# Fusion multi-sources pour l'interprétation d'un environnement routier

Alexandre Bargeton


ED n°431 : Information, Communication, Modélisation et Simulation (ICMS)

# T H E S E

pour obtenir le grade de

**DOCTEUR DE L'ECOLE NATIONALE SUPERIEURE DES MINES DE PARIS**

Spécialité "Informatique temps réel – Robotique – Automatique"

présentée et soutenue publiquement par
**Alexandre BARGETON**

le 16 décembre 2009

## Fusion multi sources pour l'interprétation de l'environnement routier

***Directeur de thèse : M. Fawzi NASHASHIBI***
***CoDirecteur de thèse : M. Fabien MOUTARDE***

Jury

| | |
|---|---|
| M. Claude LAURGEAU | Président du jury |
| M. Abdelaziz BENSHRAIR | Rapporteur |
| M. Fabrice MERIAUDEAU | Rapporteur |
| M. Maurice MILGRAM | Examinateur |
| M. Benazouz BRADAI | Examinateur |
| M. Fawzi NASHASHIBI | Examinateur |
| M. Fabien MOUTARDE | Examinateur |

*Ce mémoire est dédié à tous ceux*

*qui ne se sont pas dédiés leur mémoire à eux-mêmes.*

# Table des matières

# Introduction

Si la voiture est le moyen de transport le plus utilisé, les accidents de la route sont malheureusement la principale cause de décès des personnes âgées de 2 à 33 ans (87% étant des occupants des véhicules, 13% des piétons). Aux Etats-Unis, selon une étude de la National Highway Traffic Safety Administration (NHTSA) (Subramanian, 2006), le taux de mortalité lié aux accidents de voiture ne cesse d'augmenter. En moyenne, 117 personnes meurent chaque jour aux Etats-Unis dans un accident de voiture, c'est à dire 1 personne toutes les 12 minutes. Dans 75,4% des cas, ces accidents sont causés par des erreurs de conduite (Bhatia, 2003). Parallèlement à ce constat meurtrier, le coût économique dû aux accidents de voiture est non négligeable : aux Etats-Unis, il s'élevait pour l'année 2000 à 230,6 milliards de dollar.

En Europe, un rapport du European Road Safety Observatory (ERSO) révèle que bien que le nombre de décès liés aux accidents de la route soit en baisse, ce nombre d'accidents reste quant à lui constant (ERSO, 2007). En France, la Direction de la sécurité et de la circulation routières dans son bilan 2005 révèle la même tendance, mais constate aussi une augmentation du nombre d'accidents en milieu urbain avec d'avantage de personnes tuées chez les piétons et les cyclomotoristes (Routière, 2006). Une étude de l'Observatoire National Interministériel de Sécurité Routière (ONISR) (Chapelon, 2009) révèle qu'après le taux d'alcoolémie positif, le dépassement des limitations de vitesse est la principale cause de mortalité dû aux accidents de la route (Figure 1).

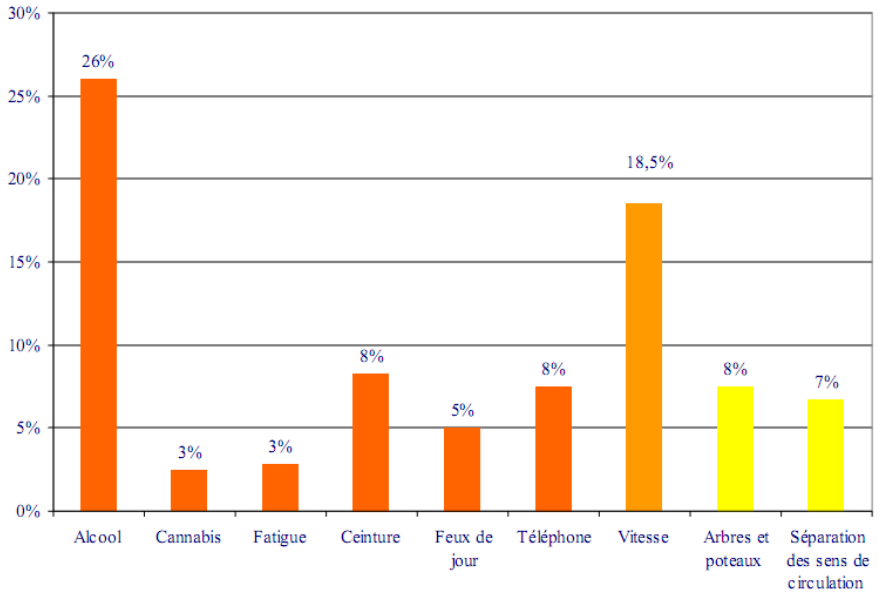



Figure 1 - Répartition des types d'infractions relevés lors d'accidents mortels de la route en France

La tendance de ce bilan n'a malheureusement pas été inversée ces dernières années. Un rapport de 2008 de la Sécurité Routière (Routière1, 2009) déplore une augmentation des infractions dues à la vitesse de 12% par rapport à l'année précédente. Ainsi, même si la mise en place du contrôle automatisé (radar) des vitesses des véhicules a permis d'engendrer une baisse significative des vitesses moyennes pratiquées sur les routes de France (Figure 2) et d'épargner 210 vies en 2008 par rapport à 2007, 820 vies auraient pu être encore épargnées si tous les conducteurs avaient respecté les limitations de vitesse (Routière2, 2009). Le dépassement des limitations de vitesse, même s'il est globalement en léger retrait en 2008, reste un comportement de masse puisque c'est le cas d'environ 34% des automobilistes, 43% des conducteurs de poids lourd et 54% des motocyclistes, tous réseaux confondus (Routière2, 2009). C'est en ville que les taux de dépassement des limitations sont les plus élevés avec 55% sur les voies d'entrées / sorties d'agglomération.

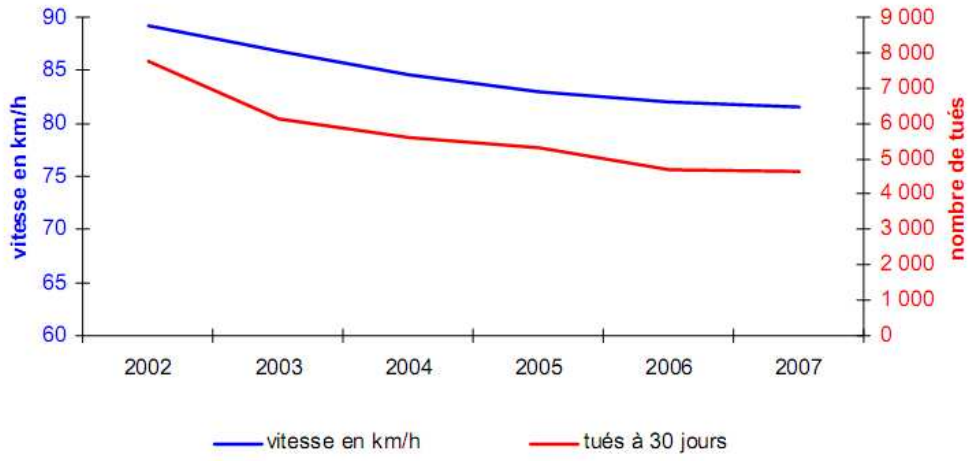



Figure 2 - Evolutions comparées des vitesses et des tuées de 2002 à 2007 en France

La vitesse représente donc encore un enjeu majeur pour la sécurité routière dans les années avenir.

Les Systèmes de Transport Intelligent (ITS) sont définis comme l'application des techniques mécatroniques et informatiques modernes et des technologies de communication aux systèmes de transport, afin d'améliorer leur efficacité, de réduire la pollution et des autres effets néfastes environnementaux et d'augmenter la sécurité des voyageurs et assurer leur confort.

Plus particulièrement, les « véhicules intelligents » offrent la possibilité d'améliorer significativement la sécurité des individus. Un « véhicule intelligent » peut être défini comme un véhicule qui perçoit l'environnement pour produire des actions automatiques et des aides à la conduite.

Les environnements urbain et autoroutier où les voitures évoluent sont intrinsèquement sujets à des changements dynamiques (les véhicules faisant parties intégrantes de l'environnement). Ainsi, l'un des problèmes de recherche présents et futurs est le développement de méthodes s'appuyant sur des capteurs de perception pour la modélisation et la prédiction en temps réel des dynamiques de l'environnement, ceci afin d'améliorer la sécurité et le confort des usagers.

L'un des moyens pour y arriver est de créer un système intégré aux « véhicules intelligents » (DSS - Driver Support System) qui assiste le conducteur dans ses réactions face aux changements des conditions routières, ceci afin d'accroître son temps d'anticipation nécessaire à l'adaptation du mouvement de son véhicule face aux évolutions des autres véhicules dans ce même environnement.

Dans ce contexte, les travaux de thèse présentés dans ce mémoire, intitulés

« Fusion multi sources pour l'interprétation de l'environnement routier »

portent sur la réalisation d'un système d'aide à la conduite automobile (ADAS – Advanced Driving Assistance System) répondant à cet enjeu majeur de la sécurité routière d'aujourd'hui : aider le conducteur à respecter les limitations de vitesse.

Pour cela le système repose sur la fusion multi sources, lui permettant de détecter, reconnaître et surtout d'interpréter les limitations de vitesse (certaines limitations ne concernant qu'un ensemble restreint de véhicules ou de voies de circulations). Les sources utilisées ici sont celles des capteurs proprioceptifs, permettant de connaître l'état interne du véhicule (vitesse du véhicule,…), celles des capteurs extéroceptifs, mesurant des caractéristiques du monde extérieur (caméra,…), ainsi que celles de nouveaux types de capteurs (données cartographiques,…).

La suite de ce rapport, décrit chapitre par chapitre chaque source mise en place pour le système de fusion :

- Le premier chapitre porte sur la réalisation d'un capteur visuel des panneaux de limitation de vitesse. Cette source, bien qu'efficace, se révèle seule inutile dans les cas de manque d'interprétation des panneaux (voie de sortie par exemple) ou d'occultation de ceux-ci.

- Le second porte sur la création et l'utilisation d'un capteur cartographique permettant de connaître la topologie de la route et de donner une information statique sur la limitation de vitesse sur la route empruntée par le véhicule. Là aussi, cette seule source d'information ne suffit pas pour déterminer la limitation de vitesse, car elle ne fournira jamais les limitations temporaires (zones de travaux par exemple) ou variables (comme pour la régulation du trafic sur autoroute).

- Le troisième chapitre décrit la détection visuelle des lignes de marquage au sol. Cette information sera très utile pour aider à l'interprétation des détections des panneaux de limitation de vitesse.

- Le quatrième chapitre, décrit le système de fusion, basé sur les précédentes sources, mis en place pour répondre à la problématique de l'aide à la conduite pour la connaissance des limitations de vitesse. Le principal but de ce système, n'est pas comme il sera démontré, de fusionner plusieurs données similaires (i.e. plusieurs limitations de vitesse), mais d'utiliser l'ensemble des sources décrites précédemment pour interpréter les résultats de chacune de ces sources et créer ainsi un système global cohérent et robuste pouvant faire face à de nombreuses situations ambiguës.

Enfin, un dernier chapitre décrit les outils développés ayant servis à l'élaboration de ce travail dont un éditeur de vérité terrain ayant permis d'évaluer la qualité des algorithmes mis en place, et un logiciel d'apprentissage numérique pour l'utilisation d'algorithmes d'intelligence artificielle utilisés dans la reconnaissance visuelle des panneaux de limitation de vitesse.

# Détection des panneaux de limitation de vitesse

## 1.1. Motivations

La détection et la reconnaissance des panneaux signalétiques (*TSR - Traffic Sign Recognition*) a connu un grand engouement ces dernières années. Cela est dû à la grande quantité d'applications qu'un tel système est capable de fournir :

1. Maintenance des panneaux

De nos jours, un opérateur humain doit visionner des films afin de vérifier la présence et l'aspect des panneaux sur les routes. C'est une tâche pénible car les panneaux n'apparaissent que de temps en temps et parce que l'opérateur doit être extrêmement attentif.

2. Inventaire des panneaux

Répertorier les panneaux présents sur les routes permet entre autre de pouvoir réaliser des bases de données des informations utiles au conducteur (c'est notamment ce qu'utilisent les systèmes GPS embarqués).

3. Aide à la conduite

Les signes apparaissent toujours clairement dans l'environnement. Cependant, si ils sont distraits ou s'ils ont un manque de concentration, il arrive que les conducteurs ne voient pas un panneau. Or le dépassement de la vitesse limite autorisée est l'une des causes principales d'accident de la route. Il est alors important qu'un système puisse leur fournir l'information qu'ils viennent de rater et de les prévenir si ils roulent trop vite ou trop lentement.

4. Véhicules intelligents autonomes

Pour des véhicules complètement autonomes (sans conducteur humain), c'est un point primordial que de reconnaître les panneaux signalétiques qui fournissent un grand nombre d'informations utiles à la navigation.

## 1.2. Etat de l'art

Les recherches dans le domaine du TSR (*Traffic Sign Recognition*) ont commencé à partir des années 1980. La première approche naïve et directe est d'appliquer un filtre de corrélation croisée normalisée aux images brutes afin à la fois de détecter et de reconnaître les panneaux. Cependant cette méthode par brute force demande un temps de calcul beaucoup trop important pour être réellement envisageable (les panneaux pouvant avoir n'importe quelle taille et se trouver n'importe où dans l'image).

C'est ainsi que très vite, dans les années 1990 et encore de nos jours, des approches généralistes basées sur des méta-heuristiques (des algorithmes d'optimisation stochastique, hybridés avec une recherche locale) ont été utilisées afin de ne pas parcourir tout l'espace de recherche (i.e. la position et la taille des panneaux dans une image) et de trouver une solution optimale au problème en un temps raisonnable. Ont notamment été utilisées les approches par recuit simulé (Betke, et al., 1994), ainsi que par les algorithmes génétiques (Escalera, et al., 2003) (Aoyagi, et al., 1996). Cependant les méta-heuristiques ne garantissent pas de fournir la solution optimale au problème ni même une solution et un compromis doit être trouvé entre le temps d'exploration de l'espace de recherche et la qualité de la solution, comme le relève (Gavrila, 1999).

En revanche, il est possible d'effectuer le processus de corrélation de manière complètement optique, c'est-à-dire à la vitesse de la lumière en ne manipulant que l'onde lumineuse (Horache, 2001). Cet avantage a motivé beaucoup d'auteurs pour des applications telle que la reconnaissance de formes en temps réel et notamment la détection automatique de panneaux routiers (Guibert, 1995) (Petillot, 1996) (Guibert, et al., 1995). Malheureusement les corrélateurs optiques souffrent de sévères inconvénients tels que le coût et la fragilité des composants optiques, l'encombrement de ces architectures, la nécessité d'un alignement précis pour les traitements effectués et le manque de robustesse mécanique de l'optique pour des applications embarquées.

D'autres approches plus utopiques ont aussi été proposées pendant les années 90 comme dans (Pacheco, et al., 1994) où ils émettent l'idée d'ajouter des codes barre de couleur sous les panneaux afin d'aider à leur reconnaissance. De nos jours on pourrait vouloir rajouter des puces RFID aux panneaux, mais ces solutions demanderaient trop de temps et d'argent pour remplacer tous les panneaux.

Mise à part la méthode de corrélation croisée, toutes les recherches actuelles découpent le processus de TSR en trois parties comme le soulève (Gavrila, 1999):

1. Détection
2. Reconnaissance
3. Intégration temporelle

### 1.2.1. Détection

Les recherches dans la détection des panneaux signalétiques dans l'image, c'est-à-dire trouver des régions d'intérêts (ROI) où ils se situent, peuvent être classées en deux grandes catégories.

#### *a) Segmentation par couleur*

Comme relevé par (Bahlmann, et al., 2005), la segmentation par la couleur est la méthode la plus couramment utilisée dans l'étape initiale qu'est la détection des panneaux. Naïvement, de nombreux panneaux étant cerclés de rouge, de nombreux travaux (de la Escalera, et al., 1997) (Kim, et al., 1997) (Zadeh, et al., 1998) (Torresen, et al., 2004) effectuent, pour détecter ceux-ci, un seuillage sur la composante rouge du modèle RGB. Dans de rares cas, l'utilisation du modèle YUV est utilisé, comme dans (Miura, et al., 2000) où ils extraient les régions blanches de l'image (qui correspondent aux zones intérieures des panneaux). Cependant, ces modèles de couleurs sont sensibles aux changements de conditions lumineuses. C'est ainsi que (Miura, et al., 2000) utilise plusieurs seuils différents pour binariser ses images, il en résulte que pour chaque image de nombreuses autres doivent être traitées, ce qui alourdit considérablement le temps de calcul en rendant l'algorithme non utilisable pour le temps réel. La plupart des travaux (Escalera, et al., 2003) (Hsien, et al., 2003) (Piccioli, et al., 1996) (Arnoul, et al., 1996) (Hibi, 1996) (Fang, et al., 2003) (Fang, et al., 2003) se basent donc généralement sur le modèle HSV (ou HSI), ou même sur l'espace de couleur Luv (Kang, et al., 1994), qui sont invariants aux changements de conditions lumineuses. Cependant, comme le stipulent (Loy, et al., 2004) (Garcia, et al., 2006), l'image provenant de la caméra n'est pas complètement invariante aux changements chromatiques de la lumière qu'elle reçoit. En effet, la composante Hue change avec les ombres, les conditions climatiques et de plus comme l'explique (Thomson, et al., 2001) la couleur des panneaux s'estompe avec le temps. Ces méthodes de segmentation par couleur peuvent donc poser problème pour la robustesse d'un système TSR et sont de plus plus coûteuses (3 canaux à traiter).

#### *b) Segmentation par contours de forme*

Les panneaux de limitation et de fin de limitation de vitesses européens sont de forme circulaire. La méthode la plus connue et la plus robuste dans les techniques de reconnaissances de formes dans le domaine du traitement d'image afin de détecter des cercles est l'algorithme de Hough qui est utilisé dans (Garcia, et al., 2006) (Miura, et al., 2000). Cependant cette technique peut s'avérer fort lente. C'est pourquoi de nombreux travaux essayent de développer d'autres. A partir de la *distance transform* (DT) (Borgefors, 1986) qui calcule la distance pour chaque pixel de l'image au contour le plus proche, il est possible d'effectuer une corrélation « rapide » (Gavrila, 1999) pour identifier la position et la tailles de panneaux potentiels dans une image. Cette technique souffre quand même d'un temps de calcul largement trop important (dû au fait de calculer la DT des images et du processus de corrélation) pour être employé en temps réel. Une variation de l'algorithme de Hough a été proposée par (Barnes, et al., 2004) basée sur les travaux de (Loy, et al., 2003) qui propose d'avoir non pas un accumulateur comme dans l'algorithme classique, mais deux où le second accumule l'amplitude du gradient. Cette approche ne peut donc pas être plus rapide que l'algorithme classique. Une autre variation est proposée dans (Ajdari Rad, et al., 2003) où l'idée est de retrouver simplement quelques paires de vecteurs sur l'image de gradient, appariés selon le diamètre des cercles

recherchés. Malheureusement cette technique, qui est plus robuste au bruit que l'algorithme de Hough, implique d'avoir des images fortement contrastées, ce qui n'est pas le cas dans la réalité pour des images routières. Dans (de la Escalera, et al., 1997), ils détectent les coins dans les images (après seuillage sur la composante rouge, donc ils souffrent toujours du changement de conditions lumineuses) par un processus de convolution avec des masques spécifiques, afin de trouver les formes des panneaux souhaités. Cette méthode implique que tous les coins doivent être détectés et donc elle ne peut fonctionner dans le cas d'occlusion partielle des panneaux. De plus, comme le même masque est utilisé à la fois pour détecter les panneaux rectangulaires et circulaires, il est impossible de différentier un panneau carré d'un panneau rond, ce qui rend la méthode encore plus difficilement utilisable en pratique.

Il est à noter que certaines approches essayent de combiner les deux techniques (détection par couleur et par contours de forme) soit de façon parallèle (Fang, et al., 2003), soit successive. Cela revient à effectuer la seconde méthode uniquement sur les régions d'intérêts trouvées par la première (de la Escalera, et al., 1997) (Miura, et al., 2000) (Priese, et al., 1993). Ces techniques, en incluant l'approche par segmentation couleur, souffrent toujours essentiellement des changements de conditions lumineuses. Par exemple, (Priese, et al., 1993) utilise l'algorithme de croissance de régions pour trouver les cercles à partir d'une pré-segmentation des couleurs dans l'image.

Il est aussi possible d'utiliser les méthodes issues du domaine de l'apprentissage numérique (ou de l'intelligence artificielle) afin d'apprendre à un algorithme à trouver les régions d'une image contenant des panneaux. Ainsi, (Bahlmann, et al., 2005) utilise la récente technique du boosting dont l'idée est de combiner beaucoup de petits classificateurs faibles (c.à.d. qui arrivent à discriminer deux classes avec un taux légèrement supérieur à 50%) afin d'obtenir un classificateur fort (c.à.d. robuste). Dans cette technique, à l'issue de la phase d'apprentissage, les 4 meilleurs classificateurs induits travaillent sur le canal rouge et le 5$^{ème}$ sur l'image en niveau de gris. Cet algorithme souffre donc (à moindre mesure) des changements de variations lumineuses. L'algorithme AdaBoost est utilisé dans (Escalera, et al., 2004) mais cette fois-ci seulement sur des images en niveau de gris. Des réseaux de neurones, de type perceptron multicouche (PMC) sont aussi utilisés dans (Fang, et al., 2003) sur les gradients des images et sur les régions rouges (dans l'espace de couleur HSI), et de type cellular neural network dans (Adorni, et al., 1996). Cependant, apprendre à reconnaître des panneaux dans une image via des réseaux de neurones implique d'avoir un modèle géométrique sur les caractéristiques de ces panneaux (par exemple la taille des anneaux rouges), or celles-ci varient selon la distance des panneaux, ce qui rend ces techniques très difficile à utiliser sur ce type de problème. On peut aussi se demander si les modèles mathématiques induits par ces algorithmes d'apprentissage, lorsqu'ils sont utilisés sur les gradients des images, ne réinventent pas tout bonnement l'algorithme de Hough.

Enfin, certaines approches utilisent des connaissances a priori sur la géométrie de la route, par exemple en supposant que la route est à peu près droite, cela permet de supprimer une large partie de l'image pour l'analyse et la détection des panneaux afin d'accélérer le processus (Hsu, et al., 2001). Cela dit cette supposition se révèle fausse dans les courbes, ou lors de passage de ralentisseurs de type dos d'âne. Une approche plus sophistiquée est de détecter la route et/ou le ciel afin d'éliminer de larges régions. Par exemple (Piccioli, et al., 1996) suggère que les grandes régions

uniformes dans l'image représentent le ciel et la route, et donc de ne rechercher les panneaux qu'entre ces deux zones. Malheureusement, cette approche ne peut fonctionner en environnement urbain où des immeubles sont présents et dans les environnements où les routes sont bordées d'arbres. De plus, n'effectuer la détection que sur les bords de route ne permet pas de détecter les panneaux temporaires (lors de travaux par exemple) posés à même le sol. Une autre idée émise par (Fang, et al., 2003) est de ne rechercher les nouveaux panneaux que près du point de fuite dans l'image. Ceci est vrai, les nouveaux objets apparaissant près de ce point, cela dit, ils seront tellement petits dans l'image qu'ils seront tout bonnement indétectables.

### 1.2.2. Reconnaissance

Cette étape consiste à reconnaître, dans les zones d'intérêts (ROI) de l'image trouvées précédemment, quels sont effectivement les panneaux si panneau il y a, ou à supprimer les faux positifs (des ROI où aucun panneau n'est en fait présent).

Une grande majorité des travaux (Piccioli, et al., 1996) (Barnes, et al., 2004) (Guibert, 1995) (Seitz, et al., 1991) (Miura, et al., 2000) (Escalera, et al., 2004) se contentent d'effectuer le processus de corrélation croisée, l'algorithme le plus simple à mettre en œuvre, qui sera ici plus rapide car les zones à couvrir sont bien moindres que la taille de l'image originale. Cependant, la taille variable des panneaux à rechercher est toujours un problème, ce qui requiert d'avoir plusieurs masques à différentes échelles. Cela dit, si la première étape retourne des ROI plus ou moins bien cadrés sur les panneaux, la taille du masque à appliquer s'induit naturellement (Guibert, 1995).

D'après (Liu, et al., 2008) qui ont testé différents algorithmes d'apprentissage bien connus du domaine (kNN, MLP, RBF, PC, SVM) pour la reconnaissances de caractères, il apparaît que, sur une base de chiffres manuscrits (donc un peu plus complexe à reconnaître que des chiffres typographiés), tous les algorithmes donnent des résultats satisfaisants (plus de 98.5% de taux de bonne reconnaissance), mis à part le processus de corrélation croisée (ce qui est confirmé par (Simard, et al., 2003)). Ainsi comme le souligne (Piccioli, et al., 1996), la phase de classification est beaucoup moins critique que celle de la détection et le choix de l'algorithme d'apprentissage à utiliser revient à choisir celui avec lequel on est le plus à l'aise à manipuler. Toutefois comme le font remarquer (Liu, et al., 2008) (Simard, et al., 2003), les Convolutional Neural Networks (CNN) se placent un cran au dessus dans la qualité de classification. En effet, la grande force de cette technique est qu'ils apprennent la topologie spatiale des caractères et ne voient donc plus chaque exemple comme juste un ensemble de pixels juxtaposés.

Dans le domaine de la reconnaissance de panneaux, les réseaux de neurones, dans leurs différents types de topologies, sont les plus utilisés parmi les algorithmes de plus « haut niveau ». Une carte de Kohonen est utilisée dans (Luo, et al., 1992), tandis que dans (Adorni, et al., 1996) un réseau de neurone cellulaire est utilisé, de type ART1 dans (Escalera, et al., 2003) et ART2 dans (Fang, et al., 2003), mais la topologie la plus couramment utilisée reste le perceptron multi couche (PMC, ou multi layer perceptron - MLP) (Aoyagi, et al., 1996) (de la Escalera, et al., 1997) (Kang, et al., 1994) (Garcia, et al., 2006) (Torresen, et al., 2004). L'entrée du PMC est une imagette normalisée (sa taille est fixe et son histogramme de niveaux de gris est normalisé), par exemple dans (Aoyagi, et al., 1996) les

images en entrée du réseau de neurone font 18x18 pixels de cotés. Une autre topologie largement utilisée sont les RBF, radial base neural network (Gavrila, 1999) (Janssen, et al., 1993), voire des classificateurs à base de noyaux laplaciens (Paclik, et al., 2000). Curieusement les SVM, bien qu'ayant eu un grand engouement scientifique ces dernières années faces aux réseaux de neurones, ne sont que très peu utilisés ; (Liu, et al., 2008) explique cela par le fait que cette technique est d'une grande complexité pour l'apprentissage.

