $faible$, et $\alpha$ est un coefficient de pondération pour chacun $g$. $S$ est le seuil de la fonction de classification $G$.
Différentes variantes de l'algorithme de dopage ont été développées : Discrete AdaBoost [23], Real AdaBoost [7], Gentle AdaBoost, etc. Nous utilisons ici la première, définie par le pseudo-code suivant :

**AdaBoost Discret**

1. Soient $N$ exemples ($x_1$,$y_1$),...,($x_N$,$y_N$)
   avec $x \in \Re$ et $y_i \in \{0, 1\}$ (positifs et négatifs)
2. Initialiser $w_i = \frac{1}{N}$, $i = 1, ..., N$
3. Pour t=1,...,T
   Pour chaque descripteur $j$, entraîner à partir des $w_i$
   un classifieur $g_j$ dont l'erreur est définie par :
   $\epsilon_j = \sum_{i=1} \omega_i |g_j(x_i) - y_i|$
   Choisir le classifieur $g_t$ avec l'erreur $\epsilon_t$ la plus faible
   Actualiser les poids : $\omega_{t+1,i} = \omega_{t,i} \beta_t^{1-e_i}$
   où $e_i = 0$ si $g_t(x_i) = y_i$ , $e_i = 1$ sinon,
   avec $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$
4. sortie : $G = \sum_{t=1}^{T} \alpha_t g_t \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t$
   avec $\alpha_t = log(\frac{1}{\beta_t})$

Pour son utilisation, il nous faut à présent définir les fonctions de classification *faibles* pour les deux types de descripteurs : Haar et HoG.

**Fonction de classification de Haar.** Nous définissons la fonction de classification faible associée au descripteur $j$, comme une réponse binaire $g_{Haar}$ :

$$g_{Haar} = \begin{cases} 1 & \text{si } p_j f_j < p_j \theta_j \\ 0 & \text{sinon} \end{cases} \quad (2)$$

où $f_j$ nous donne la valeur du descripteur, $\theta_j$ est le seuil et $p_j$ est la parité. Pour chaque descripteur $j$, AdaBoost détermine le seuil $\theta_j$ optimal qui minimise le nombre d'exemples mal classés de la base d'apprentissage (positive et négative).

**Fonction de classification de HoG.** Cette fois, nous estimons la distance entre le descripteur (histogramme $h_j(x_j, y_j, s_j, r_j)$) et un histogramme modèle $m_j(x_j, y_j, s_j, r_j)$. Nous utilisons comme modèles la valeur médiane de tous les histogrammes estimée sur la base d'apprentissage positive.
Nous définissons alors le classifieur faible $g_{HoG}$ tel que :

$$g_{HoG} = \begin{cases} 1 & \text{si } d(h_j, m_j) < \theta_j \\ 0 & \text{si non} \end{cases} \quad (3)$$

où $d(h_j, m_j)$ est la distance de Bhattacharya entre l'histogramme $h_j$ et le modèle $m_j$, et $\theta_j$ est le seuil optimal sur la distance pour ce descripteur.
La distance de Bhattacharya est définie comme :

$$d(h_j, m_j) = \sqrt{1 - h_j \bullet m_j}$$

où [$\bullet$] est le produit scalaire.
Cette distance, à valeurs dans [0,1], est largement utilisée pour mesurer la similarité entre deux histogrammes.

# 4 Implémentation

Dans cette section, nous décrivons le contenu des bases d'images utilisées pour l'apprentissage et les tests. Puis, nous détaillons l'apprentissage de trois détecteurs d'architectures différentes. En phase d'utilisation, la zone de recherche dans l'image est restreinte en utilisant des connaissances a priori sur la perspective de la route. L'image est parcourue par une fenêtre glissante évaluée par le détecteur à sept résolutions différentes, multiples de 32x32.