L'un des problèmes majeurs de ces types d'apprentissage se situe dans le fait qu'ils nécessitent d'avoir une base de données d'exemples (i.e. la base d'apprentissage) assez conséquente afin d'obtenir de bons résultats. (Liu, et al., 2008) stipule qu'utiliser des classificateurs statistiques, tels que les mixtures modèle (ex : mixture de Gaussiennes comme dans (Bahlmann, et al., 2005)), permet une meilleur généralisation lors de l'utilisation de petites bases d'apprentissage, mais les modèles neuronaux les dépassent largement sur de plus vastes.

(Soetedjo, et al., 2005) note cette même nécessité de disposer d'une base de données assez conséquente. Ils développent donc une technique basée sur la corrélation d'histogramme, ce qui leur permet de n'avoir besoin que de quelques exemples dans leur base d'apprentissage. Comme deux images différentes peuvent avoir deux histogrammes identiques (cas bien connus en traitement d'image), ils partitionnent leurs images en sous parties circulaires concentriques et n'effectuent le processus de reconnaissance que sur les histogrammes de chacune de ces sous parties. L'inconvénient majeur de leur méthode résulte dans le fait que l'histogramme doit être calculé sur une image en niveau de gris invariant à la lumière et que cela n'existe pour ainsi dire pas.

La grande majorité des cas, voir tous mise à part (Torresen, et al., 2004) se contentent d'effectuer l'apprentissage et la reconnaissance directement sur les ROI détectées lors de la première étape, c'est-à-dire sur tout un panneau (plus ou moins bien cadré). Cela impose plusieurs problèmes, par exemple si la ROI est mal centrée, il faut que l'algorithme d'apprentissage soit invariant aux translations, à la taille,… De plus cela impose une plus grande quantité d'informations en surplus à traiter par l'algorithme d'apprentissage ce qui influe forcément sur la complexité et les résultats de cet algorithme. Dans le même genre d'idées, bien que les panneaux soient normalement placés perpendiculairement à la route et droits (i.e. les chiffres sont alignés sur une droite parallèle au sol), certains d'entres eux peuvent être mal installés ou avoir subi des dommages (accidents mineurs) et donc apparaître avec une rotation.

Afin de gérer les déformations, rotations,… il existe trois grandes techniques comme le note (Cecotti, et al., 2004). Il est possible de rajouter des images dans la base d'apprentissage en transformant les exemples déjà présents en leur rajoutant par exemple du bruit, en les déformants, en changeant leur luminance,… comme le font (de la Escalera, et al., 1997) (Simard, et al., 2003). Il est aussi possible de redresser les images comme dans (Escalera, et al., 2004) où ils détectent des ellipses et non des cercles dans l'image, ce qui leur permet de remettre à plat un panneau qui pourrait apparaître déformé dans l'image. Enfin, la dernière technique est d'effectuer la reconnaissance sur la transformée de Fourier des ROI, transformée qui est invariante au bruit, à la rotation, à la translation et à l'échelle des panneaux détectés, comme le fait (Kang, et al., 1994). A noter que d'après (Cecotti, et al., 2004), la transformée de Fourrier-Melin est plus performante, pour cette application, que la transformée simple de Fourrier. (Cecotti, et al., 2004) conclut sur ces trois techniques que la

première et la dernière (ajouter des exemples ou utiliser la transformée de Fourrier) donnent des bons résultats. Cependant, ajouter des exemples donne de meilleurs résultats, car la reconnaissance s'effectue sur toute l'imagette et donc toute l'information est conservée. A noter que si l'on ne se soucie que de la rotation, il est possible de passer en coordonnée polaire où la rotation n'est plus qu'une simple translation (Cecotti, et al., 2004). Une dernière technique pour gérer les rotations est d'effectuer la reconnaissance en utilisant la distance de Hausdorff (qui est invariante à la rotation) comme le fait (Wu, et al., 1999), cependant un Perceptron Multicouches – PMC (réseau de neurones) est quand même utilisé en aval pour les cas non reconnus, ce qui traduit une certaine faiblesse de cette technique.

Enfin, afin d'améliorer les performances de ces algorithmes, comme le stipule (Soetedjo, et al., 2005), il est possible de combiner différents algorithmes d'apprentissage / reconnaissance (technique du boosting), ce qui permet de mieux identifier et rejeter des exemples ambigus, et / ou aussi de générer un plus grand nombre d'exemples comme déjà stipulé dans le paragraphe précédent.

Un dernier problème qui survient lors de la reconnaissance est qu'il est presque impossible de différentier les panneaux lorsqu'ils sont loin, c'est-à-dire lorsqu'ils sont extrêmement petits dans l'image. Afin de surmonter ce problème, (Miura, et al., 2000) utilisent deux caméras, une conventionnelle ayant un grand angle avec laquelle ils effectuent la recherche des ROI (la première étape), puis ils utilisent une caméra téléobjectif afin d'obtenir une image en grande résolution de ces ROI lointaines. Ce qui amène un coût plus important pour équiper leur voiture, une plus grande fragilité de leur système (la caméra à téléobjectif étant motorisée afin de pouvoir observer n'importe quelle zone) et surtout la nécessité d'avoir un processus de suivi temporel robuste (la caméra motorisée se déplaçant lentement comparé à la vitesse de la voiture).

Tous les travaux actuels effectuent l'apprentissage sur l'ensemble du panneau et n'essayent pas de segmenter l'information pertinente (i.e. les chiffres pour un panneau de limitation de vitesse) à l'intérieur de ce panneau.

### 1.2.3. Segmentation de caractères

Bien qu'un seul travail (Torresen, et al., 2004) en détection et reconnaissance des panneaux de limitation de vitesse n'ait envisagé cette approche, segmenter et reconnaître chacun des caractères à l'intérieur des panneaux peut être une solution intéressante. C'est pourquoi sont présentées ici les techniques classiques de segmentation de caractère.

Comme stipulé ci-dessus et confirmé par (Casey, et al., 1996), la segmentation de caractères est la partie la plus importante dans un système d'OCR (optical character recognition), et est nécessaire afin d'améliorer le taux de reconnaissance des algorithmes. Cette partie est souvent cachée dans les tests par l'utilisation de base de données de caractères bien segmentés (comme dans (Liu, et al., 2008)).

Il existe trois grands types de méthode de segmentation de caractères (Casey, et al., 1996) (Casey, et al., 1995):

1. classique où la segmentation se fait par rapport à l'histogramme, ce qui peut poser de nombreux problèmes (par exemple quand les lettres s'entremêlent).
2. par reconnaissances successives de lettres via des fenêtres de taille variable que l'on fait glisser le long du texte, ce qui n'est pas envisageable pour un processus temps réel.
3. des méthodes holistiques, où la segmentation se fait mot par mot, ce qui implique, pour la reconnaissance, d'avoir un dictionnaire fini et assez restreint de mots.

Afin d'améliorer la méthode classique, il est possible d'effectuer une sur-segmentation et d'effectuer ensuite un processus de reconnaissance en regroupant les segments (ce qui revient à peu près à la seconde méthode). Il est aussi possible d'utiliser les méthodes de suivi de contour, par exemple celui de la goutte d'eau qui tombe (drop-fall) vers le haut ou vers le bas, voire combiner les deux en analysant la convexité du point de contact (Khan, 1998). Une autre idée intéressante de (Khan, 1998) pour segmenter un nombre est de compter le nombre de minima locaux afin d'en déduire le nombre de chiffres qui le composent. D'une manière générale, l'extraction de composantes connexes donne de meilleurs résultats que la segmentation sur l'histogramme (Casey, et al., 1996).

Enfin, si on connaît a priori la taille et/ou la position des caractères à segmenter, il est possible de forcer la segmentation (Casey, et al., 1996), comme c'est le cas par exemple pour la reconnaissance des plaques minéralogiques (Zhang, et al., 2003) et non celui des panneaux de limitations de vitesse (la taille variant selon les villes, les pays, selon qu'il s'agisse d'un panneau urbain ou autoroutier,…).

Cela dit, il apparaît qu'on ne puisse pas vraiment comparer les différentes méthodes de segmentation, chacune étant élaborée pour un système, une tâche précise (Casey, et al., 1996). En revanche, il faut toujours tenir compte du contexte.

C'est ainsi que (Torresen, et al., 2004) segmente les caractères en extrayant, après avoir converti les ROI sur 3 couleurs (rouge, blanc et noir), les composantes connexes de couleurs noires avant d'en effectuer la reconnaissance via un PMC. L'inconvénient de cette approche, comme déjà stipulé, est que lorsque les chiffres s'entremêlent, il n'est alors plus possible de les reconnaître.

### 1.2.4. Intégration temporelle

Comme le font si bien remarquer (Piccioli, et al., 1996) et (Fang, et al., 2003) la détection des panneaux signalétiques en ne considérant qu'une seule image à la fois (donc sans intégration temporelle) soulève trois problèmes :

1. Cela ne permet de prédire la position et la taille des panneaux pour réduire l'espace de recherche et le temps d'exécution.
2. Il est difficile de correctement détecter un panneau quand il y a une occlusion temporaire.
3. La véracité de la détection est dure à vérifier sur une image seulement.

C'est ainsi que quelques approches emploient un processus de fusion temporelle basée sur la détection image par image afin d'obtenir une meilleure détection globale. Ces approches peuvent être divisées en deux catégories.

1. Il est tout d'abord possible de ne se soucier que de la succession des détections à chaque image. Ainsi plus un panneau sera détecté de façon successive, meilleur sera son indice de

confiance dans sa reconnaissance globale. C'est ce que fait (Bahlmann, et al., 2005) par exemple.

2. La deuxième approche consiste, à partir des détections antérieures, à prédire la position des panneaux dans les futures images (tracking temporel). Ceci peut avoir lieu en supposant que le mouvement du véhicule est localement droit et ainsi prédire la position dans une image à partir de 2 prédictions précédentes (l'idée est de tracer la droite entre les 2 anciennes prédictions, la suivante sera sur cette droite à une distance régie par la vitesse du véhicule) comme le fait (Miura, et al., 2000). Il est possible d'améliorer ce processus en y incorporant un filtre de Kalman (Piccioli, et al., 1996) (Zadeh, et al., 1998) (Garcia, et al., 2006) (Fang, et al., 2003).

   Bien évidemment, ceux qui emploient cette seconde technique utilisent aussi tous la première.

Une autre technique pour gérer les cas d'occlusion partielle consiste à rajouter dans la base d'apprentissage des nouveaux exemples soumis à diverses translations et tronqués (Yang, et al., 2003). Il est aussi possible de repérer les cas d'occlusions partielles en trouvant les zones non rouges, blanches et noires dans l'image du panneau et d'utiliser cette information pour modifier dynamiquement les règles de l'algorithme de reconnaissance (Soetedjo, et al., 2005).

Enfin, certains élaborent des règles de plus haut niveau afin d'améliorer la robustesse de leur système de reconnaissance de panneaux de limitations de vitesse. Ainsi (Torresen, et al., 2004) établit que par exemple, après avoir reconnu une limitation à 80, qu'il n'est pas possible d'avoir directement une limitation à 10, mais il est obligatoire de passer par la vitesse limite de 60 (en Norvège tout au moins).

Pour résumer, un système embarqué de détection et de reconnaissance des panneaux de limitation de vitesse est basé sur 3 grands modules comme le rappelle la Figure 3.

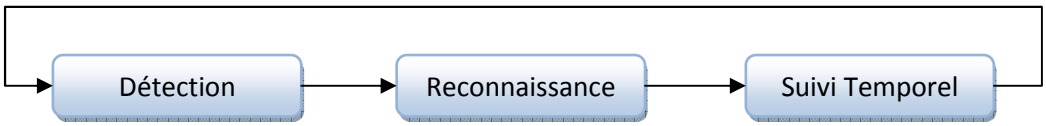



Figure 3 - Les 3 étapes de la détection des panneaux de limitation de vitesse

## 1.3. Détection et reconnaissance des panneaux

### 1.3.1. Conception générale

Le système développé doit pouvoir fonctionner sous n'importe quelle condition climatique (ensoleillé, par brouillard,…), lumineuse (de jour, de nuit, …) et quel que soit l'état du panneau (neuf avec des couleurs criantes et ancien avec des couleurs estompées). Il est donc de ce fait exclu

d'envisager d'utiliser un processus de segmentation par couleur en première étape (détection des zones d'intérêt), car, bien que rapide, cette méthode ne pourrait pas fonctionner dans tous les cas. Mais le système doit aussi pouvoir être exécuté en temps réel car embarqué au sein de véhicule pour aider les conducteurs (et donc avoir une réactivité accrue). L'enjeu principal de cette première étape de segmentation réside donc dans le développement d'un module basé sur la détection de contours qui soit robuste et rapide.

Pour l'étape de reconnaissance des zones d'intérêt, la question réside dans le fait de savoir sur quelle partie de l'image la reconnaissance se fera. Soit sur toute la zone d'intérêt, représentant donc tout le panneau, ou, pour le cas spécifique des panneaux de limitation de vitesse, essayer de n'effectuer la reconnaissance que sur chacun des chiffres à l'intérieur du panneau. Les deux solutions ont chacune des avantages et des inconvénients :

- La reconnaissance de toute la zone d'intérêt peut être sujette à une détection peu précise (c.à.d. une zone d'intérêt mal calée sur le panneau) qui pourrait fausser les résultats de l'algorithme de reconnaissance, mais elle n'entraine pas de traitement particulier en plus qui pourrait alourdir l'exécution du système (et donc son temps de traitement).
- La reconnaissance des chiffres implique de pouvoir extraire chaque caractère à l'intérieur du panneau détecté ce qui peut induire plus d'erreur (de segmentation) et un traitement plus lourd pour une exécution embarquée devant fonctionner en temps réel. En revanche, l'algorithme de classification n'aurait plus qu'à reconnaître qu'une dizaine de classes (chiffres de 0 à 9) au lieu d'un plus grand nombre de panneau de limitation de vitesse (de 10km/h à 130 km/h en France). De plus, la segmentation de caractère peut donner des zones plus précises à reconnaître. Il en résulte que le résultat de la classification basée sur cette méthode doit être meilleur.

De plus, en pratique, il n'existe pas de norme stricte nationale et encore moins européenne sur l'aspect physique des panneaux de limitation de vitesse. Ils sont certes tous cerclés de rouge avec le nombre en noir au milieu, mais ni la taille du panneau, ni la typographie des chiffres ne sont imposées. Il en résulte une grande variabilité des panneaux au sein d'un même pays (voire d'une même ville) et entre pays différents (Figure 4).


**Figure 4 - Illustration de la variabilité de l'aspect des panneaux de limitation de vitesse : En gris des panneaux allemands (à gauche de chaque paire) comparés à leur équivalent français - En couleur à droite 2 variantes d'un même panneau français.**

Le système développé, qui vise à être le plus robuste possible, est donc naturellement basé sur la segmentation de caractères afin de palier aux problèmes de variabilité des panneaux et de réduire le nombre de classes à reconnaitre. Curieusement, dans la littérature, seul les travaux de (Torresen, et

al., 2004) utilisent une technique similaire, tous les autres travaux effectuent une reconnaissance globale. L'utilisation de cette technique pourrait nous aider à adapter facilement nos travaux pour la reconnaissance des panneaux de limitation de vitesse américains qui « souffrent » eux d'une grande variabilité de leur contenu et de leur taille (Figure 5).

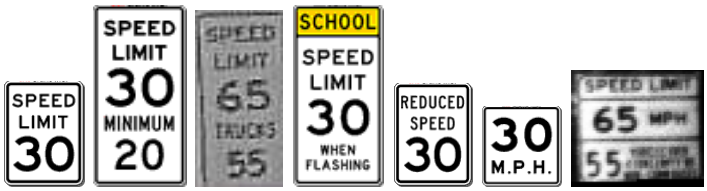



**Figure 5 - Illustration de quelques uns des pannaux de limitation de vitesse américains qui prouvent la complexité d'utiliser un processus de reconnaissance globale des panneaux dans ce pays.**

Le processus de segmentation qui fonctionne le mieux, est d'après la littérature l'extraction des composantes connexes, qui sera donc utilisée dans le système. Le choix de l'algorithme de reconnaissance d'image / de classification (SVM, RN,…) est beaucoup moins critique, ces algorithmes donnant de nos jours des résultats plus que satisfaisants. De plus, par le choix de cette méthode, il n'est pas exclu de pouvoir utiliser des algorithmes d'OCR (reconnaissance de caractère) issus par exemple de la reconnaissance des documents scannés, ayant prouvé leur bon fonctionnement depuis des années.

Afin de se rendre compte de la complexité de la tâche, une première version du système est développée en utilisant les algorithmes bien connus issus du traitement d'image. Son fonctionnement, utilisant une caméra monochromatique embarquée dans le véhicule, est schématisé sur la Figure 6.

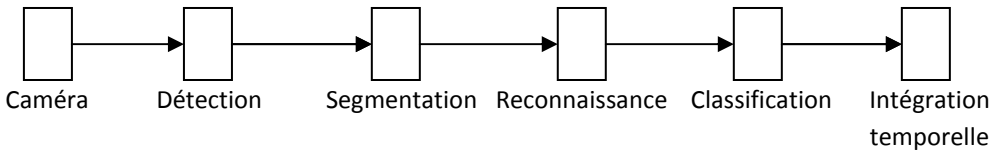



**Figure 6 - Fonctionnement général du système**

1. Un flux d'image en niveau de gris est acquis via une caméra embarquée (*Figure 7*).

2. De ce flux d'images sont extraits les panneaux potentiels via l'algorithme de Hough (i.e. sont détectés tous les cercles dans chacune des images) pour les panneaux circulaires de type européen (*Figure 8*).

3. Chaque imagette issue d'un cercle subit une normalisation d'histogramme suivie d'un seuillage binaire local adaptatif (*Figure 9*).

4. Une segmentation des caractères, via une extraction de composantes connexes (blobs) est effectuée sur chacune de ces imagettes en noir et blanc (*Figure 10*).

5. Les composantes respectant certains critères géométriques (ex : ratio hauteur/largeur) sont analysées via un réseau de neurones (de type perceptron multicouche) afin d'en reconnaître le chiffre (*Figure 11*).

   La topologie du réseau de neurone a été trouvée de façon empirique (Tableau 1) : il apparaît qu'avec 20 neurones sur la couche cachée, l'apprentissage est le meilleur (i.e. aussi bon qu'avec plus de neurones, et meilleur qu'avec moins). Pour des raisons historiques, la taille de l'entrée est de 64 neurones pour reconnaître des images de 8x8 pixels (une première version de la base de chiffre existait déjà). Le réseau de neurones a été entrainé sur cette base de données agrémentée de chiffres extraits de nos enregistrements (qui constituent donc de vrais exemples et non des exemples artificiels) dont 2772 images de chiffres (provenant de panneaux français, allemands et italiens) et 2790 exemples négatifs (des non-chiffres). L'outil utilisé (*Levis*) pour l'entrainement et le choix de la topologie du réseau de neurone a été développé spécifiquement et est décrit au chapitre 5.

6. Le flux de chiffres reconnus est classifié (i.e. assemblé et validé 1 + 1 + 0 → 110 ou invalidé 1 + 9 + 0 → 190, limitation qui n'existe pas) puis intégré temporellement afin d'extraire un niveau de confiance géométrique et temporel de la détection pour valider ou invalider la détection d'un panneau (*Figure 12*).

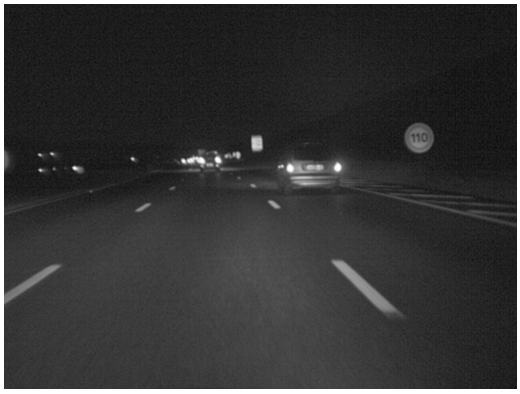


Figure 7 - Image originale

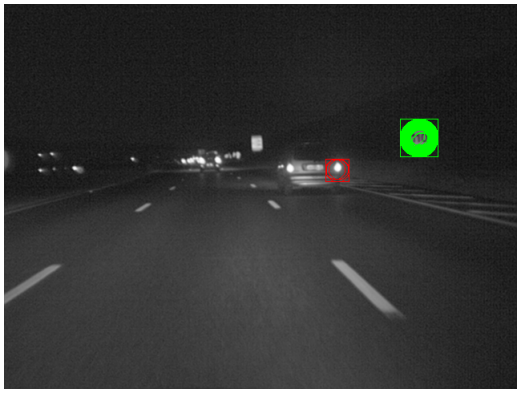


Figure 8 - Détection de cercles

Figure 9 - Image binaire

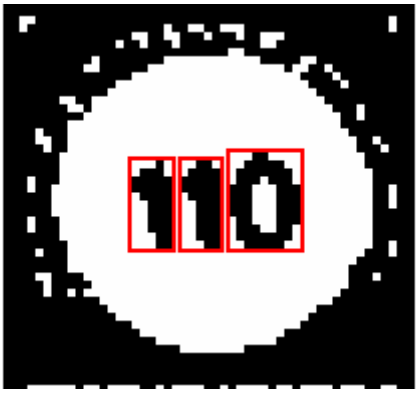
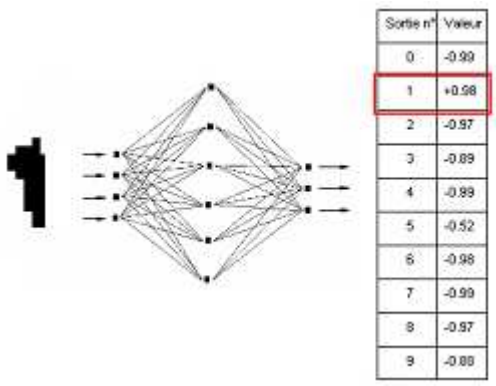


Figure 10 - Extraction des blobs



| Sortie n° | Valeur |
|---|---|
| 0 | -0.99 |
| 1 | +0.98 |
| 2 | -0.97 |
| 3 | -0.89 |
| 4 | -0.89 |
| 5 | -0.52 |
| 6 | -0.98 |
| 7 | -0.99 |
| 8 | -0.97 |
| 9 | -0.88 |

Figure 11 - Reconnaissance des blobs

Figure 12 - Résultat

| Taille de la couche cachée | Taux de bonne classification de la base d'apprentissage | Taux de bonne classification de la base de test |
|---|---|---|
| 5 | 74% | 72% |
| 10 | 95% | 93% |
| 15 | 96% | 95% |
| 20 | 97% | 96% |
| 25 | 97% | 96% |
| 30 | 97% | 96% |
| 35 | 97% | 96% |

Tableau 1 - Taux de bonne classification du réseau de neurone sur la base de chiffre

Quelques commentaires sur cette première version s'imposent. Le système fonctionne globalement bien et est très prometteur, mais il n'est, pour l'instant, pas assez efficace pour être utilisé tel quel dans un système d'aide à la conduite automobile. En effet il souffre de certains problèmes. Ainsi, comme listé dans l'état de l'art, la segmentation de caractère par extraction de composante connexe est robuste, même quand les panneaux sont inclinés et non droits (Figure 13), mais inefficace quand les chiffres s'entremêlent (ce qui arrive quand le panneau à reconnaître est loin du véhicule). La détection des cercles est, elle aussi, efficace  mais l'algorithme de Hough est inexploitable en temps réel et un compromis a dû être trouvé pour le faire fonctionner à une fréquence raisonnable. C'est ainsi que cet algorithme n'est appliqué que sur des images en demi-résolution, ce qui réduit forcément la distance à partir de laquelle un panneau peut commencer à être détecté. Par conséquence, dans les cas où le panneau est toujours trop petit dans l'image, ce qui arrive quand il est physiquement trop petit et / ou quand il est toujours loin du véhicule (par exemple sur une route à 3 voies, quand le panneau est à l'opposé du véhicule), le système ne peut le détecter.

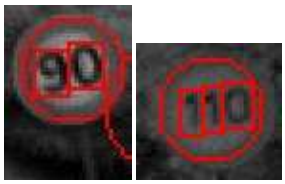



Figure 13 - Illustration de la robustesse de la segmentation par extraction de composantes connexe

Il apparaît donc que les choix initiaux soient en corrélation avec l'état de l'art et que le travail mis en place ne soit pas tombé dans les pièges mis en exergue par la facilité, cependant le système souffre de problèmes inhérents aux choix qui ont été faits. Les améliorations apportées afin de rendre le système efficace et exploitable en temps réel sont décrites dans les sections suivantes.

### 1.3.2. Segmentation globale des chiffres

Dans la majorité des cas, la segmentation des caractères via extraction des composantes connexes fonctionne correctement. Cependant, pour plusieurs raisons, il arrive que les chiffres composant la limitation de vitesse sur l'image vue par la caméra s'entremêlent et ne forme donc plus qu'une seule grosse et même composante connexe non reconnaissable par le système.

Ces raisons sont :

1. Quand les panneaux sont détectés d'assez loin, la discrétisation effectuée par le processus d'acquisition numérique fait que la séparation des chiffres n'est plus visible.

2. Quand les panneaux sont naturellement petits, pour les mêmes raisons les chiffres s'entremêlent.

3. De nuit, la caméra est éblouie par la forte réflexion des phares sur les panneaux signalétiques ce qui a pour conséquence de rendre les images plus ou moins floues.

Une solution triviale serait d'utiliser un autre réseau de neurones pour apprendre à classifier ce genre de formes. Cependant, afin d'avoir un algorithme de reconnaissance robuste, il faut disposer de nombreux exemples. Or les formes des composantes connexes trouvées sont trop variées pour pouvoir créer une base d'exemples importante. Par exemple, *Figure 14*, à gauche ne sera extrait que le '100', tandis qu'à droite, les chiffres touchent aussi le bord du panneau, ainsi la composante connexe est formée du '100' et du bord du panneau. Parfois aussi seuls deux parmi trois chiffres sont « collés », ce qui augmente le nombre de cas de figure et rendrait très complexe une approche de ce type (Figure 15).

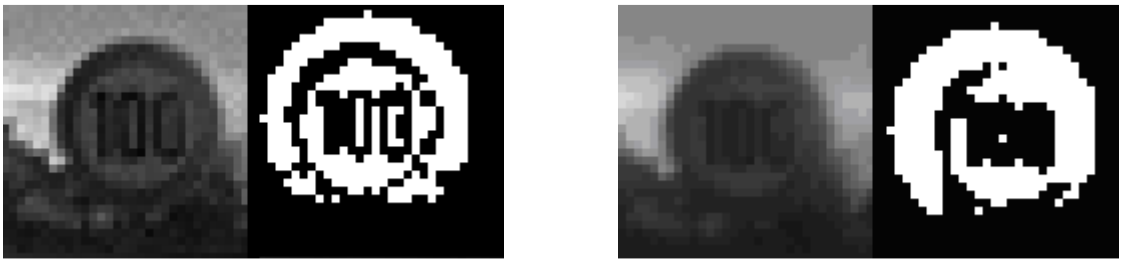


Figure 14 - Exemple d'images binaire où la segmentation ne fonctionne pas


Figure 15 - Exemple de cas où la segmentation n'est pas correcte

Il est donc nécessaire de mettre en place un processus de segmentation spécifique à notre problème.

Typiquement, dans le domaine du traitement d'image, la première idée serait d'effectuer une segmentation selon l'histogramme de nos images. Chaque chiffre, représentant un pic dans l'histogramme, il paraîtrait théoriquement facile de segmenter les chiffres qui s'entremêleraient. Cependant, avant la reconnaissance, il nous est impossible de savoir si notre image représente un panneau à trois chiffres (trois pics dans l'histogramme ?), ou à deux chiffres (donc deux pics dans l'histogramme ?). De plus, l'image binaire, dépend d'un nombre important de facteurs, comme par exemple la position du cercle détecté, l'environnement sur lequel le panneau se sur-imprime dans l'image… qui modifient quels pixels seront noirs ou blancs dans l'image binaire et donc qui modifient l'histogramme (et le nombre de pics présents dans celui-ci) comme le montrent les Figure 16, Figure 17 et Figure 18.

Il existe, comme nous l'avons vu plus haut, de nombreuses méthodes dans le domaine de la segmentation de caractères pour nous aider. Cependant, il faut proscrire les méthodes de type sur-segmentation, qui pourraient faire apparaître de fausses détections (par exemple segmenter les barres des '0' des panneaux allemands pourrait faire apparaître des '1') et de type segmentation par reconnaissance, nous avons déjà vu que cela revenait à peu près à la méthode de sur segmentation (via les fenêtres glissantes de taille variable). La dernière technique est d'effectuer une segmentation globale (i.e. extraire tout le nombre du panneau sans sa bordure), ce qui répondrait au cas de la *Figure 14*.

Le problème revient donc, pour une segmentation de type global, à trouver les limites horizontales (haute et basse) et verticales (gauche et droite) du nombre dans l'image représentant la limitation de vitesse dans le panneau.

Nous avons déjà vu qu'il est impossible de se baser sur l'histogramme pour trouver les limites verticales. La même question se pose pour les limites horizontales. Il apparaît que là aussi, comme le montre la Figure 18, il est impossible de se baser sur la technique du comptage de pics dans l'histogramme.


**Figure 16 - Impossibilité de compter le nombre de pics verticaux dans l'histogramme pour segmenter les chiffres (panneau allemand)**

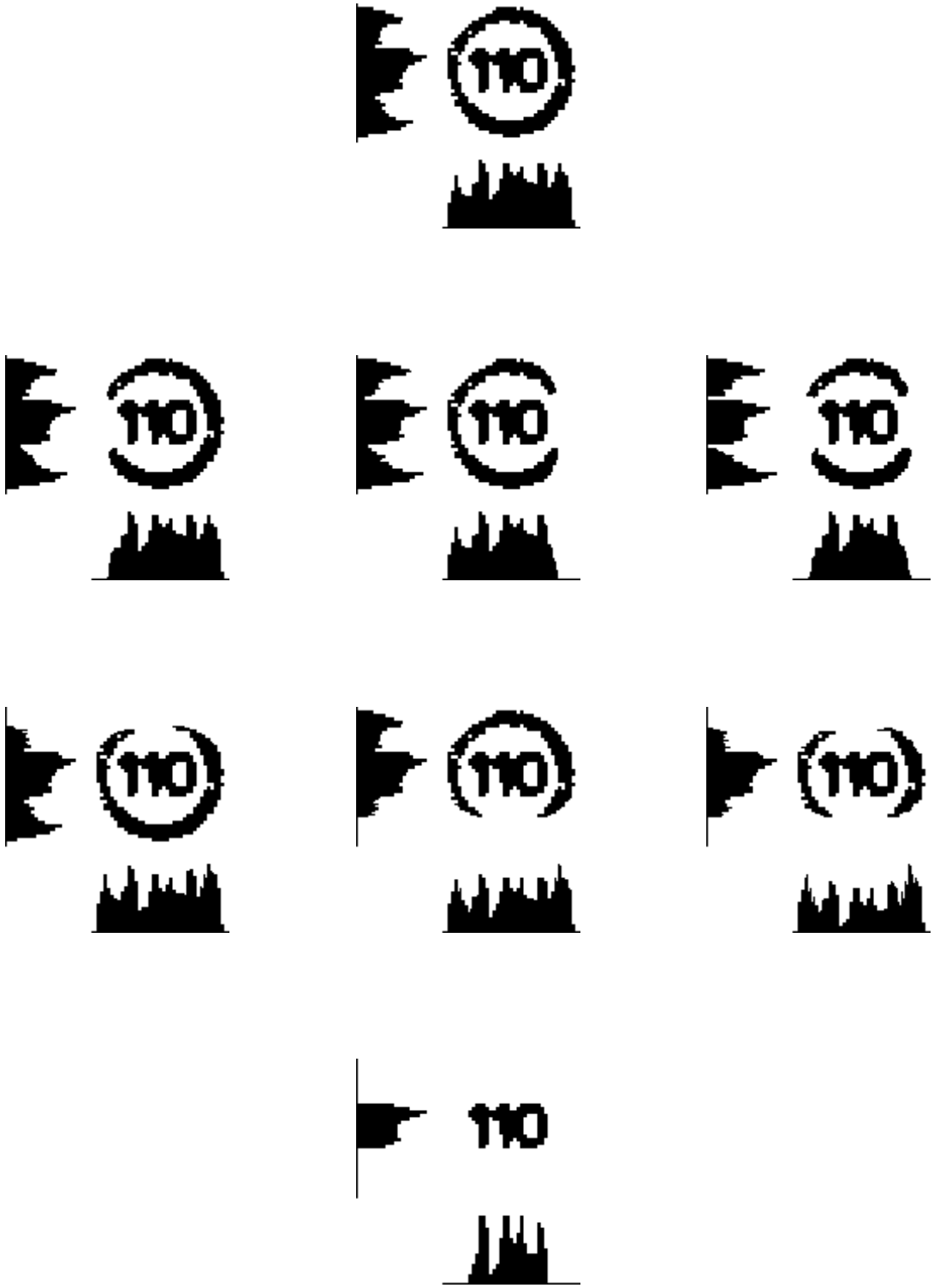



**Figure 17 - Impossibilité de compter le nombre de pics verticaux dans l'histogramme pour segmenter les chiffres (panneau français)**

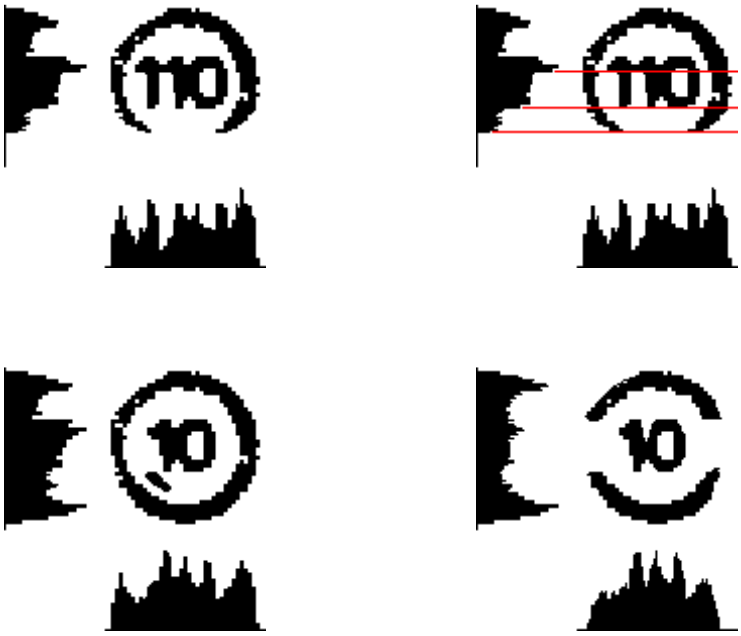



Figure 18 - Impossibilité de compter le nombre de pics horizontaux dans l'histogramme pour segmenter le nombre

La solution proposée pour trouver ces limites est l'algorithme *Global Segmentation for Speed-Limit Sign – GS-SLS* exposé ci-après. A partir de l'image binaire (Figure 19), et à partir d'une petite portion horizontale au milieu de cette image (Figure 20), l'algorithme consiste à parcourir l'image de telle sorte que l'on privilégie d'aller d'abord vers le centre de l'image (donc à droite quand on est à gauche et vice versa) puis vers le bas si on cherche la limite inférieure (ou vers le haut pour la limite supérieure) avec l'impossibilité de revenir sur ses pas et d'aller vers le haut quand on cherche la limite basse et vice versa (Figure 21). L'image binaire est donc divisée en quatre zones sur chacune desquelles on effectue le parcours décrit précédemment (Figure 22). Les limites horizontales sont les maxima trouvés par ces différents parcours (Figure 23).

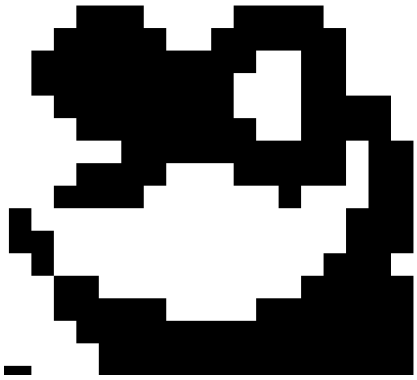


Figure 19 - Image binaire

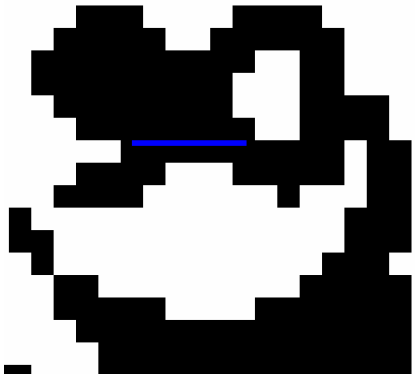


Figure 20 - Ligne de départ

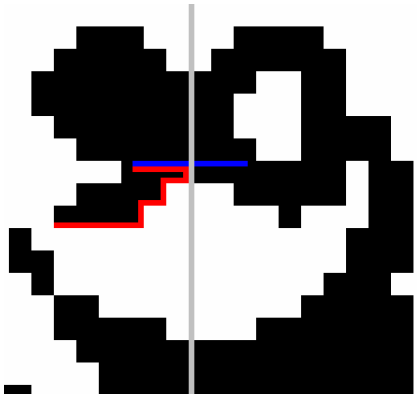


Figure 21 - Parcours gauche inférieur

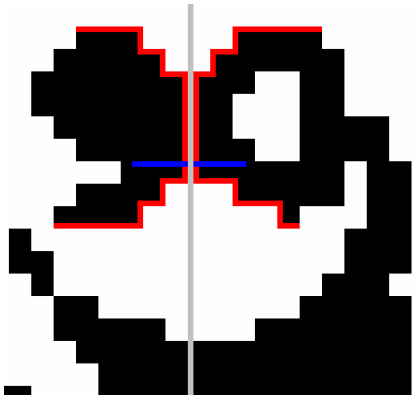


Figure 22 - Les 4 parcours

Ce parcours, pixels par pixels via un voisinage 3-connexes (4 voisins moins 1) est robuste au bruit. En effet, si l'un des points de départ est en fait un pixel de bruit (un pixel ou petit groupe de pixels noirs au milieu du fond blanc du panneau), la propagation à partir de cet endroit n'irait pas bien loin. De plus, le fait d'utiliser un voisinage 3-connexe, empêche le parcours de s'étendre dans le bruit vers le haut (ou vers le bas) à l'extérieur du nombre. En revanche, pour éviter au parcours de s'étendre le long de la bordure du panneau, une règle mise en place en place consiste à empêcher la propagation vers l'extérieur si on a dépassé une certaine limite (gauche ou droite), typiquement un pourcentage de la taille du panneau.

Il ne reste plus qu'à trouver les limites verticales du nombre dans le panneau. Pour cela, à partir du milieu de l'image, l'algorithme consiste à parcourir l'image colonne par colonne vers la droite ou vers la gauche (Figure 24, Figure 25, Figure 26 et Figure 27). Ce parcours s'arrête lorsque que la colonne de pixels noirs dépasse l'une des limites horizontales (Figure 28 et Figure 29).

Cet algorithme a été testé sur de nombreux exemples (vidéos) sous diverses conditions (de jour, de nuit, …) et semble fonctionner correctement (Figure 30).

Figure 23 - Limites horizontales

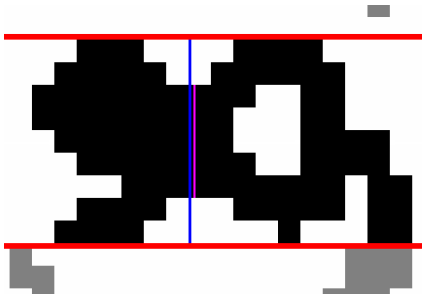


Figure 24 - Parcours par colonne

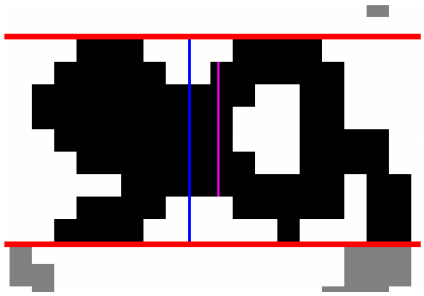


Figure 25 - Parcours par colonne

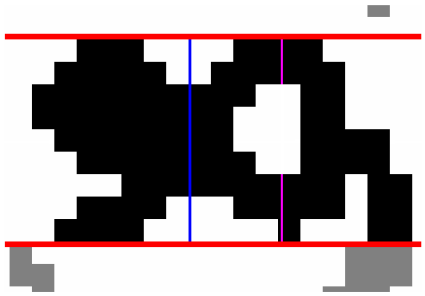


Figure 26 - Parcours par colonne

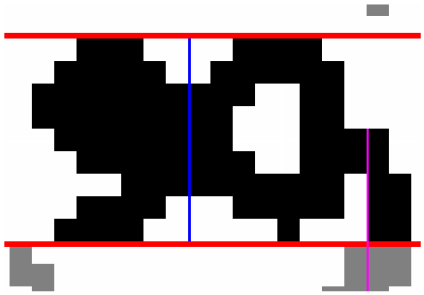


Figure 27 - Parcours par colonne

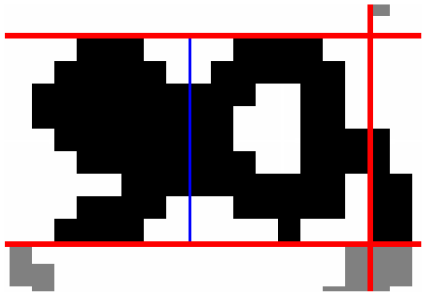


Figure 28 - Limite droite trouvée

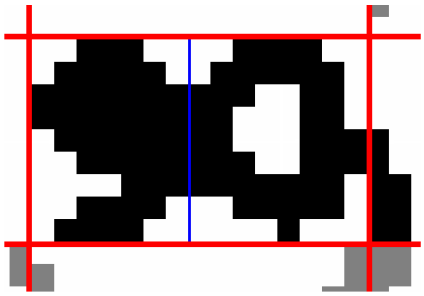


Figure 29 – Résultat

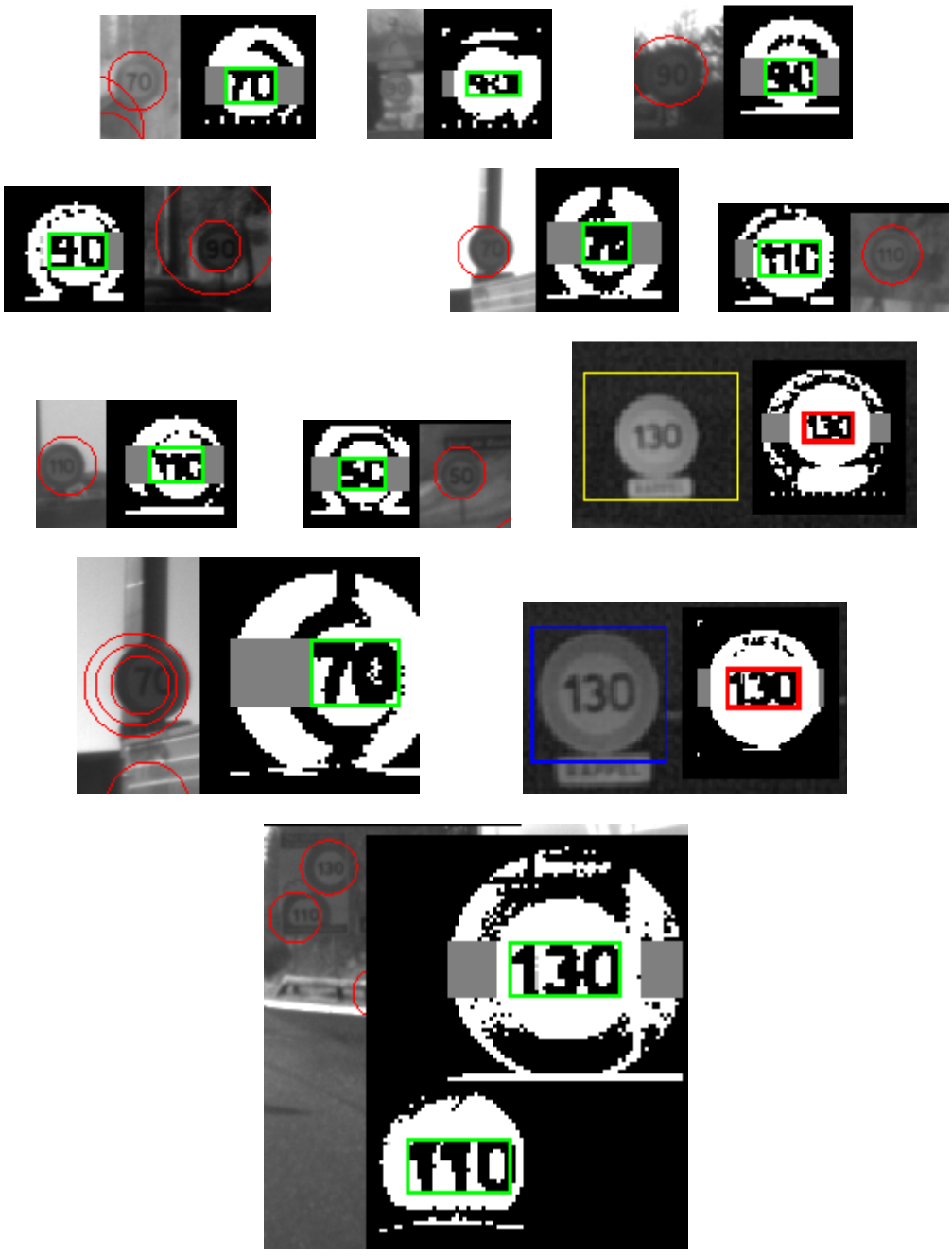


Figure 30 - Résultats de l'algorithme GS-SLS de segmentation sur divers cas

La segmentation des caractères du panneau par l'extraction des composantes connexes échouait car plusieurs chiffres se retrouvaient dans la même composante connexe. Cet effet indésirable est dans la majorité des cas dû au fait que le seuil calculé dans l'algorithme de seuillage adaptatif (pour rendre l'image binaire) est faussé car trop d'information est présente quand on considère le panneau en entier (i.e. la détection de cercle effectuée) : il y a dans cette zone des pixels représentant le cercle rouge, des pixels représentant l'extérieur du panneau (surtout si le cercle détecté n'est pas bien centré sur le panneau),… Il est maintenant possible d'effectuer le seuillage uniquement sur la zone trouvée par le nouvel algorithme de segmentation globale du nombre. Il en résulte que chaque chiffre est correctement représenté par une composante connexe (Figure 31). Néanmoins, un algorithme de segmentation basé sur l'histogramme a quand même été mis en place sur ces nouveaux résultats. En effet, il arrive que, sur cette nouvelle image binaire, les chiffres se retrouvent quand même liés par un ou deux pixels entre eux, ce qui est facilement segmentable.

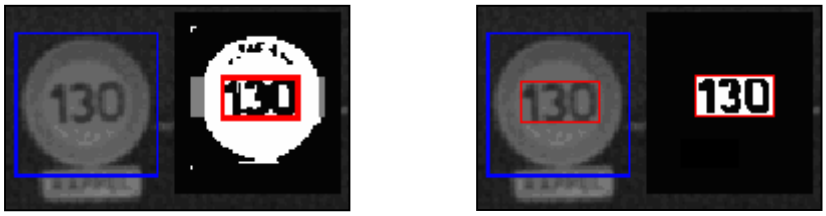



Figure 31 - Exemple d'amélioration apporté par la segmentation globale : Ancien résultat (à gauche) – Nouveau résultat (à droite)

Ce nouvel algorithme de segmentation, spécifique à l'extraction des nombres dans les panneaux de limitation de vitesse, impose quelques restrictions pour son utilisation. Dû au fait que la zone de départ soit une ligne horizontale centrée sur le panneau, il est primordial que cette zone contienne des portions des chiffres et aussi et surtout qu'elle ne chevauche pas la bordure du panneau (sinon la propagation se fera sur cette bordure et à l'extérieur des chiffres) comme le montre la Figure 32. Il faut donc que la pré-détection des cercles soit assez centrée sur le panneau et pas trop grande. Ce qui peut sembler assez contraignant face aux contraintes, plus laxistes de l'extraction des blobs.

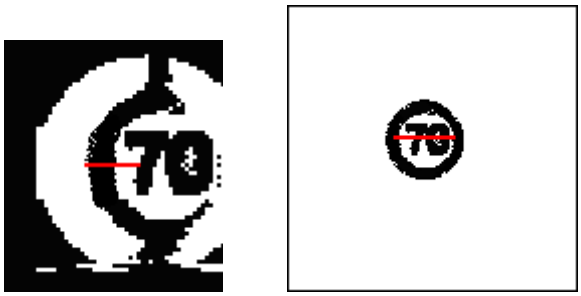



Figure 32 - Deux cas où la segmentation holistique ne fonctionne pas

### 1.3.3. Détection de cercles

Dans la première version de l'algorithme, afin de rendre l'algorithme de Hough pour la détection des cercles plus rapide et utilisable en temps réel, il n'est pas appliqué sur l'image dans sa taille originale, mais sur une version réduite (en demi-résolution) du flux vidéo. De plus, seuls les cercles dont les rayons qui sont compris dans un certain intervalle (évalué empiriquement de façon à ce que les cercles détectés restent assez grands / reconnaissables) sont recherchés.

Cependant, réduire la taille de l'image engendre évidemment une perte d'information rendant l'algorithme peut exploitable dans certains cas et notamment dans les cas à faible contraste et quand les panneaux à détecter sont loin / petits) comme on peut le voir Figure 33.

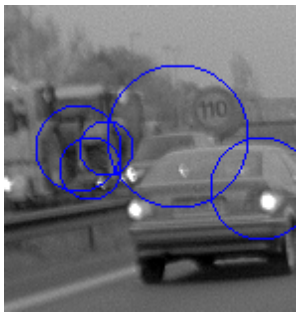


Figure 33 - Détection de cercle sur une image réduite

Même si cela ne semble pas poser beaucoup de problème dû au fait de l'extraction des composantes connexes à l'intérieur de chaque détection, il est nécessaire d'avoir une détection plus robuste (i.e. mieux cadré sur le panneau), afin d'améliorer la reconnaissance des panneaux, par exemple par l'utilisation de l'algorithme de segmentation décrit précédemment ou pour effectuer une reconnaissance globale du panneau (cf. les fins de limitation de vitesse comme on le verra plus tard).

***SUITE DE LA PAGE TEMPORAIREMENT MASQUEE***

# *PAGE TEMPORAIREMENT MASQUEE*

# *PAGE TEMPORAIREMENT MASQUEE*

# *PAGE TEMPORAIREMENT MASQUEE*

# *PAGE TEMPORAIREMENT MASQUEE*

# *PAGE TEMPORAIREMENT MASQUEE*

# *PAGE TEMPORAIREMENT MASQUEE*

# *PAGE TEMPORAIREMENT MASQUEE*

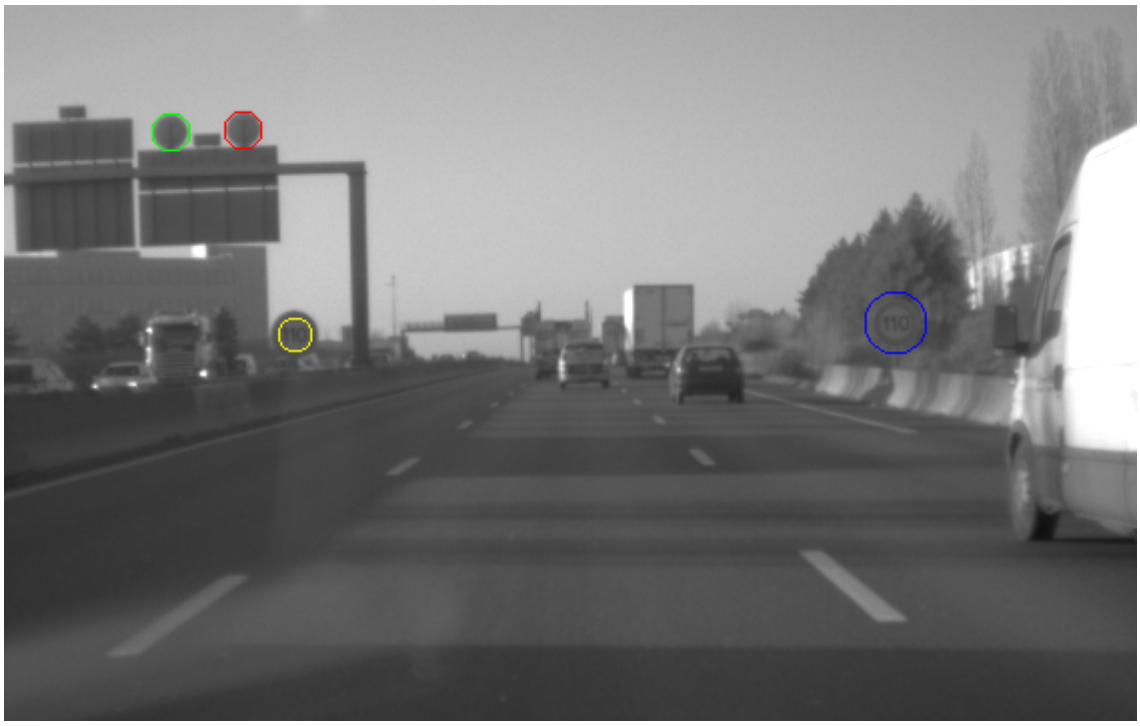


Figure 48 - Exemple de résultat de la nouvelle approche de l'algorithme de Hough

### 1.3.3. Intégration temporelle

La phase d'intégration temporelle initialement développée dans le système ne sert qu'à modifier l'indice de confiance globale que l'on a dans la détection et dans la reconnaissance d'un panneau. Un panneau réel sera en effet détecté dans plusieurs images consécutives, il suffit donc d'augmenter l'indice de confiance du panneau à chaque détection (et à l'inverse l'abaisser à chaque non détection), le panneau est validé quand son indice de confiance dépasse un seuil prédéfini. L'appariement d'un même panneau sur 2 images consécutives, dans le système initial, se fait en associant les panneaux les plus proches et de même limitation de vitesse.