## 4.1 Bases de données

La base de données utilisée a été fournie par l'industriel et n'est pas disponible pour le moment. Elle est constituée de plus de 800 images contenant les vues arrières d'un ou plusieurs véhicules pour un total de plus de 1200 images de véhicules. Ces véhicules sont de type voitures de tourisme, 4x4 et petits utilitaires. La base de données a été étiquetée à la main : chaque véhicule est encadré par un rectangle englobant.
Nous considérons les trois bases suivantes :

– Base de Véhicules : la base positive est composée de 745 vignettes. Cette quantité est doublée en synthétisant une image miroir par symétrie axiale. Du total des 1490 vignettes, deux tiers sont utilisés pour la Base d'Apprentissage Positive et le reste pour la Base de Validation Positive. Cette dernière nous sert à régler le seuil $S$ du classifieur *fort* en fonction d'un taux de Détections Correctes ($DC_{min}$) et/ou d'un taux de Fausses Alarmes ($FA_{max}$) requis lors de l'apprentissage (notamment de la cascade, cf. section 4.3), indépendamment des vignettes de la Base d'Apprentissage Positive qui servent à la sélection des classifieurs *faibles*. Des exemples de la base positive sont présentés dans la figure 5.
– Base de Test : composée de 230 images de scènes réelles, différentes de celles d'apprentissage, contenant 472 véhicules.
– Base Négative : les exemples qui la composent sont tirés aléatoirement dans un ensemble de plus de 4000 images quelconques (ne contenant pas de véhicules).

FIG. 5 – Exemples positifs utilisés pour l'apprentissage.

Étant donnée que la plus petite échelle d'apprentissage définie est une vignette de 32x32 pixels, nous considérons cette taille comme la taille minimale de détection de l'objet dans l'image. Cette taille correspond à une voiture située à plus de 80 mètres du véhicule porteur.

## 4.2 Détecteur simple

Un détecteur simple est composé d'un classifieur fort $G$, construit avec un nombre $T$ de descripteurs (classifieurs *faibles*). Trois détecteurs ont été crées selon les différents espaces de paramètres considérés. Les deux premiers ont été entraînés dans des espaces composés de descripteurs uniquement de Haar ou uniquement de HoG. Au troisième, nous avons fournis tous les descripteurs, Haar et HoG. 5000 vignettes ont été extraites de la Base Négative.
Pour évaluer la performance de la méthode d'apprentissage, nous allons effectuer une procédure de cross-validation. Nous entraînons trois réalisations de chaque détecteurs sur des bases d'apprentissage différentes : la distribution de la base positive en base d'apprentissage et base de validation se fait aléatoirement, tout comme le choix des exemples négatifs. Le taux de détections correctes (DC), est défini comme la moyenne des DC des trois réalisations. De la même façon, nous utilisons la moyenne des fausses alarmes (FA).

## 4.3 Détecteur en Cascade

Dans cette section, nous explicitons l'implémentation de l'architecture en cascade [23] composée d'une succession de classifieurs forts $G_i$. Chaque classifieur *fort* de la cascade est entraînée en utilisant l'algorithme Adaboost. Au lieu d'arrêter la boucle itérative d'AdaBoost pour un nombre maximum de descripteurs $T$, nous fixons deux paramètres de performance pour la fonction $G_i$ : le taux minimum de détections correctes $DC_{min}$ et le taux maximal de fausses alarmes acceptables $FA_{max}$.
La base négative $N_i$, utilisée pour entraîner le classifieur *fort* $G_i$ de l'étage $i$, est formée d'exemples négatifs ayant été mal classés (c'est-à-dire considérés comme des véhicules) par les étages précédents.

Dans un premier temps, nous avons défini quatre critères d'arrêt pour l'apprentissage de la cascade :

1. l'algorithme a réalisé le nombre maximum d'itérations pour l'apprentissage du classifieur fort $G_i$ sans arriver au taux maximum de fausses alarmes ou au taux minimum de détections correctes. Nous noterons dans nos résultats, certaines cascades *Non conv*, pour indiquer qu'il n'y a pas eu convergence d'Adaboost dans le dernier étage.
2. la cascade atteint un taux de fausses alarmes global inférieur à un taux donné, défini ici à $F = 10^{-8}$ (noté *F atteint*).
3. il n'a pas été possible de trouver un nombre d'exemples négatifs suffisant (noté *Nég. non atteint*).
4. nous sommes arrivés à un nombre fixé d'étage maximum, ici 20.

Nous obtenons trois détecteurs en utilisant les trois espaces de paramètres : les filtres de Haar, les HoG et leur fusion. Trois versions différentes pour chaque détecteur sont obtenues en faisant varier la quantité des exemples négatifs utilisée pour l'apprentissage : 1000, 2000 et 3000.

### 4.4 Cascade controlée

Pour éviter le cas de non-convergence d'une cascade (ce qui arrive fréquemment, cf. les résultats du tableau 2), nous allons intervenir dans l'apprentissage en modifiant quelque peu le critère d'arrêt d'apprentissage de la fonction $G_i$ : si une limitation fixée à chaque étage sur le nombre de descripteurs sélectionnés est atteinte sans forcément qu'il y ait eu convergence (au niveau des $DC_{min}$ et $FA_{max}$), l'itération dans Adaboost est stoppée et la fonction $G_i$ est conservée en l'état, puis nous passons à l'apprentissage de la fonction $G_{i+1}$ du prochain étage. Cette méthodologie ne trahit pas le concept de la cascade, étant donné que son but est toujours d'éliminer une portion importante des fenêtres négatives (ne contenant pas de véhicule) tout en gardant les fenêtres positives (contenant un véhicule). Pour fixer le nombre de descripteurs à chaque étage de la cascade, nous pouvons utiliser une loi croissante à notre convenance. Ici nous avons choisi de suivre une loi exponentielle.

## 5 Résultats

Dans cette section, nous analysons les résultats obtenus pour les trois types (Haar, HoG et Fusion) et les trois implémentations (simple, cascade et cascade contrôlée) de détecteurs décrits précédemment. Les indices de performances considérés sont les suivants ; le taux de détections correctes correspond au pourcentage de véhicules correctement détectés sur l'ensemble des véhicules présents dans les images de la base de test, le taux de fausses alarmes est calculé à partir de la moyenne de fausses alarmes par image (calculée sur l'ensemble des images de la base de test) divisée par le nombre total de vignettes évaluées par le détecteur dans une image. Au total, 31514 vignettes sont évaluées dans chaque image, en plusieurs positions et à différentes échelles. Enfin, le temps de traitement moyen par image est évalué sur un PC cadencé à $2.2GHz$.

### 5.1 Détecteur simple

Pour chacun des détecteurs, nous avons fait varier la valeur du nombre maximum de descripteurs ($T$ = 50, 100 et 150 descripteurs). La figure 6 présente les courbes ROC pour chaque détecteur simple (Haar, HoG et Fusion).

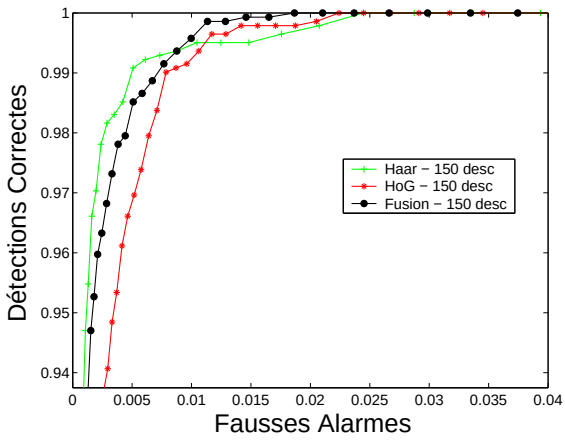



FIG. 6 – Courbes ROC pour les détecteurs simples du type Haar, HoG et Fusion.

Dans cette figure, nous constatons que :
– si on requiert un taux de fausses alarmes faibles (inférieur à 0.005), le détecteur de type Haar se montrera plus performant que le détecteur de type HoG.
– inversement, pour des taux de FA plus élevés, le détecteur de type fusion obtient de meilleurs taux de détections correctes que celui de type Haar.
– le détecteur de type HoG a un comportement similaire à celui de type fusion, mais avec une quantité plus importante de fausses alarmes.

Cette dernière remarque est confirmée par le tableau 1 qui

| Type | #Desc | DC(%) | FA | Temps(sec) |
|---|---|---|---|---|
| Haar | 50 | 99.8 | 0.0220 | 1.42 |
| Haar | 100 | 99.8 | 0.0145 | 3.51 |
| Haar | 150 | 99.0 | 0.0044 | 5.17 |
| HoG | 50 | 100 | 0.0588 | 0.90 |
| HoG | 100 | 99.9 | 0.0300 | 1.68 |
| HoG | 150 | 99.9 | 0.0233 | 2.33 |
| Fusion | 50 | 99.6 | 0.0130 | 1.67 |
| Fusion | 100 | 99.3 | 0.0093 | 3.19 |
| Fusion | 150 | 99.2 | 0.0063 | 4.75 |

TAB. 1 – Tableau des résultats obtenus pour les détecteurs simples

montre les résultats obtenus pour des détecteurs simples avec le seuil fixé de sorte à obtenir un taux de détections correctes sur la Base de Validation Positive supérieure à $99.5\%$.