Cependant, afin de pouvoir apparier correctement deux détections du même panneau mais à deux instants différents, donc à deux places différentes dans l'image (dû au mouvement du véhicule

embarquant le système) et afin de changer correctement l'indice de confiance et aussi de ne pas mélanger les détections de plusieurs panneaux différents, il est nécessaire de prédire la position future des panneaux.

Une première méthode mise en place consiste, en supposant que la trajectoire des panneaux est linéaire, à utiliser la méthode des moindres carrés, appliquée aux centres des panneaux sur plusieurs image, pour réaliser l'interpolation, i.e. trouver les coefficients a et b de la droite y = ax + b estimant la trajectoire d'un panneau (Équation 1).

$$y = ax + b \text{ avec } \begin{cases} a = \dfrac{n\sum_i x_i y_i - \sum_i x_i \sum_i y_i}{n\sum_i x_i^2 - \left(\sum_i x_i\right)^2} \\ b = \dfrac{\sum_i x_i \sum_i y_i - \sum_i x_i \sum_i x_i y_i}{n\sum_i x_i^2 - \left(\sum_i x_i\right)^2} \end{cases}$$

Équation 1 - Interpolation par moindres carrés

Cette méthode, comme le montre la Figure 49 fonctionne correctement dans le processus d'appariement des panneaux.

Figure 49 – Résultat du suivi temporel par moindres carrés d'un panneau

Il est possible ici d'améliorer le système en gardant la même logique. En effet, l'intégration temporelle n'est réalisée que pour modifier l'indice de confiance que l'on a dans la détection d'un panneau.

Cependant il arrive qu'un panneau ne soit plus détecté pendant une ou plusieurs images (ce qui ne pose pas de problème au processus actuel qui met à jour les coordonnées des panneaux en mémoire

via la méthode des moindres carrés (Équation 1). Il apparaît qu'un panneau peut ne plus être détecté pour seulement deux raisons :

1. soit la détection du contour (cercle) du panneau dans l'image a échoué, et donc notre système, à l'état actuel, n'a pas pu essayer de segmenter les chiffres puis reconnaître la limitation de vitesse.

2. soit la segmentation ou la reconnaissance a échoué

Après étude, beaucoup de cas sont dus à la première proposition. L'idée est donc ici de forcer le processus de recherche / segmentation de caractères puis de leur reconnaissance à des endroits prédits selon la position des anciens panneaux détectés (Figure 50).

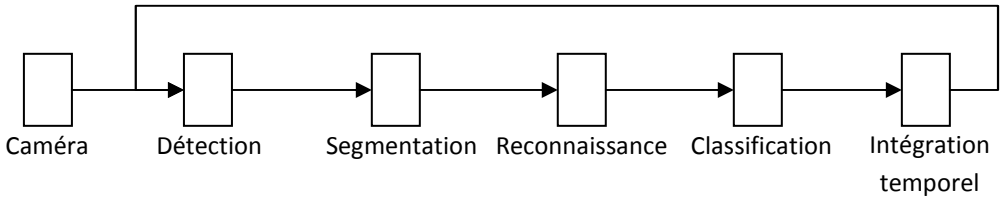



Figure 50 - Fonctionnement général du système avec suivi spatio-temporel

L'interpolation linéaire par la méthode des moindres carrés est certes très efficace, mais elle ne prend pas en compte le modèle de projection de l'image du monde réel dans la caméra.

En simplifiant le modèle physique de la caméra par le modèle sténopé (appelé aussi modèle *pin-hole* dans la littérature anglo-saxonne) il est possible de prédire la position et la taille d'un panneau dans l'image (Équations E7 et E8).

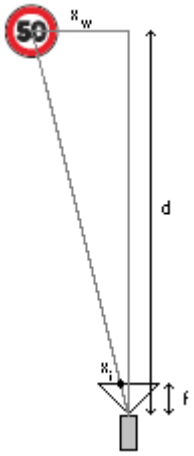



Figure 51 - Modèle sténopé

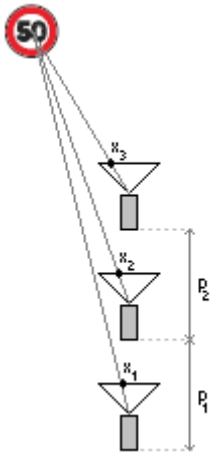



Figure 52 - Suivi spatio-temporel

Le modèle sténopé est régi par les équations de Thales, ainsi, Figure 51 $\frac{f}{x_i} = \frac{d}{x_w}$ où f est la focale de la caméra, d la distance du panneau à la caméra, $x_i$ la position du centre du panneau dans l'image et $x_w$ la position du panneau par rapport à la caméra.

Il est alors possible, d'après la Figure 52, d'écrire les équations suivantes, en supposant une trajectoire rectiligne de la voiture (i.e. qu'elle ne tourne pas) :

| | | | |
|---|---|---|---|
| $\frac{f}{x_1} = \frac{d}{x_w}$ | $\frac{f}{x_2} = \frac{d - p_1}{x_w}$ | $\frac{f}{x_3} = \frac{d - (p_1 + p_2)}{x_w}$ | E1 |
| $fx_w = dx_1$ | $fx_w = (d - p_1)x_2$ | $x_3 = \frac{fx_w}{d - (p_1 + p_2)}$ | E2 |
| $x_3 = \frac{dx_1}{d - (p_1 + p_2)}$ | $x_3 = \frac{(d - p_1)x_2}{d - (p_1 + p_2)}$ | | E3 |
| $dx_1 = (d - p_1)x_2$<br>$dx_1 = dx_2 - p_1x_2$<br>$dx_2 - dx_1 = p_1x_2$<br>$d(x_2 - x_1) = p_1x_2$ | | | E4 |
| $d = \frac{p_1x_2}{(x_2 - x_1)}$ | | $x_3 = \frac{dx_1}{d - (p_1 + p_2)}$ | E5 |
| $x_3 = \frac{\frac{p_1x_1x_2}{(x_2 - x_1)}}{\frac{p_1x_2}{(x_2 - x_1)} - (p_1 + p_2)}$<br>$x_3 = \frac{p_1x_1x_2}{\left[\frac{p_1x_2}{(x_2 - x_1)} - (p_1 + p_2)\right][x_2 - x_1]}$<br>$x_3 = \frac{p_1x_1x_2}{p_1x_2 - (p_1 + p_2)(x_2 - x_1)}$<br>$x_3 = \frac{p_1x_1x_2}{p_1x_1 - p_2x_2 + p_2x_1}$<br>$x_3 = \frac{p_1x_1x_2}{(p_1 + p_2)x_1 - p_2x_2}$<br>$x_3 = \frac{(p_1x_1x_2)\frac{1}{p_1}}{[(p_1 + p_2)x_1 - p_2x_2]\frac{1}{p_1}}$ | | | E6 |

$$x_3 = \frac{x_1 x_2}{(1+\frac{p_2}{p_1})x_1 - \frac{p_2}{p_1}x_2}$$

$$x_3 = \frac{x_1 x_2}{(1+\alpha)x_1 - \alpha x_2} \text{ avec } \alpha = \frac{p_2}{p_1} \quad \text{E7}$$

En supposant que la trajectoire du véhicule est localement droite entre 3 images successives (le processus d'acquisition étant à 10 images secondes, on suppose que la voiture sur un laps de temps de 30ms a suivi une trajectoire rectiligne, ce qui reste raisonnable), on peut écrire :

$$\alpha = \frac{p_2}{p_1} = \frac{v\delta t_2}{v\delta t_1} = \frac{\delta t_2}{\delta t_1}$$

Il est donc possible de prédire la position du panneau dans l'image, connaissant deux positions antérieures, sans connaître la focale f de la caméra ni même la vitesse du véhicule, ce qui corrobore les résultats de (Miura, et al., 2000).

De la même façon qu'ont été établies les équations pour suivre spatio-temporellement le centre du panneau, il est possible d'établir celles pour le rayon du cercle. Ainsi d'après la Figure 53:

$$\left.\begin{array}{l} \frac{x_1}{X_1} = \frac{f}{d} \rightarrow x_1 d = fX_1 \\ \frac{x_2}{X_2} = \frac{f}{d} \rightarrow x_2 d = fX_2 \end{array}\right\} \begin{array}{c} x_2 d - x_1 d = fX_2 - fX_1 \\ \rightarrow (x_2 - x_1)d = (X_2 - X_1)f \\ \frac{x_2 - x_1}{X_2 - X_1} = \frac{f}{d} \end{array}$$

Par suite, comme $x_2 - x_1 = 2r$ (le diamètre du cercle dans l'image) et $X_2 - X_1 = 2R$ (le diamètre réel du panneau), alors $\frac{r}{R} = \frac{f}{d} \leftrightarrow \frac{f}{r} = \frac{d}{R}$. On retrouve donc les équations de (E1). Ce qui nous permet de déduire que :

$$r_3 = \frac{r_1 r_2}{(1+\alpha)r_1 - \alpha r_2} \quad \text{E8}$$

Il est donc possible de prédire la taille du panneau dans l'image sans connaître sa taille réelle, ni, comme précédemment, la focale de la caméra ni la vitesse du véhicule.

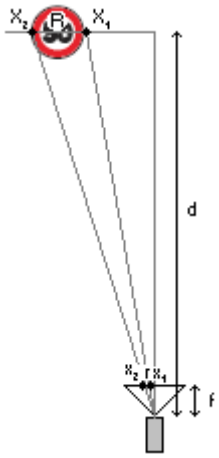



Figure 53 - Modèle sténopé

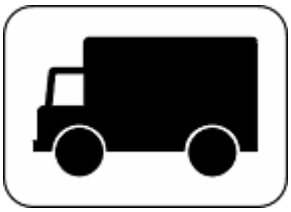
Les résultats de cette nouvelle approche (Figure 54 et Figure 55) montrent que la prédiction spatio-temporelle (à partir des deux précédentes détections) de la taille et de la position d'un panneau est correcte. Dans certains cas, cette prédiction est meilleure (mieux centrée, bonne taille) que la détection de cercle au même instant (Figure 55).


Figure 54 - Résultat de la prédiction spatio-temporelle

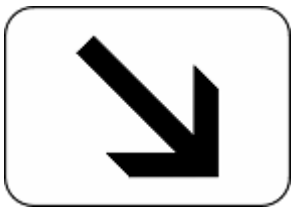
Figure 55 - Détection de cercle en haut - Résultat de la prédiction spatio-temporelle (en bas)

## 1.4. Détection et reconnaissance des panonceaux

Les panonceaux sont des petits panneaux portant des indications précises. Ils sont généralement associés aux panneaux de prescriptions absolues (ex : limitation de vitesse) et aux panneaux de danger. Leur rôle dans ces cas consiste à déterminer exactement l'étendue de la zone ou alors le type de véhicule concerné par le panneau (Figure 56), voire la plage temporelle durant laquelle le panneau sera valide.

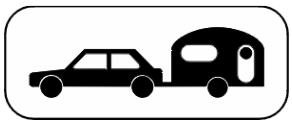


**Désigne les véhicules affectés au transport de marchandises**



**Indique la position de la voie concernée par le panneau qu'il complète**


**Rappel une limitation déjà en rigueur**



**Indique que le la limitation de vitesse ne concerne que les véhicules tractant une caravane**

Figure 56 - Certains panonceaux

Bien que la littérature n'en fasse pas mention, la détection et la reconnaissance des panneaux de limitation de vitesse n'est donc à elle seule pas suffisante pour être intégrée dans un système avancée d'aide à la conduite automobile. En effet, une limitation de vitesse peut être indiquée uniquement pour une certaine voie (la voie de sortie d'une autoroute), que pour un certain type de véhicule (les camions par exemple),… Cette information est généralement spécifiée par les panonceaux situés sous le panneau de limitation de vitesse (Figure 57).


Figure 57 - Cas d'une limitation de vitesse valide uniquement pour la voie de sortie

Pour améliorer la capacité du système SLS, il est donc nécessaire de détecter certains panonceaux sous les panneaux de limite de vitesse afin d'éliminer les panneaux ne concernant pas le véhicule.

Le problème de la détection de ces petits panneaux réside dans le fait que, même s'ils se situent toujours au-dessous des panneaux, les panonceaux n'ont pas une taille constante / proportionnelle par rapport au panneau associé et que leur distance au panneau n'est pas normalisée. Il est donc impossible de faire une reconnaissance directe du panonceau par reconnaissance d'image (via réseau de neurone ou autre) sous le panneau de limitation détectée. Il est donc nécessaire de définir une zone de recherche sous les panneaux et de réaliser une détection de rectangle dans cette zone (Figure 58).

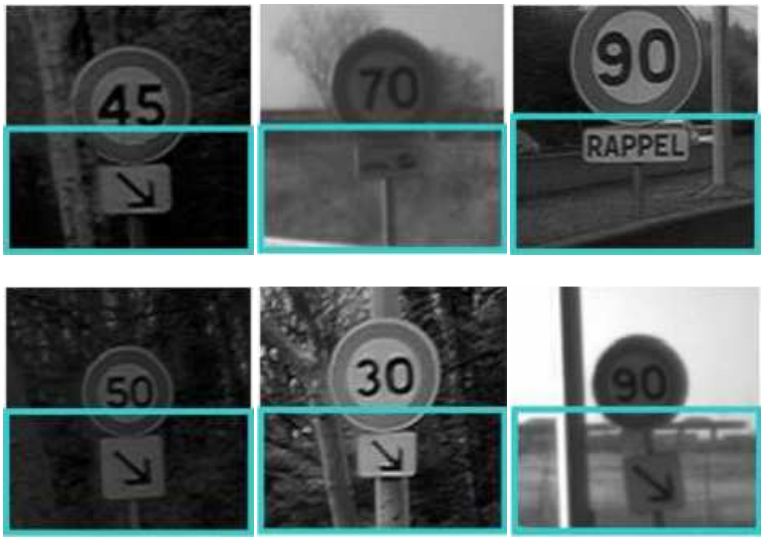



Figure 58 - Les différentes positions des panonceaux et les zones de recherche

L'algorithme de détection et de reconnaissance des panonceaux suit le même principe que celui des panneaux de limitation de vitesse (Figure 59).

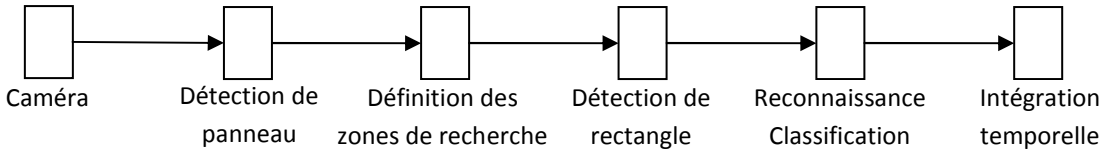



Figure 59 - Détection et reconnaissance des panonceaux

1. Un flux d'image en niveau de gris est acquis via une caméra embarquée (Figure 60).
2. De ce flux d'images sont détectés les panneaux de limitation de vitesse (Figure 61).

3. Une zone de recherche du panonceau potentiel est déterminée en fonction de la taille dans l'image du panneau de limitation de vitesse détectée (Figure 62).

4. L'image à l'intérieur de cette zone subit une égalisation d'histogramme (Figure 63).

5. Un algorithme de détection de rectangle, développé antérieurement par le laboratoire et breveté par Valeo (Herbin, et al., 2007), est appliqué sur cette zone (Figure 64). Les étapes de cet algorithme peuvent être résumées comme suit :

   – Obtenir les contours de l'image en utilisant le détecteur de Canny.
   – Eliminer les contours qui ne sont pas suffisamment horizontaux ou verticaux.
   – Assembler les segments (contours) en paires.
   – Assembler les paires en rectangles.

Figure 60 - Image originale

Figure 61 – Détection de panneau de limitation de vitesse

Figure 62 – Définition de la zone de recherche des panonceaux

Figure 63 – Egalisation d'histogramme de la zone de recherche


Figure 64 – Détection de rectangle

Là aussi, à l'instar de la détection de cercle pour les panneaux de limitation de vitesse, le fait que plusieurs rectangles soient détectés (avec des fausses détections / fausses alarmes) n'est pas un problème car elles seront invalidées ultérieurement par l'algorithme de reconnaissance.

En revanche, la difficulté de la reconnaissance des panonceaux réside dans le fait qu'ils apparaissent trop peu souvent pour constituer une base d'apprentissage correcte pour entrainer un algorithme de reconnaissance d'image. Par exemple, sur l'ensemble de nos enregistrements vidéo nous avons repéré moins d'une cinquantaine de panonceaux directionnels différents.

Pour enrichir la base d'exemples, des exemples artificiels ont été générés à partir des exemples réels en jouant sur les contrastes des images, sur les rotations et sur le bruit de sorte que les exemples générés ressemblent toujours à de vrais exemples et non à des exemples purement artificiels (Figure 65 et Figure 66).

Figure 65 - Panonceau directionnel - Image originale

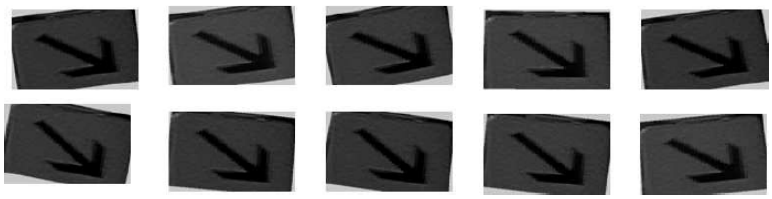


Figure 66 - Génération d'exemples artificiels

La reconnaissance est effectuée, comme pour les chiffres, via un réseau de neurones de type perceptron multi couche spécialement entrainé sur notre base d'exemples réels / semi-artificiels en utilisant notre outils d'apprentissage *Levis* (cf chapitre 5), les exemples négatifs sont quant à eux

extraits de nos vidéos et sont donc des exemples réels. Ainsi, chaque rectangle détecté sous le panneau de limitation de vitesse est analysé par le réseau de neurone pour être reconnu ou invalidé comme panonceau directionnel (Figure 67) qui seul nous intéresse dans notre étude.


Figure 67 - Reconnaissance du panonceau

La topologie du réseau (nombre de neurones de la couche cachée, et taille de la couche d'entrée, et donc taille de l'image en entrée) a été trouvée par des tests réels (Tableau 2, Tableau 3 et Tableau 4). Il apparaît qu'on pourrait utiliser une topologie de 8x8 avec 12 neurones en entrée (qui offre un bon taux de classification pour un faible temps de calcul puisque peu de neurones) mais après des tests sur des vidéos (donc avec des cas réels d'utilisation) on obtient seulement 85% de bonne détection avec cette topologie alors qu'avec une image de 12x12 pixels en entrée et 12 neurones sur la couche cachée on obtient 93% de bonne détection. C'est donc cette dernière topologie (144 – 12 – 1) qui a été retenue après les tests réels sur vidéo.

| Taille de la couche cachée | Taux de bonne classification sur la base d'apprentissage | Taux de bonne classification sur la base de test |
|---|---|---|
| 0 | 99% | 93% |
| 6 | 99% | 95% |
| 10 | 99% | 96% |
| 12 | 100% | 98% |

Tableau 2 - Taux de bonne classification du réseau de neurone sur la base de panonceau de taille 8x8

| Taille de la couche cachée | Taux de bonne classification sur la base d'apprentissage | Taux de bonne classification sur la base de test |
|---|---|---|
| 0 | 98% | 95% |
| 5 | 99% | 98% |
| 10 | 99% | 98% |
| 15 | 99% | 98% |

Tableau 3 - Taux de bonne classification du réseau de neurone sur la base de panonceau de taille 10x10

| Taille de la couche cachée | Taux de bonne classification sur la base d'apprentissage | Taux de bonne classification sur la base de test |
|---|---|---|
| 0 | 99% | 96% |
| 8 | 99% | 98% |
| 12 | 99% | 98% |
| 24 | 100% | 98% |

Tableau 4 - Taux de bonne classification du réseau de neurone sur la base de panonceau de taille 12x12

## 1.5. Détection et reconnaissance des fins de limitation

Enfin, le dernier type de panneau à reconnaître afin de concevoir un système de reconnaissance visuel complet est la catégorie de panneau de fin de limitation de vitesse. Il existe deux types de ces panneaux (Figure 68):

1. Les fins de limitation spécifique à une seule vitesse (i.e. correspondant à un panneau de début de limitation en amont) en aval d'une zone de limitation temporaire (par exemple une zone de travaux).
2. Les fins de toute limite, indiquant une zone théoriquement sans limitation de vitesse, mais qui sont généralement utilisés dans des zones de limitation temporaire comme pour le premier type.

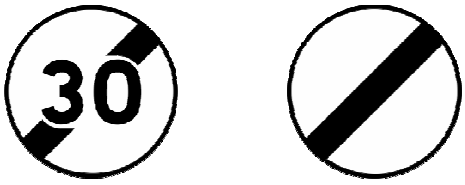



Figure 68 - Les 2 types de panneau de fin de limitation français

Le processus de reconnaissance des panneaux, tel que présenté jusqu'ici, essaye de segmenter les caractères afin de reconnaître un panneau de début de limitation de vitesse. Ce schéma pose problème pour le 1$^{er}$ type de panneau de fin de limitation qui est reconnu comme panneau de début de limite (les chiffres à l'intérieur du panneau sont segmentés et reconnus). Il est donc nécessaire d'incorporer la reconnaissance de ces panneaux en amont de celle des débuts de vitesse (Figure 69).

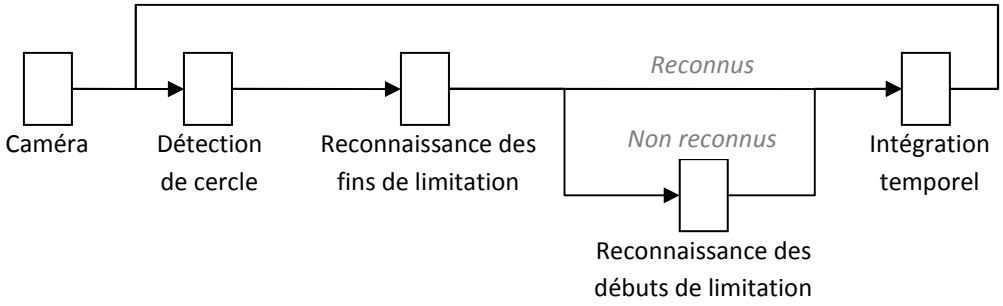



Figure 69 – Schéma d'intégration de la reconnaissance des panneaux de fin de limitation de vitesse

Comme pour les panonceaux, et bien que nécessaire à un système visuel complet, la littérature comporte peu de travaux concernant la détection et la reconnaissance des panneaux de fin de limitation de vitesse. Ces rares travaux, et comme le voudrait l'intuition, proposent 2 approches différentes afin de reconnaître ce type de panneau. La première approche consiste à reconnaître directement la globalité du panneau de fin de limitation de vitesse, i.e. d'utiliser l'image entière du panneau, issu du processus de segmentation (par contour de forme ou par couleur), dans un algorithme de classification / reconnaissance d'image (réseau de neurone, support vector machine,…) comme le font (Garcia, et al., 2006), (Broggi, et al., 2007) et (Maldonado-Bascón, et al., 2007). Il est à noter que ces approches incorporent directement la reconnaissance des panneaux de début de vitesse dans leur algorithme de reconnaissance et n'effectuent pas de traitement spécifique pour chaque type de panneau. Ainsi leur algorithme de classification se voit affublé de plusieurs sorties et doit réussir à discriminer un grand nombre de classes (i.e. types de panneau) ce qui peut poser problème quant à la pertinence de la discrimination sur plusieurs classes. La seconde approche, qui a été mise en œuvre seulement par (Caraffi, et al., 2008), consiste à détecter la bande noire présente sur ces panneaux. Cependant, cette technique mise en œuvre pour les panneaux italiens qui, comme les panneaux belges, ont une bande noire très marquée (quelque soit le type de fin de limitation), ne peut s'appliquer à tous les panneaux européens (Figure 70) et notamment aux panneaux français qui n'ont que de petits segments noirs peu présent dans l'image, pour le cas des panneaux de fin de limitation spécifique à une vitesse (Figure 68), segments qui seraient très difficilement détectable en cas réel (Figure 71).


Figure 70 - Exemples de panneaux européens de fin de limitation (français, allemand, norvégien, espagnol, belge)

La reconnaissance doit donc s'effectuer ici via une classification globale de l'image du panneau issue de la détection de cercle (Figure 69) déjà utilisée pour la détection des débuts de limitation de vitesse. Cependant le problème principal réside dans le fait que trop peu de panneaux de fin de limitation sont présents dans nos enregistrements (car très peu présents dans la réalité).

Figure 71 - Cas réel d'une fin de limitation

De ce fait, la base d'exemples d'apprentissage doit être générée à partir de prototypes artificiels (Figure 68) auxquels sont ajoutés, comme pour le cas des panonceaux :

1. Du bruit (gaussien et ou uniforme)
2. Des déformations géométriques
3. Des changements de contrastes et de luminosité
4. Des changements d'échelle

Cependant, le problème de cette technique avec les panneaux de fin de limitation est qu'ils sont circulaires. Il y a donc intrinsèquement des zones « de fond » dans l'image du prototype ne correspondant pas au panneau mais correspondant, dans la réalité, à l'environnement se situant derrière le panneau (des arbres, des immeubles,…). Afin de rendre l'apprentissage efficace sur la base d'exemple, il est donc nécessaire, voir indispensable, de ne pas laisser ces zones vides (i.e. blanches). Les exemples générés sont donc dans une dernière étape placés sur des images de fond / d'environnement extraites directement de nos séquences vidéos. Au final, notre base d'apprentissage est constituée d'exemples artificiels difficilement différentiables d'exemples réellement extraits de vidéos (Figure 72 et Figure 73).