Le fait d'augmenter le nombre de descripteurs raffine la frontière, dans le cas des filtres de Haar, et le modèle, dans le cas de HoG. Du point de vue des fausses alarmes, les descripteurs de Haar se montrent plus discriminants que les descripteurs de HoG. La fusion a un comportement intermédiaire entre les deux, en conservant un taux de détections important tout en éliminant un grand nombre de fausses alarmes.
Nous observons aussi que le temps de calcul par image augmente naturellement avec la quantité de descripteurs. L'utilisation d'un grand nombre de descripteurs devient irréaliste du point de vue d'une application en temps réel. D'où l'intérêt de l'architecture en cascade.

## 5.2 Détecteur en Cascade

Le tableau 2 précise l'architecture de chacun des détecteurs en cascade ainsi que leurs performances. Les valeurs des paramètres de performance pour la fonction $G_i$ sont les suivantes : le taux minimum de détections correctes $DC_{min} = 0.995$ et le taux maximal de fausses alarmes acceptables $FA_{max} = 0.40$. Nous observons tout d'abord une disparité importante au niveau de la quantité des étages de la cascade entre les trois détecteurs : la plupart des apprentissages se sont arrêtés du fait d'une non convergence de l'algorithme AdaBoost.

| Type | #Nég | #étages | #Desc | DC(%) | FA | Temps | Arrêt |
|---|---|---|---|---|---|---|---|
| Haar | 1000 | 12 | 430 | 95.4 | 0.00080 | 0.59 | Non conv |
| Haar | 2000 | 11 | 479 | 96.4 | 0.00070 | 0.57 | Non conv |
| Haar | 3000 | 10 | 272 | 97.7 | 0.00099 | 0.58 | Non conv |
| HoG | 1000 | 5 | 89 | 99.8 | 0,030 | 0.73 | Non conv |
| HoG | 2000 | 5 | 52 | 99.9 | 0,034 | 0.56 | Non conv |
| HoG | 3000 | 4 | 21 | 99.9 | 0,077 | 0.43 | Non conv |
| Fusion | 1000 | 14 | 392 | 94.5 | 0.00027 | 0.39 | F atteint |
| Fusion | 2000 | 12 | 369 | 93.9 | 0.00035 | 0.37 | Non conv |
| Fusion | 3000 | 12 | 358 | 94.3 | 0.00039 | 0.36 | Nég. non atteint |

TAB. 2 – Tableau des résultats pour les détecteurs en Cascade

Lorsque nous augmentons le nombre de vignettes négatives dans l'apprentissage, les processus ne convergent plus pour un nombre d'étages plus faible. Ceci s'explique aisément : une base négative de grande taille permet de générer un modèle ou une frontière suffisamment robustes pour éliminer dès les premiers étages de la cascade un grand nombre de fausses alarmes. Très vite, il ne reste que des cas plus difficiles à rejeter, d'où la non convergence de l'algorithme. De plus, si nous observons le nombre de descripteurs retenus à chaque étage (figure 7), nous constatons que très vite le nombre de descripteurs de HoG augmente fortement. C'est un comportement assez symptomatique des classifieurs génératifs : ils arrivent très bien à modéliser les échantillons positifs (d'où les taux de détections correctes élevés pour les détecteurs de HoG dans le tableau 2), mais ils ont besoin de complexifier fortement le modèle pour dessiner une frontière nette avec les échantillons négatifs proches des échantillons positifs. A noter aussi que les fonctions de classication forte de HoG convergent avec un nombre nettement plus faible de descripteurs dans les premiers étages que ceux de Haar. En effet, il faut peu de descripteurs pour éliminer les vignettes négatives relativement éloignées du modèle, dont le nombre est encore suffisant pour arriver à convergence. Inversement, le détecteur de Haar nécessite un plus grand nombre de descripteurs pour estimer correctement une frontière entre les classes.

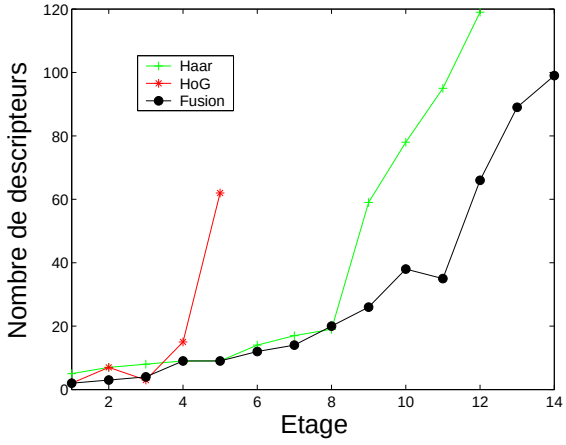



FIG. 7 – Quantité de descripteurs retenus à chaque étage pour le détecteur à 1000 négatives.

A nouveau, nous pouvons constater que la combinaison des deux descripteurs dans le détecteur de fusion permet d'améliorer les performances : il utilise les descripteurs de HoG pour éliminer les échantillons négatifs éloignés du modèle et ceux de Haar pour ceux qui sont proches de la frontière. Cependant, les cascades ne s'arrêtant pas au même nombre d'étages, ce comportement se traduit mal dans les résultats du tableau 2.
Nous vérifions aussi que le nombre de fausses alarmes est fortement lié au nombre d'étages dans la cascade. Une quantité plus importante d'étages permet d'éliminer plus des fausses alarmes[2]. Le controle du processus d'apprentissage sur le nombre de descripteurs limité à chaque étage, proposé dans la section 4.4, nous permet comme nous allons le voir d'augmenter le nombre d'étages et donc les performances des détecteurs. De plus, cette approche nous permet d'obtenir un même nombre d'étages pour tous les détecteurs, et ainsi de confirmer les hypothèses que nous venons d'exposer.

## 5.3 Détecteur en cascade controlée

La figure 8 présente la quantité de descripteurs par étage des détecteurs entrainés en utilisant 1000 exemples négatifs pour l'apprentissage. Lorsqu'un point est situé sous la loi exponentielle, cela signifie que l'apprentissage de la fonction forte a convergé avant d'arriver à la limite maximale du nombre de descripteurs pour cet étage.

[2]Cependant le nombre des détections correctes peut aussi diminuer.

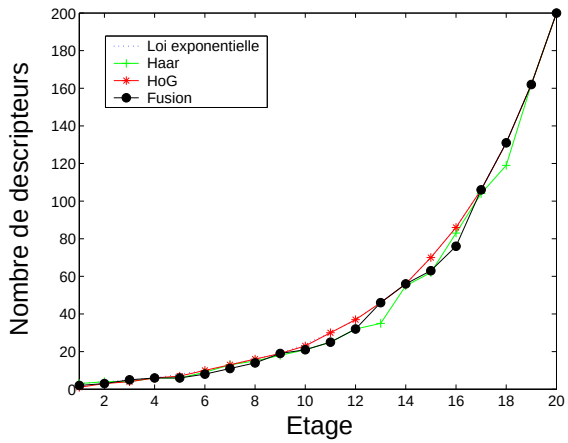



FIG. 8 – Quantité de descripteurs par étage pour les trois détecteurs.

Le détecteur de type HoG converge avant d'arriver au maximum dans les premiers étages avec une très faible quantité de descripteurs. Par contre, dans les étages supérieurs, il sature, ne pouvant plus arriver à la convergence. Le détecteur basé sur les filtres de Haar n'arrive pas à la convergence dans les premiers étages, mais il le fait à partir de l'étage 10. Le détecteur de type fusion a un comportement intermédiaire entre les deux antérieurs. Dans la figure 9, illustrant l'évolution de la proportion de descripteurs de HoG par rapport au nombre de descripteurs total sélectionnés à chaque étage, nous observons que, dans les étages initiaux, ce sont les descripteurs de HoG qui ont été choisis comme étant les plus discriminants. Dans les derniers étages, ce sont plutôt les descripteurs de Haar qui composent les fonctions *fortes*.