Figure 72 - Exemples de panneaux réels avec différents fonds

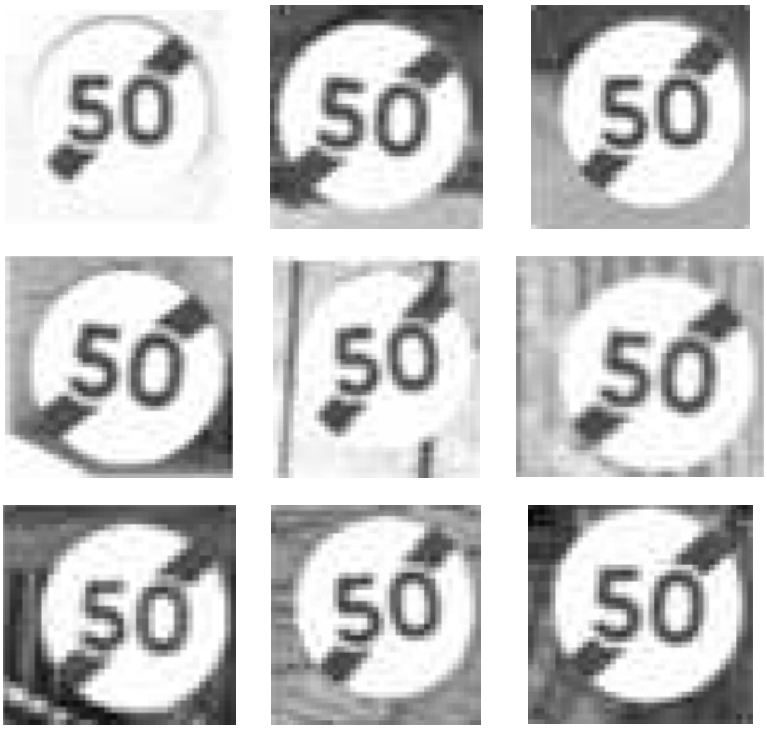


Figure 73 - Quelques exemples synthétiques de panneaux "Fin de limite 50"

L'apprentissage sur cette base d'exemples est effectué par un réseau de neurones de type perceptron multi-couches. Les exemples négatifs de la base d'apprentissage sont de vrais exemples extraits de nos vidéos. La topologie du réseau est là aussi trouvée empiriquement (Tableau 5, Tableau 6 et Tableau 7). Il en résulte qu'une topologie de 256 entrées (16x16 pixels en taille de l'image), avec 10 neurones en couche cachée est le meilleur compris temps de calcul / performance.

| Taille de la couche cachée | Taux de bonne classification sur la base d'apprentissage | Taux de bonne classification sur la base de test |
|---|---|---|
| 0 | 93% | 90% |
| 5 | 87% | 86% |
| 10 | 97% | 95% |
| 15 | 98% | 96% |

Tableau 5 - Taux de bonne classification du réseau de neurone sur la base de fin de limitation de taille 12x12

| Taille de la couche cachée | Taux de bonne classification sur la base d'apprentissage | Taux de bonne classification sur la base de test |
|---|---|---|
| 0 | 80% | 75% |
| 5 | 60% | 58% |
| 10 | 99% | 97% |
| 15 | 99% | 98% |

Tableau 6 - Taux de bonne classification du réseau de neurone sur la base de fin de limitation de taille 16x16

| Taille de la couche cachée | Taux de bonne classification sur la base d'apprentissage | Taux de bonne classification sur la base de test |
|---|---|---|
| 0 | 79% | 73% |
| 5 | 62% | 62% |
| 10 | 99% | 99% |
| 15 | 99% | 99% |

Tableau 7 – Taux de bonne classification du réseau de neurone sur la base de fin de limitation de taille 20x20

## 1.6. Expérimentation et résultats

### 1.6.1. Véhicules prototypes

Ce travail de détection et reconnaissance visuelles des panneaux de limitation de vitesse a été réalisé en collaboration entre le Centre de Robotique de l'Ecole des Mines et l'équipe Driving Assistance Domain de Valeo. Le système a donc pu être testé sur différents prototypes (voitures) équipés de différents capteurs (caméras). Au sein du Centre de Robotique, nous utilisons une Citroën C3 équipée d'une caméra Marlin couleur 8 bits (Figure 74). L'équipe de Valeo utilise une Volvo S80 comme véhicule de démonstration utilisant une caméra Small noir et blanc 12 bits. Les 2 caméras fournissent des images de résolution 640x480 pixels traités à 20 images/s avec un processeur Dual Core à 2Ghz.

Des essais concluants ont aussi été réalisés avec d'autres caméras dont une Cypress couleur 12bits fournissant des images couleurs de 752x480 pixels et une caméra Sony analogique fournissant des images (après conversion numérique) de 720x576 pixels 8 bits. Là aussi le système a été capable de traiter ces flux vidéo en temps réel.


Figure 74 - La flotte de véhicules prototype du Centre de Robotique

Vues de l'extérieur, ces voitures « intelligentes » ne montrent pas de signe distinctif particulier, hormis les couleurs du laboratoire (Figure 74). Néanmoins, à l'intérieur le coffre est rempli de matériel électrique (disjoncteur, transformateur, onduleur,…) et d'instruments de calcul (PC embarqué, GPS, centrale inertielle,…) qui y sont connectés (Figure 75).

On retrouve à l'intérieur du véhicule, à l'avant, la caméra embarquée filmant la route près du rétroviseur central ainsi que l'écran et le clavier de contrôle de l'ordinateur (Figure 76).

Figure 75 - Le coffre d'une C3 prototype et ses instruments

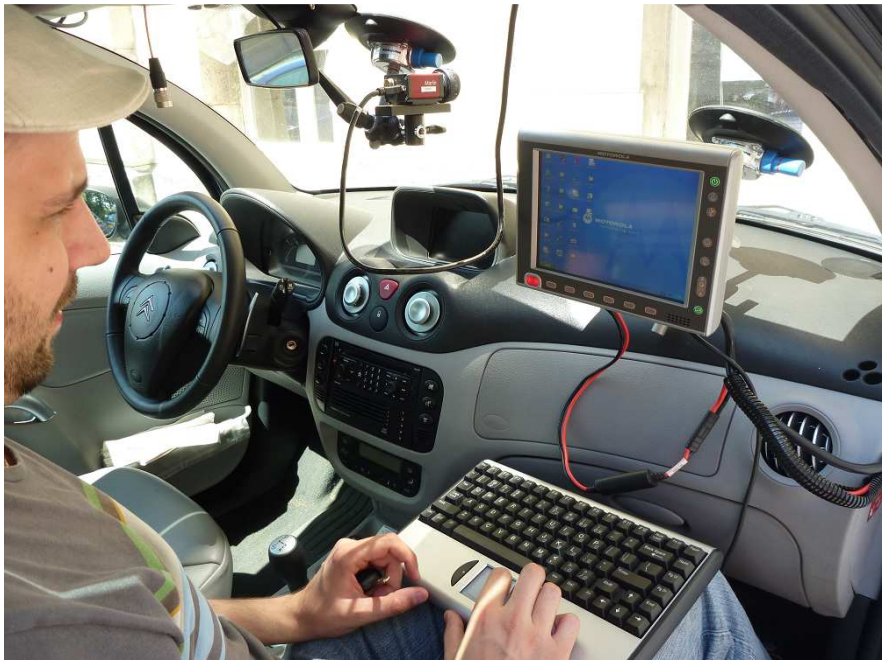


Figure 76 - A l'avant d'une C3 prototype

Tous nos véhicules de démonstration (du Centre de robotique et de Valeo) sont équipés de la plateforme de prototypage [rt]Maps initialement développée au Centre de Robotique (Steux, 2001) maintenant commercialisée par la société Intempora. Il s'agit d'une plateforme qui permet d'effectuer des acquisitions avec les différents capteurs incorporés sur le véhicule instrumenté et de gérer ce flux d'informations complexes. Ce logiciel permet d'acquérir en temps réel les signaux à haut débit de type vidéo, radar, laser, GPS, centrale inertielle, de les synchroniser et de les gérer en temps réel dans une solution embarquée (Figure 77).

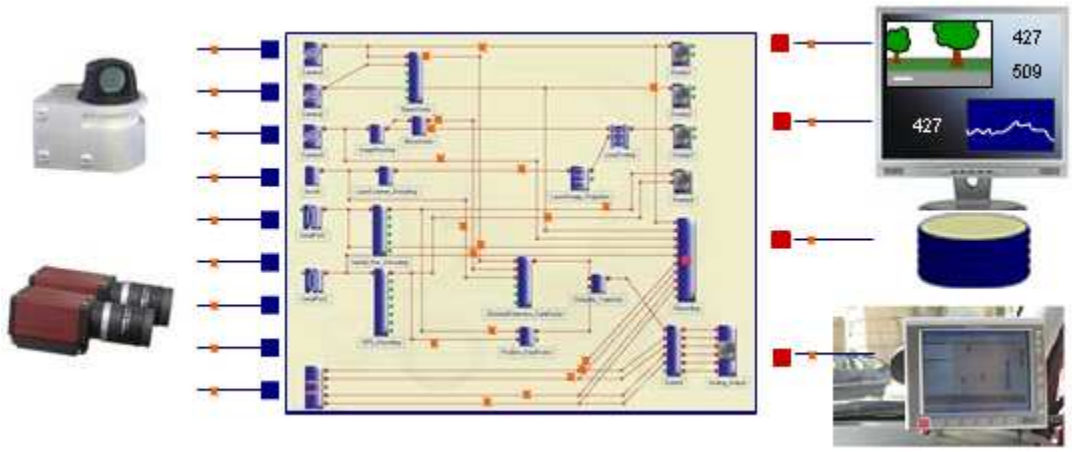



Figure 77 - Acquisition de données par le logiciel [RT]Maps

On obtient donc une base de données enregistrée et datée. Un rejeu synchronisé de cette base permet de simuler l'expérimentation (prototypage) avec une grande fidélité. L'utilisateur peut intégrer ses propres algorithmes de traitement sous la forme de modules présentant des entrées et des sorties de données standards. Ce qui permet de concevoir et de tester en toute liberté des algorithmes qui agissent sur les informations provenant en temps réel d'un grand nombre de capteurs (Figure 78).

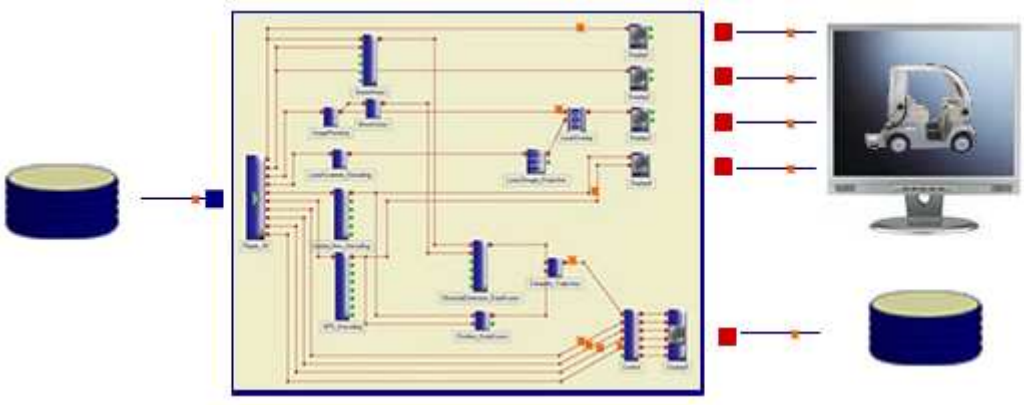


Figure 78 - Rejeu synchronisé de données acquises par le logiciel [RT]Maps

La gestion de ces modules et données est en outre rendue plus confortable par une interface graphique permettant de visualiser les connexions entre les différents modules de traitement, voire de les modifier dynamiquement (Figure 77 et Figure 78).

Une fois les algorithmes et les applications validés en laboratoire, leur installation sur l'ordinateur de notre véhicule de démonstration, équipé également de RTMaps, n'est qu'une formalité.

### 1.6.2. Résultats généraux

Un système de détection et de reconnaissance de panneaux de limitation de vitesse peut être évalué par différents critères. Par exemple, on peut comparer les détections des panneaux image par image. Mais cette mesure de performance « statique » n'est pas la plus importante puisqu'un panneau est visible sur plusieurs images successives. Il n'est donc pas nécessaire de réussir à le détecter sur toutes les images pour le reconnaître. En revanche, l'important réside dans le fait que le panneau soit détecté et validé durant la portion de vidéo où il apparaît.

L'évaluation du système est donc réalisée de façon globale en comparant les types, le temps, et la position des panneaux détectés et validés par rapport à une vérité terrain. Une vérité terrain est un fichier comportant « la détection et la reconnaissance » réalisées par un opérateur humain des panneaux (i.e. type et position sur chaque image) d'une vidéo. Le développement d'un outil spécifique (SamGT – cf chapitre 5) proposant à la fois l'édition de ces fichiers et le rejeu des données (des détections manuelles) au sein de RTMaps, permet d'évaluer facilement et efficacement les algorithmes mis en œuvre.

Les résultats calculés et présentés dans cette section correspondent au pourcentage de panneaux de limitation de vitesse détectés et validés pendant la durée durant laquelle ils sont visibles dans une vidéo. Evidemment, ce calcul n'est effectué que sur des séquences vidéo qui n'ont pas été utilisés pour l'apprentissage de nos algorithmes de reconnaissance (de chiffre, de panonceau et de fin de limitation).

Le système aurait pu être évalué facilement sur simulateur (SiViC, Oktal,…) ; néanmoins, ce type d'outils fournit des images bien trop propres et trop éloignées de la réalité (les panneaux sont bien contrastés dans l'image, ils ne sont pas détériorés,…), pour réellement témoigner de l'efficacité d'un système ADAS (quelque soit le système) comme en témoigne la Figure 79. Nous avons donc tiré parti de la puissance du logiciel RTMaps afin de réaliser des enregistrements des données issus de nos capteurs embarqués (caméra, gps,…) dans nos divers véhicules de démonstration pour nous constituer des bases de données de tests pour le système.

C'est ainsi que le système de détection de panneaux de limitation de vitesse (de type européen) a été évalué en utilisant un jeu d'enregistrement vidéo « réelle » de routes françaises et allemandes sous diverses conditions lumineuses (au matin, au soir, en pleine journée, nuageux ou non,…) provenant des 2 véhicules de démonstration (et donc issu de caméra différentes). Ce jeu de vidéos contient à peu près 280 panneaux de limitation de vitesse distincts couvrant 11 limitations différentes (30, 40, 50, 60, 70, 80, 90, 100, 110, 120 et 130 km/h) pour un peu plus de 150 minutes d'enregistrement. Toutes ces vidéos ont été manuellement annotées (via SamGT) pour réaliser les fichiers de vérités terrain.

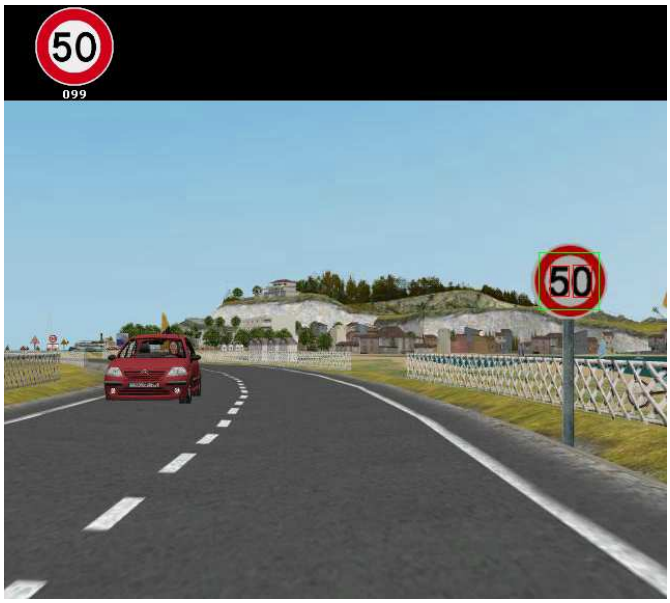



Figure 79 – Exemple de résultat de détection obtenu sous le simulateur Oktal

La performance générale du système est de 94% de bonnes détections sur ce jeu de vidéo et le taux de mauvaise classification (i.e. un panneau détecté avec une mauvaise limitation de vitesse) est inférieur à 1%. La totalité des 6% de mauvaises détections sont des *non détections* causées soit par une occultation partielle (par un autre véhicule par exemple), soit parce que la segmentation individuelle des chiffres ne parvient pas à se faire correctement, notamment quand le panneau est trop petit et/ou loin, ou quand les chiffres sont trop « collés » (cas de certains panneaux à 3 chiffres). Il est par ailleurs remarquable qu'aucune fausse alarme (détection d'un panneau qui n'existe pas) n'a été validée durant ces 150 minutes de vidéo testées, paramètre qui est réellement important d'atteindre pour qu'un système ADAS puisse être embarqué au sein d'un véhicule (que le système ne perturbe pas le conducteur avec des fausses informations).

Il serait intéressant ici de pouvoir comparer ces résultats avec ceux des autres travaux présentés dans la littérature. Cependant, cette comparaison ne peut avoir de sens que si les différents systèmes sont évalués sur les mêmes bases de test. Malheureusement, il n'existe aucune base de test partagée par les différents centres de recherche travaillant sur la problématique de détection et reconnaissance des panneaux de limitation de vitesse. En fait, cette question relève un problème récurrent de la comparaison des systèmes de détection visuel : il existe rarement des bases de test ouvertes.


Figure 80 - Exemple de détection de panneau français. Tous les cercles détectés sont affichés en rouge, ainsi que les chiffres candidats segmentés. Tous les panneaux reconnus sont affichés en haut (dans le bandeau noir) et le panneau validé est affiché au centre de l'image (caméra SMALL 12bits).


Figure 81 - Exemple de détection de panneau français (Caméra CYPRESS 12bits)

Enfin, le système proposé s'avère pouvoir fonctionner même sur des enregistrements vidéos effectués à l'aide d'un simple appareil photo numérique tenu à la main à l'intérieur d'un véhicule, comme en témoigne la Figure 82 obtenue sur une vidéo enregistrée à bord d'un autocar au Sénégal (où les panneaux sont identiques aux modèles français).


**Figure 82 – Panneau de limitation de vitesse sur une séquence enregistrée (au Sénégal) avec un simple appareil photo numérique tenu à la main à bord d'un autocar (à noter la grande instabilité face au « tangage » de la vidéo, ces 2 captures étant successives)**

### 1.6.3. Apport de la segmentation globale des chiffres

Ici est évalué à la fois l'apport de l'algorithme de segmentation globale de caractère mais aussi, car intrinsèquement lié, de l'apport de l'amélioration de l'algorithme de Hough.

Ce nouvel algorithme apporte des améliorations dans de nombreux cas. Par exemple, dans le cas le plus courant, quand le panneau est trop loin pour que l'algorithme initial puisse correctement segmenter tous les chiffres (ie quand les chiffres s'entremêlent dans l'image binaire), le nouvel algorithme arrive à correctement segmenter tous les chiffres (Figure 83). Les panneaux temporaires posés à même le sol posaient eux aussi problème, car mal orientés (non perpendiculaire au champ de la caméra), le nouvel algorithme arrive là aussi à segmenter les chiffres et le système de reconnaissance valide le panneau comme en témoigne la Figure 85. Enfin, dans de rares cas, un panneau se retrouve tagué et est très difficilement reconnaissable. Là aussi, le nouvel algorithme de segmentation peut améliorer la reconnaissance, si le tag n'est pas trop destructeur (Figure 84).


**Figure 83 – Exemple de l'apport de l'algorithme de segmentation globale (en bas) face à l'algorithme initial (en haut) – Cas d'un panneau trop petit. A noter le panneau occulté sur la gauche.**


**Figure 84 – Exemple de l'apport de l'algorithme de segmentation globale (en bas) face à l'algorithme initial (en haut) – Cas d'un panneau taggué.**

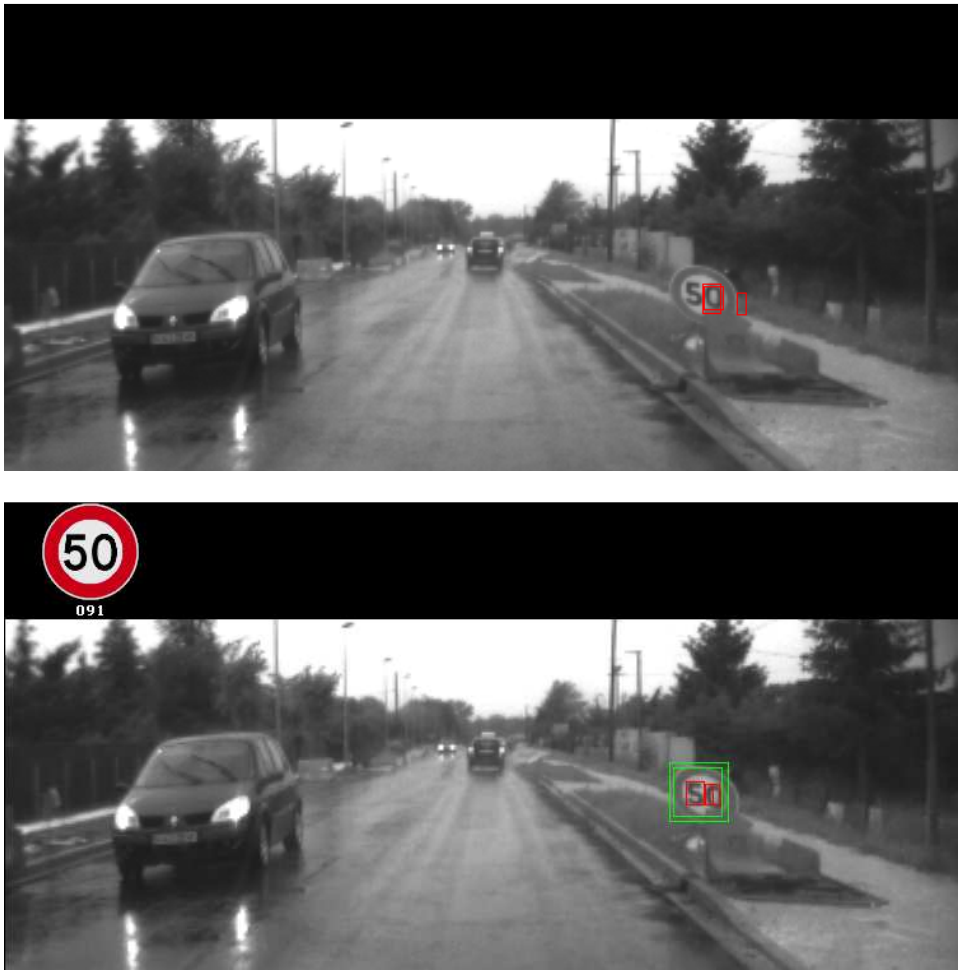



**Figure 85 - Exemple de l'apport de l'algorithme de segmentation globale (en bas) face à l'algorithme initial (en haut) – Cas d'un panneau temporaire mal incliné**

Ces résultats visuels, sont confirmés par une évaluation automatique sur notre jeu de test. Ainsi, comme il peut être vu dans le Tableau 8, ce nouvel algorithme améliore considérablement le taux de bonne détection qui passe de 85 à 94%. Le taux de mauvaise classification (i.e. les panneaux détectés mais ayant été reconnu avec une mauvaise limitation de vitesse) reste en dessous de 1%. Les 6% des panneaux non détectés, comme dit précédemment, sont des signes non vus par le système (la plupart du temps en raison du fait que les bords du panneau ne sont pas assez contrastés dans l'image, mettant en échec la détection de cercle sur celui-ci).

| Méthodes de reconnaissance | Panneaux détectés, reconnus et validés avec la bonne limitation | Panneaux mal classifiés |
|---|---|---|
| Méthode initiale de segmentation | 85% | 0.7% |
| Nouvelle méthode de segmentation globale de chiffre | 94% | 0.7% |

**Tableau 8 - Evaluation globale du système de reconnaissance visuel des panneaux de limitation de vitesse européen sur des routes françaises et allemandes**

### 1.6.4. Évaluation de la robustesse du système

Le système ici développé se voulait plus robuste par rapport aux systèmes proposés dans la littérature, face notamment à tous les facteurs dégradant les couleurs des panneaux que ce soit réellement (la couleur qui s'estompe avec le temps quand un panneau est trop vieux) ou « artificiellement » à cause des changements de conditions lumineuses (les couleurs vues par la caméra embarquée vont être différentes de jour et de nuit, s'il fait ensoleillé ou nuageux, quand il y a du brouillard ou non,…) car basé sur une détection via contours de forme sur des images en niveaux de gris plutôt que sur une segmentation couleur.

Nous venons de voir que le système fonctionne correctement dans les conditions réelles de nos vidéos, qui couvrent un large panel de ces cas. Cependant, il serait intéressant de pouvoir évaluer le système face à une dégradation volontaire et plus ou moins poussée de ces paramètres afin d'étudier l'évolution des performances du système et évaluer sa robustesse.

C'est ainsi que le système a été évalué, sur le même jeu de données que précédemment, mais en changeant artificiellement le contraste et la luminosité des vidéos (pour simuler les changements de conditions lumineuses).

Ce travail qui pourrait être fastidieux si réalisé manuellement (i.e lancer les tests l'un à la suite des autres pour chaque couple contraste / luminosité sur nos vidéos, puis noter les résultats) a été rendu possible par l'automatisation de cette tâche via le logiciel SamGT (cf chapitre 5) intégré au sein de notre plateforme de prototypage $^{RT}$Maps (et donc sans changer d'une quelconque façon l'algorithme de détection).

Il résulte de cette série de tests, illustrée par la Figure 89, que les performances globales du système restent autour des 90% du taux de bonne détection sur une large plage de valeurs de contraste / luminosité et ne se détériorent réellement que sur des contrastes très élevés artificiellement avec une luminosité très faible (ce qui ne pourrait avoir lieu dans la réalité) comme en témoigne la Figure 87. A noter que dans ce dernier cas, l'image est trop dégradée et le panneau ne peut être reconnu même visuellement par un humain (Figure 86).