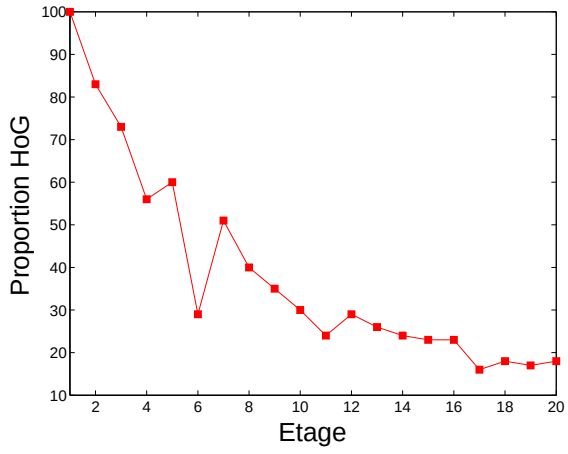
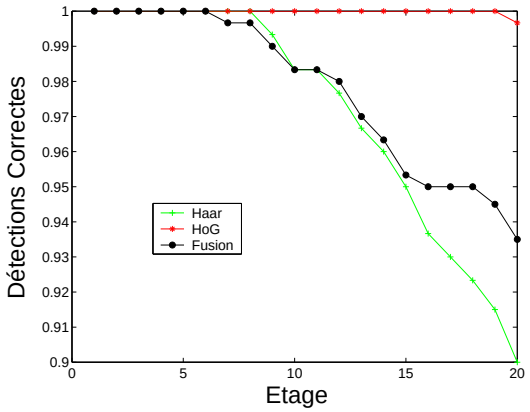



FIG. 9 – Proportion de descripteurs de HoG sélectionnés pour chaque étage de la cascade du détecteur de type Fusion.

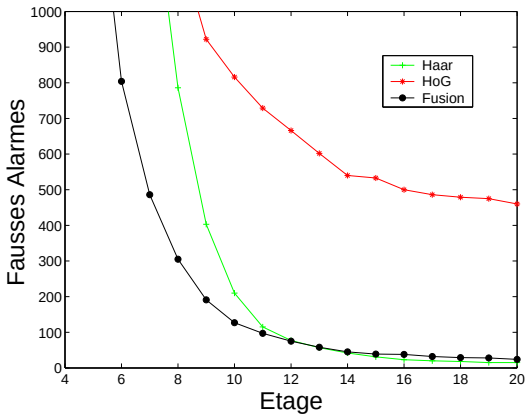
Ceci confirme les remarques énoncées précédemment pour les détecteurs en cascade sans contrôle du nombre de descripteurs sélectionnés. De plus, cela se traduit dans le tableau 3 qui montre les performances globales des détecteurs en cascade contrôlée. Le détecteur de HoG obtient un taux de détections correctes important, cependant le taux de fausses alarmes reste aussi élevé.

| Type | # Neg | # Desc | DC (%) | FA | t (seg) |
|---|---|---|---|---|---|
| Haar | 1000 | 1016 | 93.8 | 0.00031 | 0.66 |
| Haar | 3000 | 942 | 89.83 | 0.00018 | 0.69 |
| HoG | 1000 | 1027 | 97.8 | 0.0045 | 0.51 |
| HoG | 3000 | 1031 | 99.6 | 0.0114 | 1.07 |
| Fusion | 1000 | 1022 | 94.0 | 0.00029 | 0.36 |
| Fusion | 3000 | 1021 | 93.5 | 0.00032 | 0.40 |

TAB. 3 – Tableau des résultats des détecteurs en Cascade.




(a) taux de détections correctes par étage




(b) quantité de fausses alarmes par étage

FIG. 10 – Comportement des détecteurs en fonction des étages de la cascade pour le détecteur à 3000 négatifs.

Le détecteur Haar a un comportement inverse, à la fin de la cascade, le détecteur obtient une faible quantité de fausses alarmes, mais de nombreux exemples positifs ont été éliminés dans les étages précédents (figure 10). Cette figure décrivant l'évolution des taux de détections et de fausses alarmes selon le nombre d'étages considérés illustre bien, avec la figure 9, le fait que le détecteur Fusion combine les avantages des deux descripteurs : génératifs pour HoG et

discriminants pour Haar. Dans les premiers étages, il utilise essentiellement les descripteurs génératifs pour éliminer les échantillons négatifs éloignés du modèle en conservant un taux de détections élevés, puis dans les étages élevés de la cascade, il se sert des descripteurs de Haar pour dessiner des frontières nettes entre les exemples positifs et ceux négatifs encore présents (proches du modèle).
Ceci se reflète très concrètement dans les images de scènes routières de la figure 11, où les carrés blancs signalent une détection (correcte ou fausse) obtenue. Nous observons que le détecteur de Haar ne détecte pas tous les véhicules mais ne produit que peu de fausses alarmes. En revanche, le détecteur de HoG réalise de nombreuses fausses alarmes, mais détecte quasiment tous les véhicules . Enfin, le détecteur de fusion réduit sensiblement le nombre de fausses alarmes tout en détectant les mêmes véhicules que celui de HoG.
Ainsi le détecteur de type Fusion obtient des performances supérieures aux deux autres, en particulier au niveau du temps de réponse. Cela s'explique aisément en analysant la courbe 10 (b) où le nombre des hypothèses rejetées par le détecteur de type Fusion est très important dès les premiers étages de la cascade.

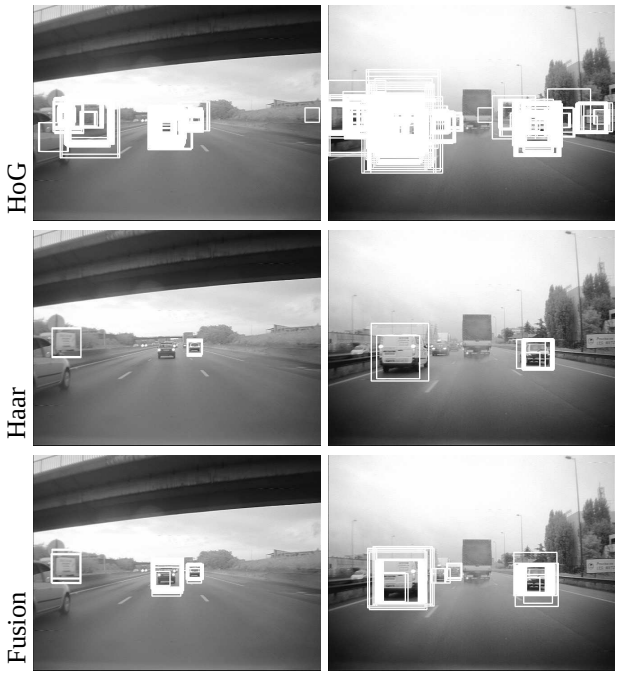



FIG. 11 – Exemples de résultats reportés sur des images de scènes autoroutières pour les trois détecteurs.

# 6 Conclusions

Nous avons présenté dans cette communication deux espaces de paramètres, les descripteurs de Haar et les histogrammes de gradient orienté, appliqués à la détection de véhicules et en utilisant une approche similaire à celle de Viola-Jones. Les premiers sont associés à des classifieurs *faibles* de type discriminants et les seconds à des classifieurs de type génératifs. Un troisième détecteur est obtenu à partir d'une fusion de ces deux espaces.
Nous avons étudié le comportement de différentes architectures : le détecteur simple et le détecteur en cascade. Pour optimiser la performance du détecteur en cascade, nous avons fixé le nombre maximum des descripteurs dans chaque étage de la cascade, c'est-à-dire dans chaque fonction de classification *forte*.
Le détecteur réalisant la fusion des deux descripteurs combine les avantages des précédents détecteurs de Haar et de HoG : un taux de détections correctes élevé et un faible nombre de fausses alarmes. Il utilise les classifieurs de type génératifs pour éliminer les échantillons négatifs éloignés du modèle, puis il se sert des descripteurs discriminants pour dessiner des frontières nettes entres les exemples positifs et ceux négatifs proches, encore présents.
Cette étude démontre que les méthodes de dopages (*boosting*) sélectionnent automatiquement les classifieurs génératifs en premier (pour leur efficacité), puis les discriminants. Auparavant, ceci était fait de façon intuitive en combinant séquentiellement les deux types de classifieurs [8, 18].
Nos travaux futurs seront consacrés à l'utilisation de ces descripteurs pour associer à la détection des véhicules, une classification de ceux-ci en plusieurs classes de type : véhicules de tourisme, 4x4, utilitaires, poids lourds et bus.