Figure 86 - Image à fort contraste et faible luminosité artificiels

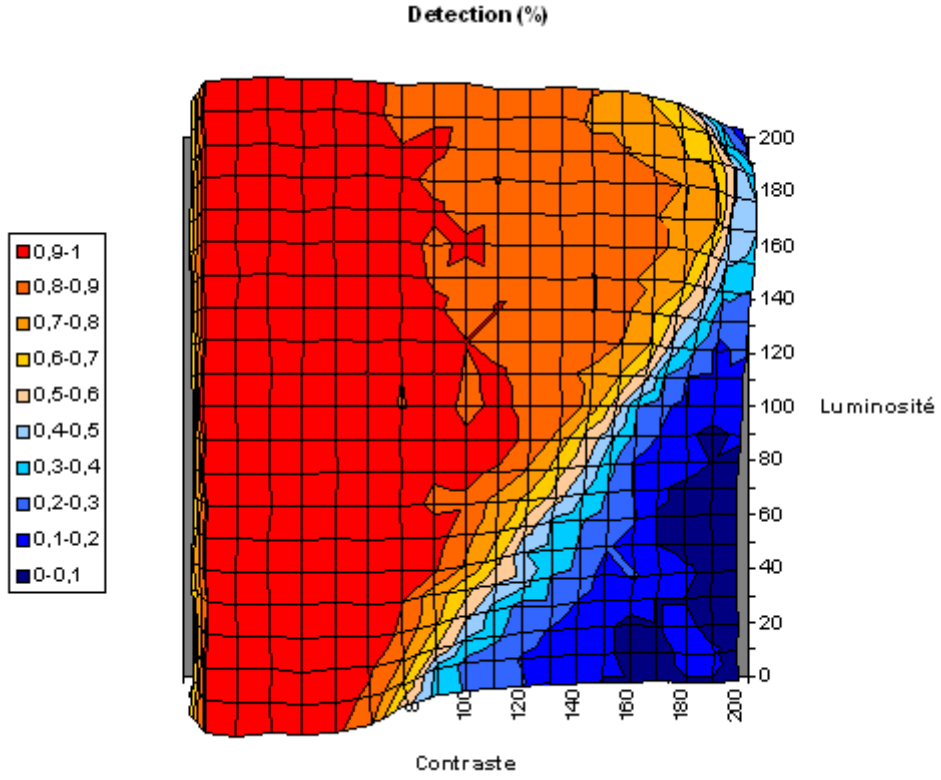



Figure 87 - Evaluation de la robustesse face aux variations de luminosité / contraste

En complément de ces tests, une évaluation en cas réel sur route du système sous de mauvaises conditions météorologique a aussi été effectuée par la société Valeo. Cette évaluation (Tableau 9) sur un trajet d'une longueur de 50km en France s'est déroulée sous un temps nuageux et pluvieux. Parmi les 13 panneaux de limitation de vitesse non vus, 3 sont partiellement occultés par d'autres véhicules ou détériorés et sont par conséquent impossible à reconnaître, 4 panneaux sont filmés avec un taux de contraste très faible à l'intérieur de tunnels, et 3 sont correctement reconnus mais non validés dû fait que leur niveau de confiance ne dépasse pas le seuil fixé.

| Limitation de vitesse | Panneaux correctement détectés, reconnus et validés | Panneaux correctement reconnus mais non validés | Panneaux non vus |
|---|---|---|---|
| 30 | 1 | 0 | 1 |
| 50 | 10 | 0 | 3 |
| 70 | 21 | 0 | 3 |
| 80 | 6 | 0 | 0 |
| 90 | 28 | 3 | 3 |
| 110 | 4 | 0 | 0 |
| Total | 70<br>**84%** | 3<br>**4%** | 10<br>**12%** |

Tableau 9 - Evaluation sur route en France, sous de mauvaises conditions météorologiques

## 1.6.5. Résultats préliminaires des systèmes annexes

Les systèmes de détection et de reconnaissance des panonceaux et des panneaux de fin de limitation ne sont pas ou que très peu traités dans la littérature. Il est ainsi difficile d'avoir du recul et une idée sur la méthodologie à mettre en place pour construire de tels systèmes. Ici sont montrés des résultats dits « préliminaires » car issus de premières approches (présentées plus haut).

### *a) Détection et reconnaissance des panonceaux*

Une première évaluation a été réalisée sur un jeu de vidéos françaises contenant 50 panneaux de limitation de vitesse dont 18 avec un panonceau de sortie de voie.

Les résultats obtenus sont illustrés Figure 88 et Figure 89, et quantifié dans le Tableau 10. Le taux de bonne reconnaissance de 78% est déjà très satisfaisant pour une première implémentation et laisse la possibilité d'améliorer la méthodologie de détection mise en place en incorporant par exemple plus d'exemples (synthétiques et réels) dans la base d'apprentissage et en collectant plus d'exemples négatifs.


Figure 88 - Illustration d'une détection de panonceau français de sortie de voie


Figure 89 - Exemple d'évaluation panneau et panonceau avec changement automatisé de contraste / luminosité

| Panneaux de limitation de vitesse avec panonceau de sortie correctement détectés et reconnus | Fausses alarmes sur d'autres panneaux |
|---|---|
| 14/18 = **78%** | 3 / 32 = 9% |

Tableau 10 - Première évaluation de reconnaissance des panonceaux de sortie de voix français

### b) *Détection et reconnaissance des panneaux de fin de limitation de vitesse*

Une première évaluation systématique a été réalisée sur un petit jeu d'enregistrements vidéo français et allemands (Figure 90) contenant 22 panneaux de fin de limitation. Les résultats obtenus (Tableau 11) ne sont pas parfaits, mais déjà satisfaisants pour une première utilisation au sein du système global. Là aussi les résultats laissent la possibilité d'améliorer la technique mise en œuvre en collectant par exemple plus d'exemples positifs et négatifs.

| Panneaux de fin de limitation de vitesse correctement détectés et reconnus | Fausses alarmes validées |
|---|---|
| 18/22 = **82%** | 1 |

Tableau 11 - Première évaluation de reconnaissance des panonceaux de fin de limitation de vitesse européens (francais + allemands)

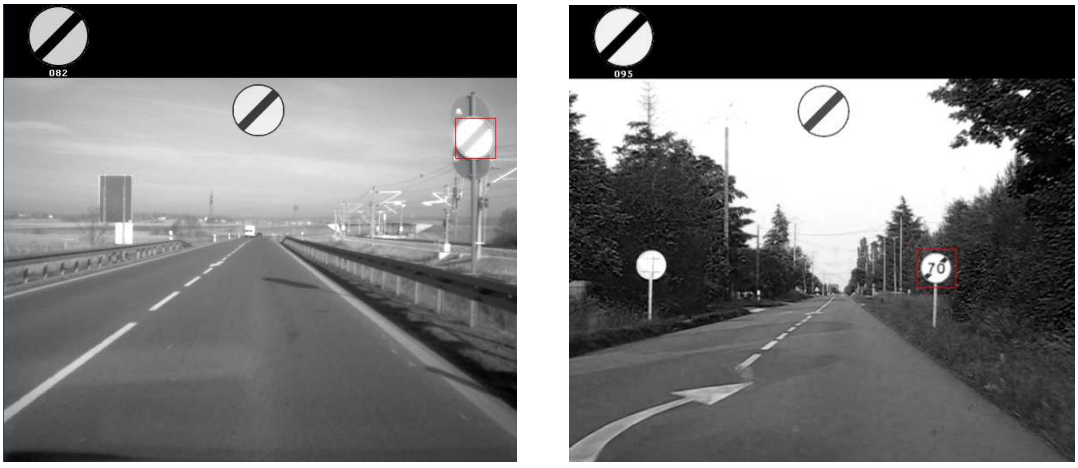


Figure 90 – Illustration d'une bonne reconnaissance de panneaux de fin de limitation en Allemagne (à gauche) et en France (à droite)

### 1.6.6. Résultats de nuit

Parce que le système repose sur un traitement d'images en niveaux de gris, et donc insensible aux changements de conditions lumineuses (comme déjà présenté plus haut), il montre des résultats très prometteur de nuit (Figure 12 et Figure 91). Ceci a été récemment quantifié par Valeo par une première campagne de test de nuit : pendant une heure de trajet, 78% des 60 panneaux de limitation de vitesse rencontrés ont été correctement détectés et reconnus.


Figure 91 - Détection et reconnaissance d'un panneau de limitation de vitesse de nuit

### 1.6.7. Panneaux LED

En basant le système de détection des panneaux de limitation sur un processus de segmentation des chiffres au sein des panneaux, il est très simple de l'adapter pour reconnaître les panneaux LED généralement utilisés pour des limitations variables. Ainsi l'algorithme appliqué est exactement le même, excepté que les images sont de niveau de gris inversés (Figure 92).

Figure 92 - Exemple de segmentation sur des images inversées de panneau LED (français à gauche, allemand à droite)

Une première évaluation sur 10 panneaux LED français et allemand montre 8 / 10 = 80% de bonne reconnaissance (sans même réapprentissage spécifique du réseau de neurone de reconnaissance des chiffres avec des chiffres extraits de ces panneaux). Un exemple de bonne détection est montré Figure 93.


Figure 93 - Illustration de la détection de panneaux LED

### 1.6.8. Panneaux américains

Du fait d'un développement complètement modulaire du système il est très facile de remplacer la détection de cercle par la détection de rectangle (utilisé pour les panonceaux) afin de détecter les panneaux rectangulaires américains. De plus, le fait que le système effectue la reconnaissance des

panneaux en segmentant les chiffres contenus à l'intérieur (ce qui constitue l'une de ses originalités face aux systèmes proposés dans la littérature qui effectuent tous une reconnaissance globale du panneau) permet ici, sans changer l'architecture général de l'algorithme de reconnaître les panneaux américains détectés.

Le système, dans sa version américaine, reconnaît pour l'instant les panneaux les plus courants (i.e. ceux où est marqué « SPEED LIMIT » juste au dessus de la limitation de vitesse). Les autres types de panneaux (Figure 5) ne sont pas encore reconnus, mais il serait très aisé d'adapter, grâce à son architecture modulaire, le système pour les reconnaître.

Une première évaluation de ce système américain montre un taux de reconnaissance des panneaux de  90% (Figure 94).

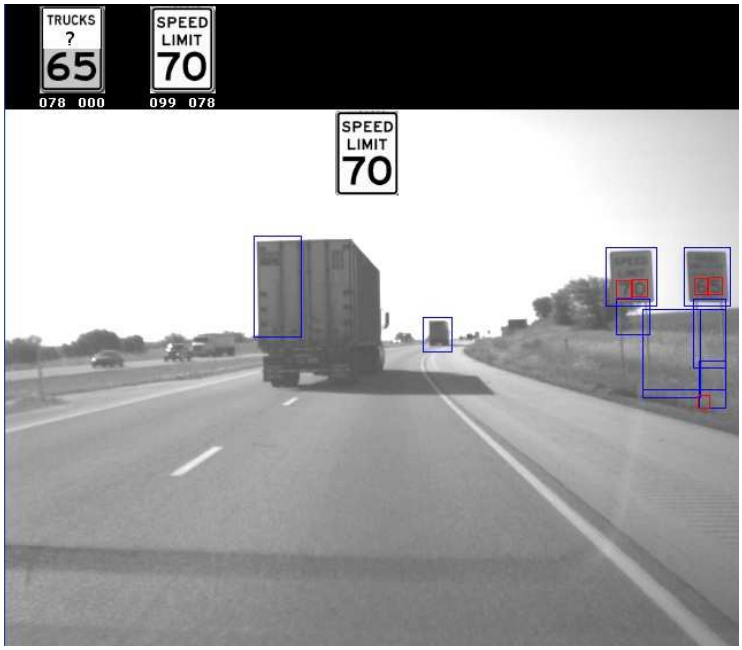



Figure 94 - Exemple de détection de panneaux américains

## 1.7. Conclusion et perspectives

Il a été présenté ici un système de détection et de reconnaissance visuelle des panneaux de limitation de vitesse visant à être intégré dans un système d'aide à la conduite automobile. L'enjeu majeur était de développer un système capable de tourner en temps réel dans un véhicule et insensible aux problèmes de changement de luminosité.

Pour cela, un système original a été développé de façon modulaire, basé à la fois sur de la détection de contours de forme (cercle ou rectangle) sur des images en niveau de gris et sur la reconnaissance de chiffres. Il a été démontré sa robustesse face aux variations de luminosité et de contraste pouvant survenir aux différents moments de la journée et de l'année. De surcroît, le système a été éprouvé

sur des heures de tests réels montrant un taux de bonne détection en Europe (en France et en Allemagne) de l'ordre de 94% avec un remarquable faible taux de fausses alarmes (une seule fausse alarme). Des évaluations dans d'autres pays d'Europe, comme en Italie, où les chiffres des panneaux sont sensiblement différents, ont aussi été effectuées de façon qualitative donnant des résultats prometteurs (Figure 95). La modularité du système a permis de l'adapter et de le tester facilement aux Etats-Unis prouvant son bon fonctionnement sur des panneaux rectangulaires et de l'adapter pour la reconnaissance des panneaux LED.

Enfin, un tel système de détection et de reconnaissance de panneaux de limitation de vitesse, dont le but premier est d'informer le conducteur de la limitation en cours, ne doit pas se contenter de reconnaître les seuls panneaux de début de limitation, mais doit aussi, afin d'être complet et utilisable, détecter les autres types de panneaux jouant un rôle important : les panneaux de fin de limitation et les panonceaux apportant une information cruciale pour la prise de décision et être sûr que la limitation vue par le système se réfère bien à notre véhicule. C'est pourquoi, le système présenté ici propose un tel framework à l'état de prototype.


**Figure 95 - Reconnaissance d'un panneau de limitation de vitesse italien, illustrant des résultats prometteur pour un système européen global**

L'amélioration du système doit passer par l'ajout de certains types de panneau pas encore ou peu pris en compte. Par exemple, les panneaux de limitation de vitesse « zone 30 » situés en ville n'ont été que peu étudiés durant les tests, alors que ces panneaux sont amenés à se développer, pour des raisons de sécurité, afin de limiter la vitesse dans les zones urbaines à 30km/h. En effet, un piéton a 90% de chance de survie quand il est renversé par une voiture roulant à 30km/h, alors qu'il n'en a que 20% pour une voiture roulant à 50km/h. Les débuts de limite de « zone 30 » ne posent pas,

véritablement, de problème (à part le fait qu'ils sont plus difficilement détectables du fait de leur petitesse). Ces panneaux étant toujours représentés de la même façon que leurs homologues. Les fins de limitation de ces zones sont quant à elles plus problématiques en France, la barre noire étant à l'extérieur du cercle (Figure 96), alors que dans les autres pays européens, à l'instar de leurs congénères « fin de limitation », cette barre traverse le cercle et le nombre sur les panneaux « fin de zone ».


Figure 96 - Exemple de panneau de fin de Zone 30 en France

Il est aussi évident qu'il faut améliorer le framework proposé pour la détection et la reconnaissance des panneaux de fin de limitation et des panonceaux. Pour cela, il faudrait procéder à la collecte de nouveaux exemples réels positifs et négatifs afin d'améliorer les bases d'apprentissage. Il est aussi évident qu'il faudrait pouvoir reconnaître différents types de panonceau (autre que le seul panneau de sortie de voie) passant là aussi par la collecte d'exemples afin d'enrichir les bases d'apprentissage, et de pouvoir peut être différentier les types de fin de limitation (i.e. pouvoir différentier une fin de limitation 30 à une fin de limitation 50 par exemple).

Afin d'améliorer la robustesse de ce framework face aux rotations des panneaux et à leurs multiples résolutions possibles (les panneaux étant plus ou moins loin) dans l'image, une première investigation a été portée sur l'étude de l'apprentissage / reconnaissance non pas sur l'image originale du panneau, mais sur un ensemble de descripteurs (signature) invariants aux transformations extraits de ces panneaux. De nombreux descripteurs ont ainsi été développés depuis les travaux de Hu en 1962 (Hu, 1962) sur les moments invariants (comme les moments de Zernike qui sont très utilisés dans la littérature). Les travaux de (Adam, et al., 2001) ont montré que la transformée de Fourrier-Mellin donne de bien meilleurs résultats que les autres signatures généralement employées dans la littérature pour la reconnaissance de caractère multi-orienté multi-échelle avec des rotations sur leurs images jusqu'à 180° (ce qui est bien supérieur à nos cas réels) et face au bruit. Le caractère insensible aux variations de rotation et de translation de cette transformée s'explique de part sa construction (Figure 97): c'est l'enchainement d'une transformée log-polaire (qui transforme les rotations et les changements d'échelle en translation, donc insensible aux rotations - Figure 98 et Figure 99) de l'image puis d'une transformation de Fourrier (insensible aux translations - Figure 100).

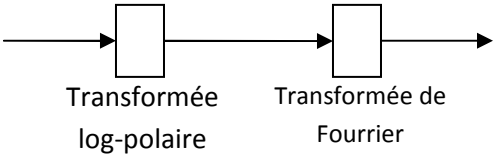



Figure 97 – La transformée de Fourier-Mellin

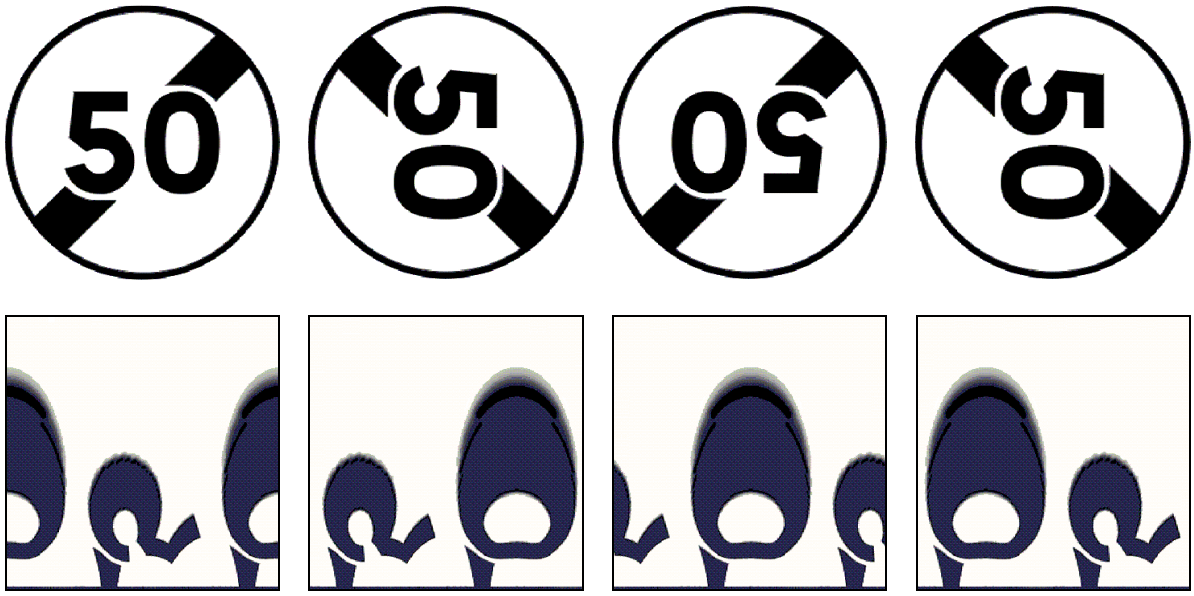


Figure 98 - La rotation dans le domaine cartésien (en haut) se transforme en une translation le long de l'axe angulaire (ici, l'axe horizontal) dans le domaine log-polaire (en bas)

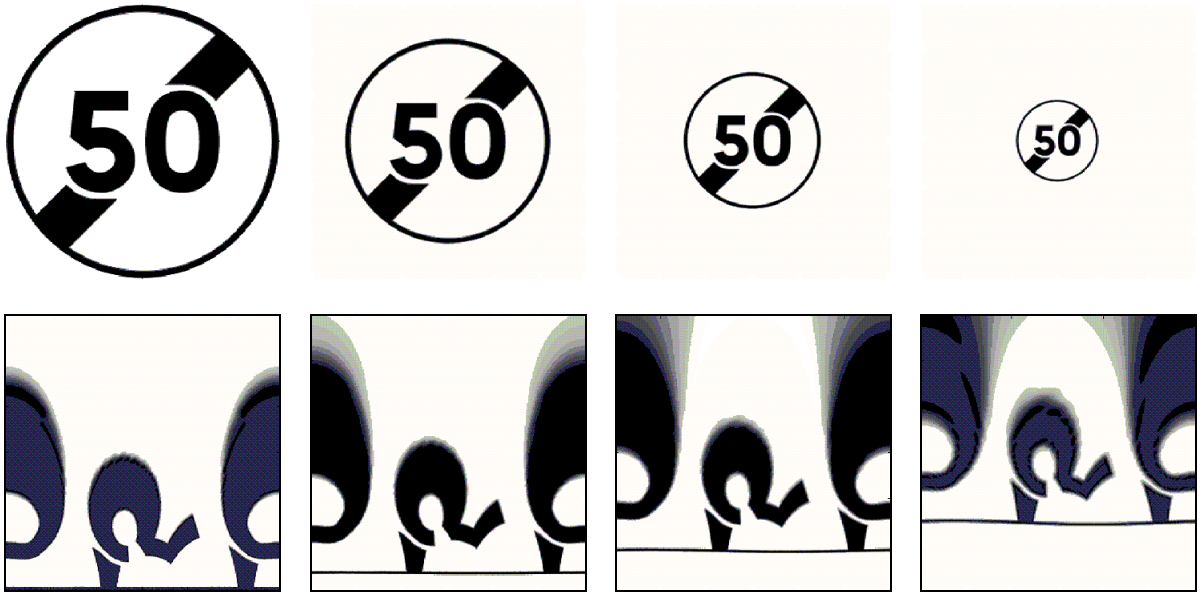


Figure 99 - Le changement d'échelle dans le domaine cartésien (en haut) se transforme en une translation le long de l'axe rayon (ici l'axe vertical) dans le domaine log-polaire (en bas)

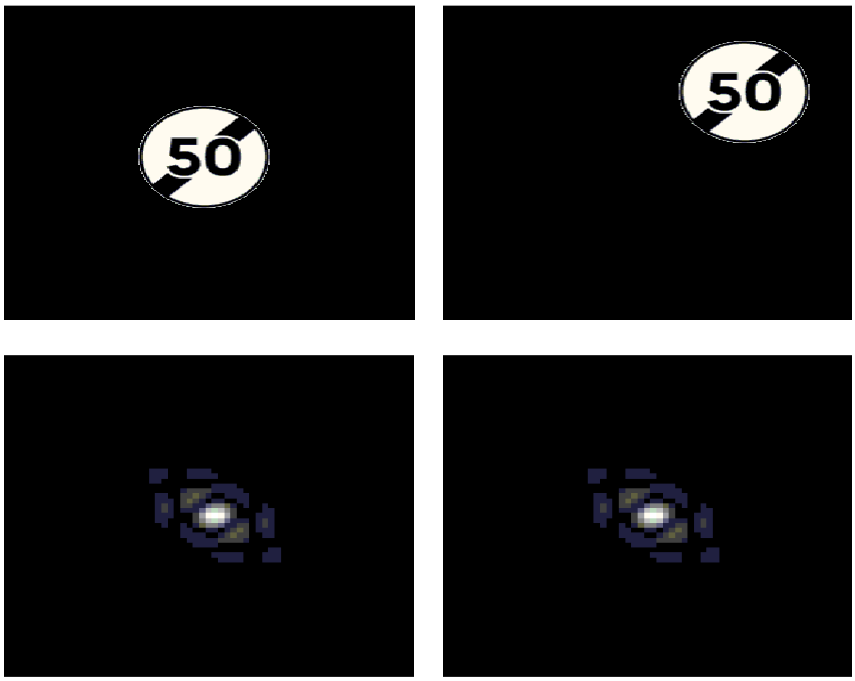



Figure 100 – Spectre de la transformée de Fourier (en bas) est insensible aux translations des images (en haut)

Pour améliorer notre framework de reconnaissance l'idée est donc d'utiliser la transformée de Fourier-Mellin de nos images comme base d'apprentissage. Une première comparaison en utilisant un réseau de neurone de type perceptron multicouche sur les 2 bases (la base d'images originales et la base d'images avec une transformée de Fourier-Mellin) montre que l'utilisation de cette transformée donne de bien meilleurs résultats face aux rotations (Tableau 12) et au bruit (Tableau 13).

| Angle de rotation | Taux de bonne classification avec transformée de Fourier-Mellin | Taux de bonne classification sur les images brutes |
|---|---|---|
| 6° | 92% | 90% |
| 14° | 91% | 81% |
| 22° | 91% | 36% |

Tableau 12 - Comparaison de l'apprentissage face aux rotations avec et sans transformation de Fourier-Mellin

| Var | Taux de bonne classification avec transformée de Fourier-Mellin | Taux de bonne classification sur les images brutes |
|---|---|---|
| 10 | 91% | 40% |
| 20 | 86% | 32% |
| 30 | 83% | 25% |

Tableau 13 - Comparaison de l'apprentissage face au bruit avec et sans transformation de Fourier-Mellin

Malheureusement, il n'est pas possible d'inclure cette perspective d'amélioration tel quelle. En effet, l'inconvénient majeur de cette technique réside dans le choix du centre de transformation du repère cartésien en repère log-polaire car cette première transformée n'est pas insensible aux translations. Ainsi, comme le montre Figure 101, si on utilise le centre de l'image, le résultat de la transformation log-polaire ne sera pas le même avec une translation de l'image originale. Une méthode pour contourner ce problème consiste à calculer le centre de masse (centre de gravité) de l'image et à utiliser celui-ci comme point origine de la transformation (Figure 101). Cependant, calculer le centre de masse d'une image réelle de la détection d'un panneau, qui peut être plus ou moins bien centré, et qui contient forcément une image de l'environnement derrière lui (dû au fait de sa forme circulaire et non rectangulaire) est quasiment impossible. Les résultats de cette technique sont donc certes très prometteurs, mais elle n'est pas utilisable directement tel quel dans le système.

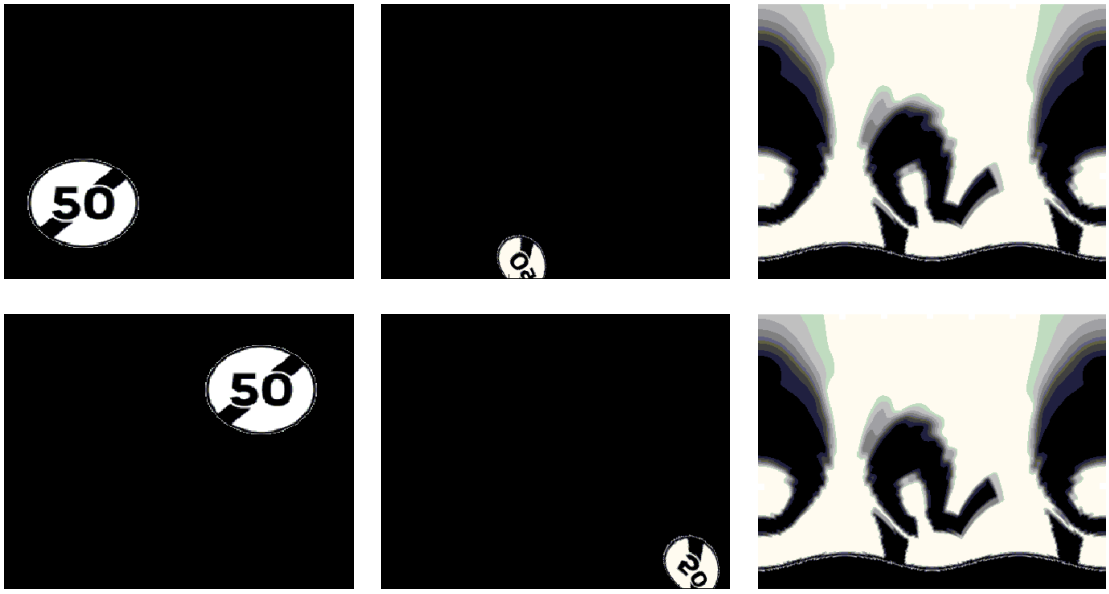


Figure 101 - Comparaison de la transformée log-polaire par rapport aux translation de l'image originale en fonction du point origine (à gauche) - résultat en prenant comme originie le centre de l'image (au centre) - résultat en prenant le centre de masse (à droite)

Enfin, le vrai problème d'un système de détection et de reconnaissance purement visuel des panneaux de limitation de vitesse, bien que prenant en compte tous les types de panneau (début de limite, fin de limite et panonceau) réside dans l'interprétation de la reconnaissance. Par exemple, dans le cas de la Figure 102, la voiture roule sur une autoroute française, la limitation est donc de 110 ou 130 km/h, mais le système détecte et reconnaît le panneau de sortie de voie. Bien que le panonceau stipulant une limitation pour la voie de sortie soit aussi reconnu, le système seul (i.e. non couplé à d'autres algorithmes) est incapable de dire si la voiture emprunte ou non cette sortie, et

valide bien malgré lui cette limitation de 90 km/h. Il est donc clair que d'autres algorithmes / informations doivent être fusionné au système afin de pouvoir effectuer une prise de décision intelligente. Dans notre exemple (Figure 102), il suffirait par exemple d'utiliser (ie fusionner) les résultats d'un algorithme de détection de ligne afin que le système puisse « savoir » qu'il n'est pas sur la voie de sortie.