## Remerciements

Cette recherche est soutenue par Peugeot Citroën Automobile (PCA). Nous voulons remercier M. Fabien Hernandez de la Direction de la Recherche et de l'Innovation Automobile de PCA pour son support.

# Références

[1] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *ECCV*, pages 113–130, London, UK, 2002. Springer-Verlag.

[2] J. Begard, N. Allezard, and P. Sayd. Real-time humans detection in urban scenes. In *BMVC*, University of Warwick, UK, 2007.

[3] J.M. Collado, C. Hilario, A. de la Escalera, and J.M. Armingol. Model based vehicle detection for intelligent vehicles. In *ISIV*, pages 572–577, 2004.

[4] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, volume 2, pages 886–893, June 2005.

[5] M. Dubuisson, S. Lakshmanan, and A. Jain. Vehicle segmentation and classication using deformable templates. *IEEE Transactions on PAMI*, 18(3) :293–308, March 1996.

[6] Y. Freund and R. Schapire. Experiments with a new boosting algorithm. In *International Conference on Machine Learning*, pages 148–156, 1996.

[7] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression : a statistical view of boosting. *The Annals of Statistics*, 28(2) :337–374, April 2000.

[8] M. Fritz, B. Leibe, B. Caputo, and B. Schiele. Integrating representative and discriminative models for object category detection. In *ICCV*, pages II : 1363–1370, 2005.

[9] D. Geronimo, A. Lopez, D. Ponsa, and A.D. Sappa. Haar wavelets and edge orientation histograms for on-board pedestrian detection. In *IbPRIA*, pages 418–425, 2007.

[10] S. Han, E. Ahn, and N. Kwak. Detection of multiple vehicles in image sequences for driving assistance system. In *ICCSA 2005*, volume 3480, pages 1122–1128, 2005.

[11] R. Lienhart, A. Kuranov, and V. Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *DAGM03*, pages 297–304, 2003.

[12] D.G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, pages 1150–1157, 1999.

[13] P. Negri, X. Clady, and L. Prevost. Benchmarking haar and histograms of oriented gradients features applied to vehicle detection. In *ICINCO*, pages 359–364, 2007.

[14] A. Y. Ng and M. I. Jordan. On discriminative vs. generative classifiers : A comparison of logistic regression and naive bayes. In *NISP*, pages 841–848, 2001.

[15] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio. Pedestrian detection using wavelet templates. In *CVPR*, pages 193–199, Puerto Rico, 1997.

[16] C. Papageorgiou and T. Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1) :15–33, 2000.

[17] F. Porikli. Integral histogram : A fast way to extract histograms in cartesian spaces. In *CVPR*, pages 829–836, 2005.

[18] L. Prevost, L. Oudot, A. Moises, C. Michel-Sendis, and M. Milgram. Hybrid generative/discriminative classifier for unconstrained character recognition. *Pattern Recognition Letters*, 26(12) :1840–1848, 2005.

[19] N. Srinivasa. Vision-based vehicle detection and tracking method for forward collision warning in automobiles. In *IVS*, volume 2, pages 626–631, 2002.

[20] Z. Sun, G. Bebis, and R. Miller. On-road vehicle detection using evolutionary gabor filter optimization. In *IEEE Transactions on Intelligent Transportation Systems*, 2005.

[21] Z. Sun, G. Bebis, and R. Miller. On-road vehicle detection : A review. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(5) :694–711, 2006.

[22] T. N. Tan and K. D. Baker. Efficient image gradient based vehicle localization. *IEEE Transactions on Image Processing*, 9 :1343–1356, 2000.

[23] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, volume I, pages 511–518, 2001.

[24] H. Yang, J. Lou, H. Sun, W. Hu, and T. Tan. Efficient and robust vehicle localization. In *ICIP*, volume 2, pages 355–358, Thessaloniki, Greece, 2001.

[25] A.H.S. Yung, N.H.C.and Lai. Detection of vehicle occlusion using a generalized deformable model. In *ISCAS*, volume 4, pages 154–157, 1998.

[26] Qiang Zhu, Mei-Chen Yeh, Kwang-Ting Cheng, and Shai Avidan. Fast human detection using a cascade of histograms of oriented gradients. In *CVPR*, pages 1491–1498, Washington, DC, USA, 2006. IEEE Computer Society.