Figure 102 - Cas d'un manque d'interprétation du système de détection visuel seul

Il peut aussi survenir que le système ne connaisse pas la limitation en cours (i.e. n'ait pas encore vu de panneau de limitation de vitesse), par exemple, quand le conducteur démarre sa voiture et commence à rouler, le système s'initialise et ne sait qu'elle limitation est en cours ; ou à cause d'une simple occultation d'un panneau par un autre véhicule. Pour palier à cet inconvénient, il est envisageable d'utiliser l'information de positionnement provenant du GPS et d'un capteur cartographique afin de savoir où le véhicule se situe et de connaître la limitation de vitesse à appliquer (soit parce qu'elle est renseignée dans la base cartographique, soit parce qu'elle peut être induite via des règle : en France, la limitation sur autoroute est de 110 ou 130km/h).

Le chapitre suivant présente l'utilisation du capteur cartographique afin d'améliorer le système de reconnaissance de limitation de vitesse.

# Capteur cartographique

## 2.1. Motivations

Un capteur cartographique (i.e. une base de données cartographique couplée à un système de localisation – GPS) peut fournir un grand nombre d'informations statiques à un système d'aide à la conduite automobile relatives à la route mais aussi à l'environnement et à l'infrastructure proche ou « à venir ». Cette possibilité d'aller faire des requêtes en aval dans la cartographie numérique, pour annoncer au conducteur un virage en épingle à cheveux non visible sur la droite ou encore l'approche d'un carrefour dangereux est communément appelée « horizon électronique ». Il est ainsi possible par exemple, par l'utilisation d'un capteur cartographique, de connaître :

- La topologie de la route (i.e. le graphe routier et donc les prochains carrefours et le plus court chemin d'un point A à un point B).
- Des informations relatives à chaque route / portion de route : le type de route (urbaine, rurale, route nationale, autoroute,…), la vitesse limite, le rayon de courbure (de façon statique dans la base ou par calcul dynamique sur la topologie),…
- Des informations relatives à une position donnée (Point d'intérêt – POI) : l'emplacement des radars, des stations essences,…

L'utilisation d'aide à la conduite la plus couramment effectuée de nos jours d'un tel capteur embarqué au sein des véhicules est le guidage routier (Figure 103) : guider l'utilisateur sur le trajet à effectuer pour se rendre à sa destination[1] et l'aider par exemple, si besoin est, à trouver la station essence la plus proche de sa position. Les moyens techniques proposés de nos jours, permettent une aide « avancée » à la conduite en proposant un guidage (trajet) en temps réel : ce trajet est recalculé / corrigé si l'utilisateur n'emprunte pas le chemin proposé. Récemment, ces systèmes se sont vu améliorer par la prise en compte de l'état du trafic en temps réel et des statistiques d'utilisation des routes selon les heures de la journée. Ils proposent désormais une aide à la conduite encore plus avancée (intelligente ?) qu'elle n'était avant, en fournissant des trajets différents en cas de bouchon et selon l'heure de la journée.

Ces systèmes se veulent aussi des aides à la sécurité en permettant par exemple d'avertir le conducteur de ses éventuels excès de vitesse. Cette information issue de ces appareils autonomes, n'est malheureusement pas très fiable dû au fait que la vitesse est généralement calculée par rapport aux positions GPS successives, et que ces positions souffrent d'une certaine imprécision (de 10 mètres pour un récepteur GPS commun). La localisation du véhicule subit aussi cette imprécision fortement et il arrive souvent que le « GPS » mette un certain temps avant de « comprendre » que le véhicule n'a pas emprunté le bon itinéraire.

[1] Système qui est simplement appelé dans le commerce, et ce par abus de langage, GPS


Figure 103 - Ancienne et nouvelle interface de navigation du capteur cartographique TomTom : Les « GPS » se veulent de plus en plus réalistes

D'autres systèmes avancés d'aide à la conduite automobile ont été proposés ces dernières années dans la littérature. Ils utilisent des informations fournies par un tel capteur afin d'améliorer la sécurité de conduite. Par exemple, (Polychronopoulos, et al., 2005) utilise ce capteur pour l'aide au maintien du véhicule dans la voie via la prédiction de la géométrie à venir de la route. Dans le même ordre d'idée, les systèmes d'avertissement pour virage, se basant sur un capteur cartographique, proposent d'anticiper l'approche des virages afin de définir la vitesse appropriée, d'adapter l'éclairage des phares (Lauffenburger, et al., 2007), etc. Un autre facteur accidentogène se situe au niveau des carrefours. Ainsi (Revue, et al., 2002) proposent d'avertir le conducteur des risques potentiels à l'approche de carrefour en se basant toujours sur l'analyse du graphe routier courant.

Un panel des applications possibles de l'utilisation d'un capteur cartographique (seul) est proposé et résumé par la société Navteq (éditeur de base de données routière cartographique) dans le schéma de la Figure 104 ci-dessous.


Figure 104 - Applications possibles aux systèmes ADAS d'un capteur cartographique (source NAVTEQ)

Ainsi, il est clair qu'il est possible et nécessaire d'exploiter les informations d'un capteur cartographique afin de prédire l'environnement « statique » futur pour réaliser des applications d'aide avancée à la conduite automobile (en anticipant la géométrie de la route par exemple, l'approche d'un virage ou d'un carrefour). En effet, l'anticipation basée sur la connaissance de la topologie et des autres informations fournies par un capteur cartographique est primordiale pour créer des systèmes ADAS intelligents.

Dans notre cas, et comme précisé au chapitre précédent, notre système de détermination de la limitation de vitesse visuelle souffre de certains problèmes et peut ne pas connaître la limitation en cours, ou ne pas pouvoir prendre une décision sur la prise en compte ou non d'un panneau, ne connaissant pas le positionnement du véhicule sur la route. Il est donc nécessaire de « fusionner » les informations issues d'un capteur cartographique pour réaliser un système intelligent de détection des limitations de vitesse.

## 2.2. Etat de l'art

### 2.2.1. Projets ADAS utilisant la cartographie

Comme le souligne (Bastiaensen, 2004), qui dresse un état de l'art des projets ADAS utilisant un capteur cartographique, une cartographie digitale peut être utilisée comme :

- Un capteur prédictif et une source d'information additionnelle (pour améliorer l'interprétation, pour de la prévention des tournants, de la reconnaissance des panneaux routiers, des applications de navigation,…)
- Une mémoire
- Un filtre (par exemple, bloquer les appels entrants dans des zones accidentogènes)
- Un moyen de créer de nouveaux services

Les capteurs cartographiques ont donc un potentiel très important pour des applications de sécurité routière. Ainsi, depuis quelques années, de nombreux projets sur l'étude et la réalisation de système avancée d'aide à la conduite automobile utilisant un tel capteur ont vu le jour, tels que les projets :

- NextMAP (2000 - 2001) [Navteq, TeleAtlas, BMW, Fiat, Daimler Chrysler, Jaguar, Renault, ERTICO] : Les objectifs de ce projet étaient d'étudier la faisabilité technique et économique de la création d'une base de données cartographique enrichie spécialement pour des projets ADAS et de développer et proposer des extensions au standard GDF. Leurs conclusions sont qu'il est techniquement possible d'utiliser de telles bases cartographiques pour des projets ADAS et que cela permet d'avoir de réelles améliorations pour ces systèmes. [sources (Bastiaensen, 2004)].

- ActMAP (2002 – 2004) [ERTICO, TeleAtlas, Siemens VDO, BMW, Daimler Chrysler, NAVIGON, Fiat, Navteq] : Comme son nom l'indique le projet ActMAP (Actual and dynamic MAP) a pour but d'examiner, de développer, de tester et de valider des méthodes pour renseigner dynamiquement des bases de données cartographiques utilisées par des applications ADAS embarquées dans des véhicules. Ils ont validés leur approche par différentes applications ADAS dont notamment l'échange dynamique des changements de limitations vitesse et de topologie de la route à cause de zones de travaux. Afin de parvenir à leurs buts, ils ont notamment dû développer un format d'échange de données standardisé permettant de traduire les formats propriétaires des deux fournisseurs de cartes et d'utiliser ces données de façon transparentes dans leurs applications. [sources : (ERTICO, et al., 2005)].

- ADASIS (2001 – aujourd'hui) [ERTICO, Navteq, TeleAtlas, BMW, DaimlerChrysler, Ford, Honda, Mitsubishi, Nissan, Opel, PSA, Renault, Siemens, Toyota, Volkswagen, Volvo,…] : Ce projet initié par NavTech et qui est maintenant un forum de ERTICO est partit du constat que les bases de données cartographiques ne sont pas accessibles par d'autres applications que le système de navigation et que les données y sont enregistrées dans le format propriétaire de ce système de navigation. Et que, de plus, les systèmes ADAS ont besoins d'accéder et d'utiliser ces données cartographiques afin d'améliorer leurs performances ainsi que pour développer de nouvelles fonctionnalités. De ce constat, de multiples objectifs ont été définis, dont notamment :
    a. Définir, entre les différentes professionnels membres, un modèle et une structure de données ouverts et standardisés pour représenter les données cartographiques au voisinage de la position du véhicule.
    b. Définir une API ouverte et standardisée pour permettre aux applications ADAS d'accéder à l'horizon électronique du véhicule et aux données relatives à la position du véhicule.

- PreVENT / MAPS&ADAS (2004 – 2007) : Ce sous-projet (MAPS&ADAS) du projet PreVENT (55 partenaires, 10 pays représentés, 15 sous-projets) est inspiré par les besoins identifiés lors du forum ADASIS en ce qui concerne l'utilisation de la cartographie numérique comme capteur primaire et ou secondaire pour les systèmes ADAS. Leur conclusion est que l'accès aux données cartographiques autres que celles réservées à la navigation nécessite la mise au point d'une interface normalisée afin d'éviter la mise en place de solutions spécifiques dépendantes des OEMs ou des fournisseurs d'applications. Cela permettrait de réduire les coûts de mise en œuvre et une commercialisation plus rapide. De plus, la production de cartes ADAS nécessite un approvisionnement en attributs ADAS. De nouvelles méthodes ont ainsi été présentées, pour l'acquisition d'attributs ADAS permettant de proposer un système rentable de cartes numériques précises et mises à jour régulièrement qui soient conformes aux conditions d'application des systèmes ADAS. L'application présentée comme preuve de conception était un avertisseur d'excès de vitesse et de points névralgiques. [sources : (Fallou, et al., 2008) et (Daimler, 2008)].

- SafeMap (2003 – 2005) [TeleAtlas, Navteq, Bast, LCPC, Renault, DaimlerChrysler, PSA, egis mobilite] : Le but de ce projet est d'estimer les possibilités de réalisation socio-économique de cartes numériques comprenant des données pour la sécurité routière en proposant notamment des fonctions de sécurité basée sur la cartographie numérique et en analysant les exigences en matière d'exactitude des cartes numériques. Un démonstrateur a été développé proposant des aides afin d'avertir le conducteur de certains dangers sur la route, comme les virages ou les points noirs, ainsi que certaines fonctions propres aux poids lourds. Leurs essais démontrent que les systèmes d'information présentent un effet positif sur le contrôle de la vitesse et sont globalement bien jugés par les usagers. [sources : (Fallou, et al., 2008)].

- EDMap (2001 – 2003) [General Motors, Ford, Toyota, Navteq, Nissan] : Ce projet vise à fournir une preuve de concept pour des applications simple de sécurité routière à base de cartographie numérique et plus particulièrement des spécifications des bases de données cartographiques ainsi que l'évaluation du projet ambitieux de réalisation de cartes très détaillées pour appuyer ces applications. Les principales applications étudiées dans ce projet sont l'assistance pour la vitesse en virage (alerte et intervention), l'assistance pour les panneaux stop (alerte et intervention), l'avertissement de feux de signalisation (alerte) et l'assistance au suivi de voie (alerte). Pour chaque application, sont étudiés : les implications cartographiques, des démonstrations, l'étude de faisabilité de la base de données cartographiques et les technologies sans fil et d'échange de données. [sources : (Fallou, et al., 2008)]

- CVIS (2006 – 2010) [Navteq, TeleAtlas, Autoroutes du Sud de la France, BMW, CNRS/Heudiasyc UTC, DaimlerChrysler, Fiat, Inria, Itempora, LCPC, Renault, Siemens, TNO, Thales, Volvo,…]
- SAFESPOT (2006 – 2010) [Navteq, TeleAtlas, Centre National de la Recerche Scientifique, CG Côtes d'Armor, Compagnie Financière et Industrielle des Autoroutes, Continental, IBEO, LCPC, Siemens, Renault, DaimlerChrysler, TNO, Volvo,…]

  Ces deux projets ont en commun le fait de vouloir développer, entre autre, la communication véhicule / infrastructure (V2I) afin d'améliorer la « vue » du conducteur sur son environnement. Cette communication leur permet d'informer le conducteur d'informations dynamiques tels que la présence de zone de travaux, d'accidents, d'éventuels ralentissements, de variation de la vitesse limite,… Ces nouveaux concepts sont appelés « carte locale dynamique » (local dynamic map). [sources : (ERTICO, 2009)].

Ces projets réalisés depuis les années 2000 dénotent certains problèmes récurrents quand à l'utilisation d'un capteur cartographique pour la réalisation d'applications ADAS. Il n'existe pas de standard de données cartographiques entre les 2 grands fournisseurs (Navteq et TeleAtlas), certains projets ayant comme tâche, afin d'arriver à leur but, de créer une interface de traduction entre des données dans un format normalisé et celles de l'un ou des deux providers. Les applications les plus recherchées sont celles traitant des aides sur la topologie du réseau routier (ex : prévention des

virages) et sur les limitations de vitesse. Enfin l'un des futurs enjeux est de pallier le problème des données statiques en créant des bases de données cartographique localement dynamique en utilisant entre autre la communication V2I (véhicule à infrastructure).

### 2.2.2. Fusion cartographie / vision pour aide à la limitation de vitesse

La fusion des informations d'un capteur cartographique avec d'autres informations dynamiques (issues de capteurs extéroceptifs) n'a été que très récemment (et donc très peu) étudiée. Par exemple, (Revue, et al., 2002) fusionnent les informations issues d'une caméra, d'un radar et d'un capteur cartographique afin d'informer des risques de collisions avec d'autres véhicules aux approches des carrefours et des virages. De la même manière (Gehrig, et al., 2003) fusionnent les informations cartographiques pour orienter la recherche par caméra de facteurs accidentogènes aux approches de carrefour.

En revanche, l'utilisation d'un capteur cartographique couplé à un système visuel (une caméra) pour la reconnaissance des limitations de vitesse n'est quasiment jamais envisagée dans la littérature. Cela est sans doute dû au fait que l'information sur la limitation de vitesse est considérée soit comme déjà existante dans les bases de données cartographiques, soit au contraire pas du tout. Cependant ces dernières remarques ne sont pas très perspicaces. En effet, l'information statique d'une telle base n'est jamais conforme à la réalité dont les limitations peuvent changer. Il faut donc mettre à jour la base cartographique régulièrement pour suivre l'évolution des restrictions sur les routes. De plus, elle ne comportera jamais les limitations temporaires (en cas de travaux par exemple) ni les limites variables (panneau LED) qui sont de plus en plus utilisées pour réguler le trafic sur les routes. Cela dit, au regard des informations secondaires que peut fournir un tel capteur, par exemple, si on considère que la limitation de vitesse dépend le plus souvent du contexte géographique (zone urbaine, extra-urbaine,…) et sur les cas particuliers de conduite (voie d'entrée ou de sortie d'autoroute, approche d'une école,…), il est alors possible d'utiliser cette information fournie par le capteur cartographique pour déduire la limitation de vitesse dépendant de la situation courante et coupler / fusionner cette déduction (croyance) avec notre croyance dans la détection issue d'un autre capteur (une caméra par exemple).

Dans les rares travaux de la littérature traitant ce sujet (Lauffenburger, et al., 2007) (Janssen, et al., 2004) (Bahlmann, et al., 2008) (Nienhüser, et al., 2009), la fusion des deux sources (cartographie et vision) est effectuée dans le cadre de la théorie des croyances (la théorie de Dempster-Shafer). Dans de tels systèmes, chaque information issue des 2 capteurs se voit attribuer un coefficient de confiance (une masse de croyance). Ces 2 informations peuvent ainsi être fusionnées plus ou moins intelligemment en prenant en compte d'autres types d'informations (critères) issus du capteur cartographique ou de la vision (type de route, heure de la journée,…) donnant plus ou moins d'importance (i.e. ayant une croyance plus ou moins grande) aux détections de l'un ou de l'autre capteur selon le contexte.

Par exemple dans (Lauffenburger, et al., 2008) la masse de croyance de chacune des détections (les *éléments focaux*) de la vision correspond à leur niveau de confiance directement donnée par l'algorithme de détection visuelle (qui est elle-même calculée à partir de la confiance calculée par le réseau de neurone utilisé pour la classification du panneau et du nombre de fois que le panneau a été vu par le suivi temporel). Il est aisé de comprendre que la détection par vision puisse retourner plusieurs résultats (i.e. avec des fausses détections). Cependant, afin de pouvoir effectuer la « fusion » il est nécessaire que le capteur cartographique donne lui aussi plusieurs résultats (*éléments focaux*). Or un capteur cartographique ne donne généralement qu'une limitation de vitesse pour un segment routier. Ainsi (Lauffenburger, et al., 2007) définit les éléments focaux (i.e. les potentielles limitations de vitesse) du capteur cartographique comme étant ceux pouvant être retournés si le système s'était trompé (par exemple avec une erreur de positionnement). Par exemple si la navigation stipule que la limitation de vitesse est de 110 km/h, les éléments focaux seront 110, 50 et 70. La masse de croyance attribuée à chacun de ces éléments focaux dépend de divers critères (C1, C2,…) prenant en compte l'erreur de positionnement du GPS, le type de route emprunté,… (Figure 105). La masse totale de chaque élément (i.e. la fusion) est calculée via la *loi de fusion conjonctive* énoncée dans la théorie de Dempster Shafer. L'élément focal (i.e. la limitation de vitesse) ayant la plus grosse masse (i.e. dont on a la plus grande croyance) est validé comme étant la limitation de vitesse à appliquer.

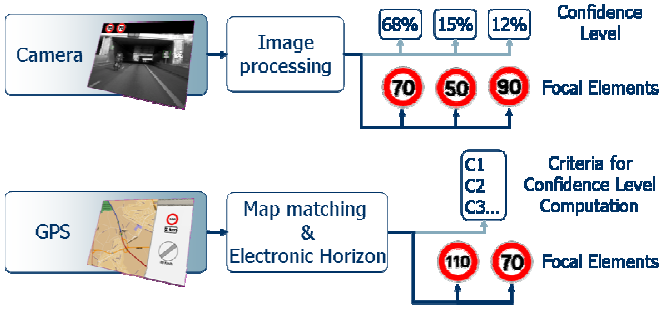



**Figure 105 - Détermination des éléments focaux dans une approche de fusion par la théorie des croyances**

Dans leurs travaux, (Nienhüser, et al., 2009) essayent de prendre en compte plus d'information dans le processus de fusion par la théorie des croyances en jouant sur le fait que les deux capteurs ont des comportements très complémentaires : la confiance à avoir dans chacun des capteurs doit dépendre des conditions d'éclairage. C'est ainsi qu'ils apportent plus de confiance aux éléments focaux issus de la vision quand il fait beau, quand une zone de travaux est détectée ou quand il s'agit d'un panneau à message potentiellement variable (panneau LED), tandis qu'ils accordent plus de confiance aux informations issues de la navigation dans les autres cas (de nuit, quand il y a du brouillard,…). Tous ces critères environnementaux sont évalués en temps réel à partir de la caméra pour agir sur le calcul des masses de croyance des divers éléments focaux.

Ces quatre travaux s'accordent tous à dire que l'utilisation unique d'un seul capteur (caméra ou cartographie) n'est pas suffisante : la cartographie ne prend pas en compte les zones temporaires (les travaux), les zones à messages variables (les panneaux LED) et les évolutions des limitations sur les routes, tandis que la caméra peut ne pas connaître la limitation de vitesse actuelle (quand un

panneau est non vu, en cas de forte dégradations climatique – brouillard, pluie forte,…). Ils concluent tous sur la nécessité de l'utilisation complémentaire des deux sources afin de fusionner leurs détections et d'en déduire la limitation de vitesse.

Il est donc clair qu'un capteur cartographique est d'une réelle nécessité dans notre système. La section suivante décrit les outils cartographiques existants pouvant être utilisés et intégrés dans un système ADAS.

## 2.3. Outils existants

### 2.3.1. Les Systèmes d'Information Géographique

Un système d'information géographique (SIG) permet d'organiser et de présenter des données spatialement référencées. Son rôle est de proposer une représentation plus ou moins réaliste de l'environnement spatial en se basant sur des primitives graphiques telles que des points, des vecteurs, des polygones ou des maillages,…. À ces primitives sont associées des informations attributaires telles que la nature (route, voie ferrée, forêt, etc.) ou toute autre information contextuelle (nombre d'habitants, type ou superficie d'une commune par exemple). Typiquement, les SIG doivent permettent d'effectuer des recherches sur les données attributaires basées sur la localisation (filtrage spatial) et vice versa, de rechercher seulement les informations géographiques qui ont certains attributs.

Il n'existe que 2 formats possibles pour la représentation des données dans les SIG :

- Le format matriciel (*raster model*) est la représentation de la zone géographique par un pavage ou chaque cellule représente une valeur. Les cellules sont généralement représentées par des carrés (par exemple les pixels d'une image) mais en principe elles peuvent être de n'importe quelle forme. Par exemple, il est possible de représenter l'élévation d'une zone géographique par une image en niveau de gris de cette zone ou chaque valeur de pixel représente l'altitude (Figure 106). Pour un rendu plus réaliste, un SIG peut afficher par superposition plusieurs modèles matriciels (par exemple une orthophoto et un modèle numérique de terrain - Figure 107).

- Le format vectoriel (*vector model*) quant à lui contient des objets individuels qui représentent des phénomènes réels. Les composants spatiaux de chaque objet sont composés de points, de lignes, d'aires (des polygones) et de volumes. Seules les coordonnées des points particuliers (les extrémités des segments, les sommets des polygones,…) sont enregistrés dans le SIG faisant de cette représentation un format plus compact que le modèle matriciel (i.e. aucune information géographique n'est enregistrée là où il n'y a pas d'objet). L'aspect spatial d'une donnée vectorielle peut être traité comme un graphe avec toutes les propriétés et opérations définies dans le domaine de la théorie des graphes. Les

objets ont généralement des attributs associés (à l'instar de la valeur d'une cellule pour le format matriciel) permettant à l'utilisateur d'effectuer différentes requêtes sur les données.

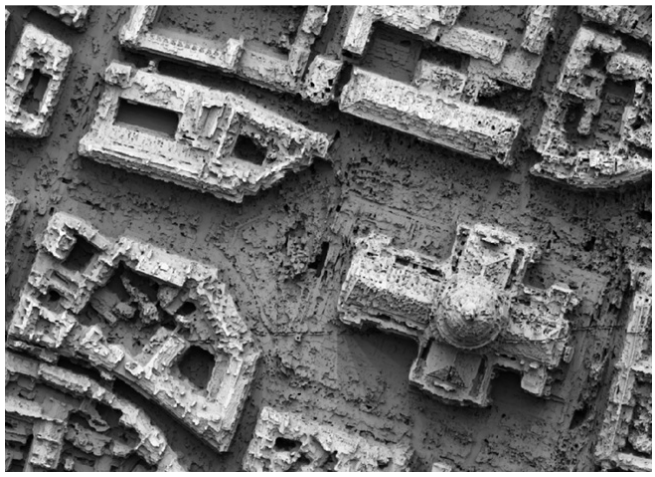


Figure 106 - Exemple de données matricielle : Modèle numérique de surface - MNS (source IGN projet TerraNumerica)

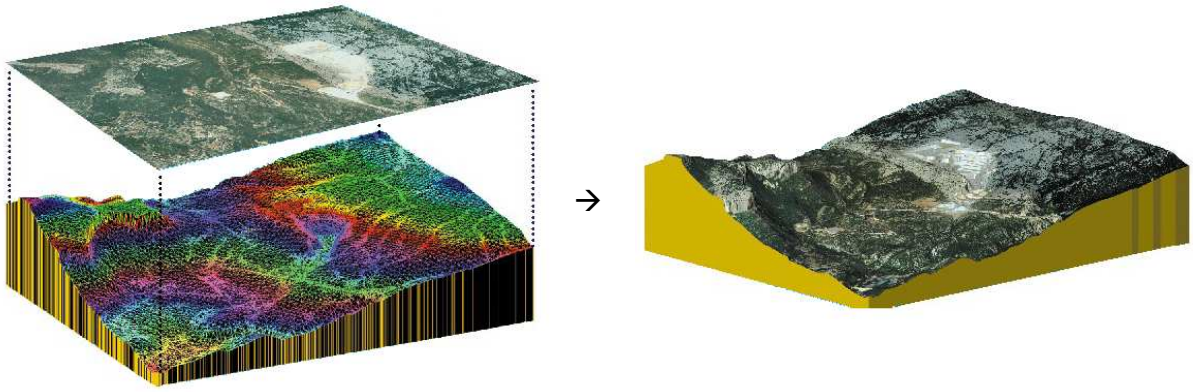


Figure 107 – Utilisation de plusieurs modèles matriciels : Modèle numérique de terrain (source projet TerraNumerica)

L'information géographique dans un SIG peut donc être représentée par cinq types de primitives différentes (Figure 108) : le point, la ligne, le polygone, le relief et les objets 3D.

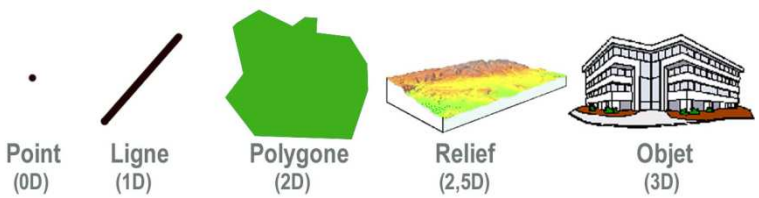



Figure 108 - Primitives d'un SIG

## 2.3.2. Les bases de données cartographiques routières

Ces bases de données représentent le réseau routier adjoint de caractéristiques. Elles sont couramment représentées (Figure 109) par un modèle vectoriel sous forme de graphe, comprenant des nœuds (points), des liens (des segments) et des aires (des polygones). Chacun de ces 3 éléments pouvant posséder des attributs (coordonnées, adresses, type de route, limitation de vitesse,…). D'autres éléments peuvent être inclus comme des points d'intérêts, la forme des bâtiments et les zones administratives (i.e. arrondissement, département, région, pays).

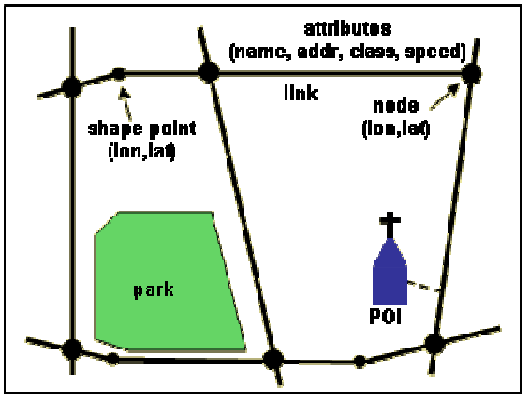



Figure 109 - Entités et attributs d'une base de données cartographique routière

Chaque nœud représente un point à la surface de la Terre et est représenté par une paire de coordonnées GPS latitude / longitude. Chaque lien représente un morceau de route entre deux nœuds. Un lien pouvant être un simple segment (correspondant à une portion droite d'une route) ou une courbe dont la forme est généralement décrite par des points intermédiaires (appelés *shape points*) le long de ce lien. Ces points intermédiaires sont représentés comme les nœuds par un couple de coordonnées GPS latitude / longitude mais ils n'ont pas pour but de représenter une interconnexion entre deux routes (à la différence des nœuds).

La question qui se pose est de savoir où trouver de telles bases en vue d'être utilisées et exploitées dans un premier temps au sein de ces travaux de recherche, puis pour être éventuellement intégrées facilement au sein des différents systèmes ADAS mis en œuvre au laboratoire.

Aux Etats-Unis, il est possible de récupérer gratuitement ce qui est appelé les *TIGER files (Topologically Integrated Geographic Encoding and Referencing)*, des fichiers contenant le graphe routier des USA contenant certains attributs (nom, plage d'adresse,…) et couvrant tout le pays. Cette base de données a été entièrement développée par l'administration américaine, mais cette dernière ne fournit pas de logiciel pour les exploiter (il faudra donc se tourner vers des logiciels tiers ou redévelopper son propre logiciel afin d'exploiter ces données).

En Europe, il n'existe pour l'instant rien de similaire (i.e. des bases de données cartographiques mises gracieusement à disposition par les administrations) et il faudra donc se tourner vers des fournisseurs tiers spécialisés dans la création de base de données routière. Cependant cela risque de

changer dans un avenir proche. En effet, le parlement européen a publié le 25 avril 2007 dans le journal officiel la directive INSPIRE (Infrastructure for Spatial Information in the European Community). Celle-ci vise à établir à l'horizon de 2014, à harmoniser et à mettre à disposition par les états membres leurs informations géographiques et notamment des « réseaux de transport » (dont les réseaux routiers). Cela dit, cette directive n'impose pas de nouvelles collectes de données géographiques.

Les fournisseurs de cartes ne sont pas nombreux. Seules les sociétés Navteq et TeleAtlas fournissent des cartes routières détaillées au niveau international. Ces 2 sociétés ont été créées respectivement en 1985 et 1984. Elles ont toutes les deux été rachetées récemment : Navteq par Nokia en 2007 et TeleAtlas par TomTom également en 2007. L'historique de ces deux sociétés montre donc clairement que l'engouement dans la création de ces bases est un enjeu majeur pour la société d'aujourd'hui. Au niveau national (français), il n'y a guère que l'IGN (Institut Géographique National) qui créé et revend des bases de données géographiques.

Le principal problème de ces données acquises par ces sociétés réside dans le fait qu'elles coûtent très cher. Par exemple, l'IGN fournit la base de données routière GEOROUTE qui couvre la France entière pour la somme de 40 000 euros (Brunel, 2005). Navteq propose l'exploitation de ses bases de données pour 25 000 euros par an. On comprend aisément ces prix quand on sait que TeleAtlas y a investi en 20 ans près de 600 millions d'euros (Dessalle, 2006).

L'exploitation des données peut se faire de différentes façons. Il est possible de les acquérir directement auprès de ces 3 gros fournisseurs (Navteq, TeleAtlas ou l'IGN ; aux prix indiqués ci-dessus) qui transfèreront leurs données (plusieurs giga octets) dans un format texte ouvert et lisible (généralement le format GDF - Geographic Data Files, norme ISO). Le projet européen INSPIRE semble aussi se baser sur ce format pour l'harmonisation des données géographiques des réseaux routiers. Malheureusement, ce format n'est pas fait pour être utilisé directement et il devra donc être nécessaire de retraiter ces données en vue d'être intégrées dans un système ADAS. Il est en revanche possible d'acheter auprès de revendeurs des données directement post-traitées munies d'un SDK (Software Developpement Kit) afin d'être exploitées facilement. Parmi les SDK les plus connus, on peut citer (avec leur prix pour en licence monoposte) : MapInfo (3,25k€), MacMap (5k€), AdasRP de la société Navteq (50k€),… L'inconvénient de ces SDK est simple : ce ne sont pas de vrais SDK indépendants, ils sont fournis sous forme d'une plateforme graphique de visualisation des données (Figure 110) qu'il est impossible de ne pas utiliser et sont donc des logiciels indépendants dans lesquels il serait possible d'adjoindre des « plugins » développés soi-même, et non de vrais SDK intégrables dans des logiciels externes. Il en résulte qu'une intégration (propre) dans sa propre plateforme de développement n'est donc pas possible et rend la possibilité d'accès aux données très dépendant du SDK d'accès fourni avec ces données (par exemple il pourrait être difficile de retrouver certaines données selon certains critères, comme retrouver toutes les routes qui sont à moins de 100 mètres d'un point donné).

Le constat est simple : les données brutes coûtent très cher et les SDK des revendeurs ne sont pas appropriés au vu des limitations d'accès aux données induites par leur utilisation et au vu de l'impossibilité de l'intégration propre dans un logiciel tiers.

Ce constat avait aussi été réalisé en 2004 par Steve Coast qui lança le projet OpenStreetMap : un wiki de la cartographie où tout le monde peut ajouter des informations géolocalisées (des routes, des zones,…) dans une base de données communautaire. Depuis 2004, OpenStreetMap est en constante évolution et s'est enrichi de nombreuses données. L'intérêt de ce projet en Europe n'est plus à démontrer et c'est ainsi que, par exemple, la société néerlandaise Automotive Navigation Data, qui est un fournisseur local de cartographie aux Pays Bas, a gracieusement fourni sa cartographie des Pays-Bas. Certaines villes donnent aussi des fonds à OpenStreetMap afin qu'ils organisent des Mapping Party (weekend de rencontre entre utilisateurs, ou chacun se voit attribué une partie de la ville à géoréférencer) pour créer la cartographie de la ville.

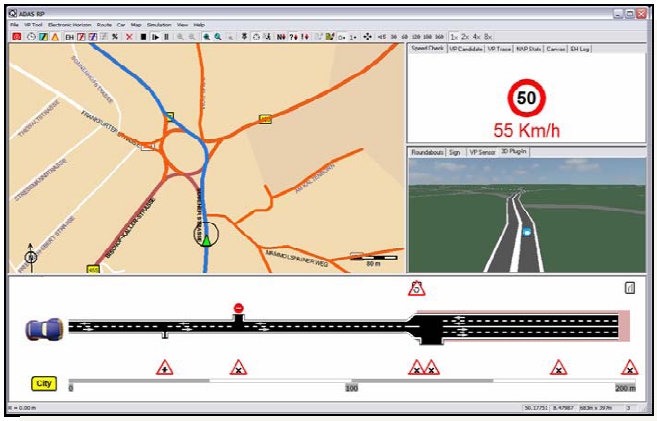



Figure 110 - SDK AdasRP et son interface graphique indissociable

La qualité des données issues d'OpenStreetMap par rapport à Navteq et TeleAtlas est relativement équivalente. En effet, ils ont la même façon de collecter les données (tous géocalisent les routes du monde entier en les parcourant munis d'un récepteur GPS et en reportant ces informations dans leur base). Cependant Navteq / TeleAtlas prévoient un certains nombre d'attributs à collecter pour chaque route, OpenStreetMap au contraire n'impose rien aux utilisateurs, d'où une plus grande flexibilité et un plus grand nombre d'attributs possibles, mais l'information peut ou non être présente dans la base.

A ce jour, OpenStreetMap couvre les grandes villes d'Europe et d'Amérique du nord ainsi que le réseau routier constitué des autoroutes reliant ces villes.

Créer son propre capteur cartographique basé sur ces données en libre accès est donc tout à fait pertinent. En effet, cela permet de s'affranchir de beaucoup de problèmes : l'acquisition (le prix) des données bruts de Navteq ou TeleAtlas, ou l'utilisation non aisée des divers SDK du marché. L'avantage majeur d'un tel développement est de pouvoir définir son propre format de stockage des données géographiques avec sa propre API (SDK) d'utilisation et donc de pouvoir créer les fonctionnalités d'accès aux données à la demande. Enfin l'intégration dans divers logiciels, et notamment au sein de $^{rt}$Maps (notre logiciel de prototypage) en serait fortement plus aisée et plus

« propre ». C'est ainsi qu'est né le projet CAORTO au sein du laboratoire décrit dans la section suivante.

## 2.4. La CAORTO

### 2.4.1. Format

La CAORTO, la cartographie du CAOR (le Centre de Robotique de l'Ecole des Mines de Paris), est ainsi définie comme étant une base de données cartographique routière se basant principalement sur des données gratuites et en libre accès (pouvant donc être mises à jour facilement). L'objectif est de concevoir une base de données du réseau routier en 2D (un graphe), et de fusionner dans le futur les travaux de l'équipe de numérisation 3D (MMS – Mobile Mapping System) du laboratoire pour y intégrer des informations 3D.

Bien sûr, le développement de ce nouveau capteur cartographique doit répondre aux problèmes des systèmes « concurrents » actuels, et vise donc à intégrer les données cartographiques du projet OpenStreetMap qui sont gratuites et en libre accès (sous licence *Creative Commons*), et fournir une API d'accès (le fameux SDK) simple d'utilisation et ouverte afin d'être facilement intégrable au sein de nos logiciels. Evidemment dans un souci de compatibilité et de portabilité l'ensemble de la CAORTO vise à être indépendante de la plateforme (donc pouvant fonctionner aussi bien sous Windows que sous Linux ou autre) et ne doit pas s'appuyer sur des bibliothèques externes (il n'est pas rare de trouver des outils développés mais qui ne fonctionnent plus car s'appuyant sur des bibliothèques externes n'existants plus). L'API d'accès aux données sera donc écrite dans un langage informatique de base (typiquement le C++) sans faire appel à des fonctionnalités avancées proposées par certains systèmes d'exploitation.

Les données OpenStreetMap ne sont pas diffusées sous le format GDF mais en XML (Extensible Markup Language), ce qui constitue des centaines de mégaoctets de données sous forme d'un fichier texte à traiter. La réalisation d'application ADAS, qui requiert évidemment que les algorithmes fonctionnent en temps réel (i.e. analyser les données dynamiques décrivant l'entourage de la voiture de façon très rapide) afin de fournir une aide quasi instantanée au conducteur, requiert donc un accès très rapide à l'information. Or, la recherche de données dans un fichier XML de grande taille est loin de répondre à cette contrainte. Il est donc nécessaire de post-traiter les données OpenStreetMap afin de définir son propre format de stockage du graphe routier.

Le schéma conceptuel de la CAORTO est décrit par la Figure 111. Un éditeur indépendant permet d'importer les données OpenStreetMap, enrichissant la base ou la mettant simplement à jour, et de modifier les données existantes de la base. Afin de rendre l'utilisation plus conviviale et plus aisée pour l'édition des données, cet éditeur repose sur une interface graphique. Là aussi, et pour répondre aux même spécifications que l'API d'accès, cet éditeur est entièrement développé en Java qui propose nativement la création d'interfaces graphiques sans dépendance vis-à-vis du système

d'exploitation. L'API d'accès (sous forme d'une bibliothèque) permet d'interroger la base et de récupérer des données afin d'être utilisées dans nos logiciels. Elle joue l'intermédiaire entre la base et les logiciels et doit permettre une utilisation aisée et simple d'accès aux données.

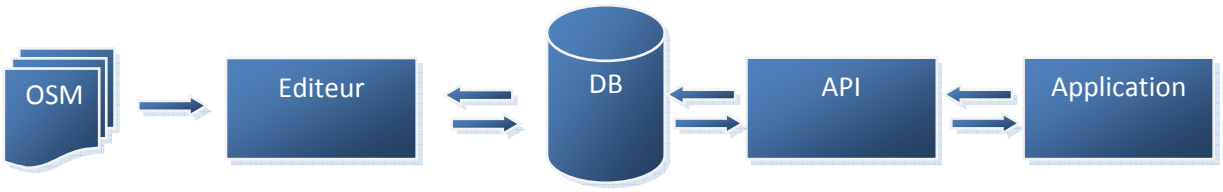



Figure 111 - Schéma conceptuel de la CAORTO

Toute la question réside dans le format de stockage des données dans la base. Il doit à la fois être facile à lire et rendre possible l'accès rapide aux informations quelque soit la zone du globe terrestre visée. La facilité de lecture par un opérateur humain n'est supportée que par les fichiers textes. Il ne faut donc pas escompter créer une base de données au format SQL où les données seraient enregistrées de façon binaire. La rapidité de recherche impose de découper et d'organiser intelligemment les données en plusieurs fichiers. Un format de stockage hiérarchique s'impose donc. Par exemple, Google utilise les QuadTree (une structure de données de type arbre dans laquelle chaque nœud peut compter jusqu'à quatre fils) pour accéder à n'importe quelle donnée de sa base géographique (des données au format matriciel, ici des images). Ainsi pour accéder à une zone sur le globe, il faut savoir à chaque niveau dans quel quart du planisphère la donnée se trouve et descendre successivement de niveau en niveau (Figure 112). L'intérêt d'une telle structure, en plus de la rapidité d'accès, est de permettre de stocker des informations macroscopiques au niveau des nœuds internes (par exemple les autoroutes) et microscopiques au niveau des branches (par exemple les petites ruelles).

La base géographique de la CAORTO n'est pas définie en QuadTree mais en « grille » (Figure 113) : chaque niveau de stockage est divisé en carrés de coté de 10°. Ainsi le premier niveau (le niveau mondial) qui fait 360° (+/-180° de longitude) sur 180° (+/- 90° de latitude) comporte 36 x 18 = 648 pavés. Les niveaux inférieurs ne comportent eux que 100 pavés (10x10). La taille du plus bas niveau à été choisie empiriquement à 0.01° de coté, ce qui correspond à un petit quartier d'une ville. Les informations à y stocker ne seront donc pas nombreuses et il sera très rapide d'y accéder. Ainsi le monde est découpé sous forme d'un arbre à 4 niveaux.

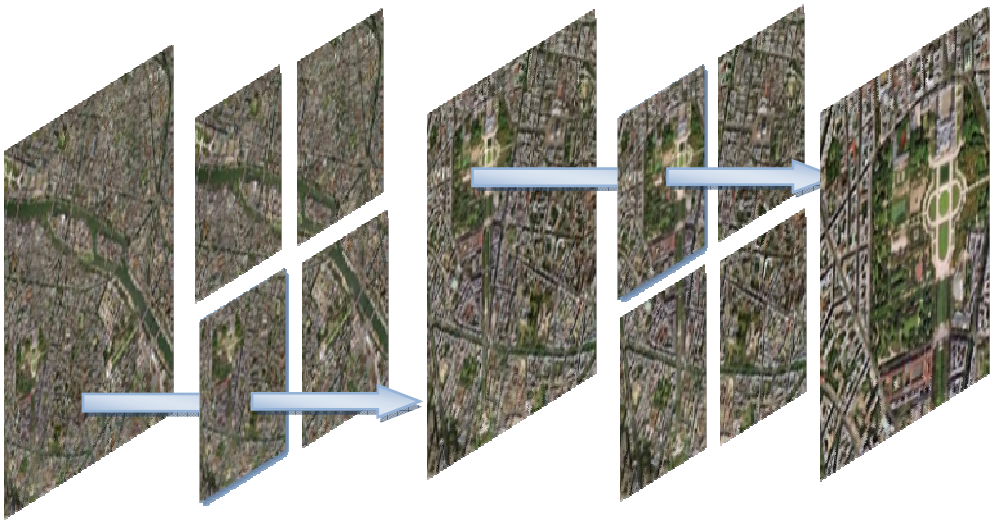


Figure 112 - Stockage sous forme de QuadTree des données matricielles chez Google Maps

D'un point de vue plus concret, chaque niveau est représenté physiquement sur le disque dur par un répertoire comportant les répertoires des niveaux inférieurs. Les derniers répertoires (ceux du niveau 4), contiennent le graphe routier (sous forme de fichier texte). Le nommage des répertoires et des sous-répertoires est corrélé avec le système de coordonnées latitude longitude (en degré) défini par l'arbre et se fait comme dans l'Exemple 1. Il est aisé de s'apercevoir que retrouver le bon répertoire où se situe une information se fait de façon quasiment instantanée (une dizaine d'opérations arithmétiques de base).

Au niveau d'une feuille est stocké le graphe routier de la zone en cours aves les attributs. Typiquement ces informations sont enregistrées dans 3 fichiers texte différents :

- un fichier référençant les nœuds, comportant pour chaque nœud un identifiant unique au sein de cette zone et son couple de coordonnées latitude / longitude ;
- un fichier pour les segments, comportant pour chaque segment un identifiant et le couple d'identifiants des 2 nœuds qui le définissent ;
- un fichier pour les attributs, comportant pour chaque attribut son type, sa valeur et l'identifiant du segment à qui il se réfère.

Le contenu de ces fichiers est complètement formaté, rendant leur lecture bien plus rapide (il n'y a pas de règle compliquée de parsing à appliquer comme il serait le cas avec le format XML générique). Par exemple, le contenu des fichiers de nœuds est ainsi sur chaque ligne : « id latitude longitude ».

Le poids total de l'ensemble de ces données au niveau d'une feuille n'est pas important. Par exemple le poids des données de la zone autour du jardin du Luxembourg, fait moins de 10ko répartis sur les 3 fichiers. Retrouver des informations dans 10Ko de données textuelles complètement formatées est aisément faisable en *temps réel* pour des applications d'aide à la conduite automobile.

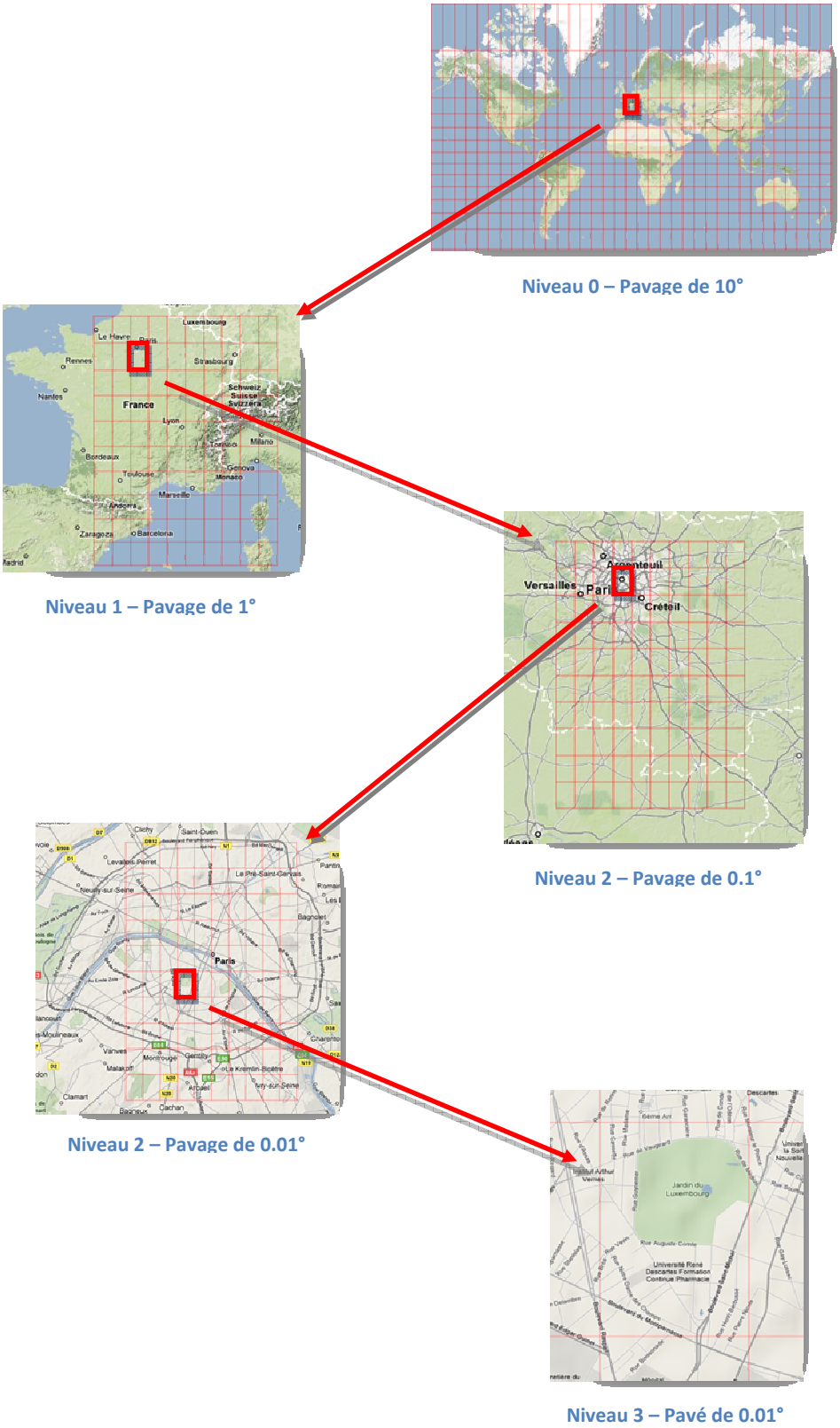



Figure 113 - Format hiérarchique de la CAORTO

On cherche une information d'un objet situé à la coordonnée (latitude, longitude) en degré de :

12.3456789 9.87654321

A noter que seuls les 2 premiers chiffres décimaux servent à l'indexation (puisque le plus grand niveau hiérarchique est constitué de carrés de 0.01 degré de côté), on peut donc retenir les coordonnés suivantes pour notre objet :

12.34 9.87

Il faut dans un premier temps normaliser ses valeurs entre de [-90;90] à [0;180] pour la latitude et de [-180;180] à [0;360] pour la longitude :

12.34 + 90 = 102.34
9.87 + 180 = 189.87

Le nom du 1er répertoire est la juxtaposition "des dizaines" de longitude et de latitude :

102.34 / 10 = 10 → nom1 = 1810
189.87 / 10 = 18

Le nom du 2nd répertoire est la juxtaposition "des unités" de longitude et de latitude :

102.34 % 10 = 2 → nom2 = 92
189.87 % 10 = 9

Le nom du 3ème répertoire est la juxtaposition "des dixièmes" de longitude et de latitude :

(102.34 * 10) % 10 = 1023.4 % 10 = 3 → nom3 = 83
(189.87 * 10) % 10 = 1898.7 % 10 = 8

Le nom du 4ème répertoire est la juxtaposition "des centièmes" de longitude et de latitude :

(102.34 * 100) % 10 = 10234 % 10 = 4 → nom4 = 74
(189.87 * 100) % 10 = 18987 % 10 = 7

Finalement, l'arborescence de répertoire pour trouver notre objet est constituée des noms trouvés :

db_path\nom1\nom2\nom3\nom4 → db_path\1810\92\83\74

**Exemple 1 - Accès une donnée dans la base CAORTO**

Le problème du stockage du graphe routier sous forme d'arbre est qu'une rue peut chevaucher plusieurs zones (passer d'une zone à l'autre, comme c'est le cas du Boulevard Saint Michel sur la Figure 114). Il faut donc au sein d'une zone que les segments puissent faire référence à un nœud qui se trouverait dans une zone voisine. C'est ainsi que les identifiants des nœuds peuvent être de 2 types :

- interne, représenté par un nombre compris entre 0 et 100000000 (non inclus)
- externe, représenté par un nombre supérieur ou égal à 100000000

Le choix d'une limite à 100000000 étant une limite arbitrairement choisie. Un identifiant externe référencie à la fois un objet d'un répertoire et le répertoire lui-même, tandis qu'un identifiant interne ne référencie qu'un objet d'un répertoire (on suppose donc que les références se font pour des objets du même répertoire). Ainsi, lors de la lecture d'un fichier, savoir si un identifiant réfère à un objet d'une même zone ou non revient à effectuer un test sur sa valeur.

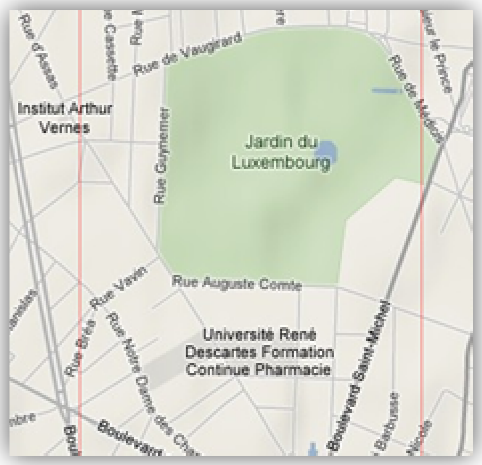



Figure 114 - Exemple de route passant par 2 zones différentes (limite des zones en rouge)

Dans l'Exemple 2 schématisant un fichier de segments de la base, le premier segment, d'indice 0, est délimité par les nœuds d'indice 0 et 1 de la même zone (du même répertoire). En revanche le second segment, d'indice 1, est délimité par le nœud 1 du même répertoire et par le nœud 8 d'un répertoire voisin identifié par « 1813282838 » (le début du nombre) correspondant au répertoire « db_path\1813\28\28\38\ ». La correspondance identifiant / chemin du répertoire est donc là encore très aisée.

| | | |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 1 | 181328283800000008 |

Exemple 2- Exemple de contenu d'un fichier de segment de la base CAORTO

## 2.4.2. Fonctionnalités

### *a) Présentation générale*

Les fonctionnalités d'utilisation sont au cœur de l'API d'accès, car c'est celle-ci qui va dialoguer avec la base et récupérer les données selon les requêtes de l'utilisateur. Les principales fonctionnalités utiles pour un système ADAS sont :

- de pouvoir récupérer le segment (la route) sur lequel on se situe (i.e. le map-matching);
- retrouver les attributs (vitesse, nombre de voies,…) d'un segment (par exemple le segment en cours) ;
- de pouvoir se « balader » virtuellement de nœud en nœud par rapport au segment courant (i.e. pouvoir récupérer la position de la prochaine intersection qui peut être un carrefour ou tout simplement l'embranchement avec une voie de sortie sur une autoroute) ;
- de pouvoir retrouver les segments (les routes) dans un périmètre donné autour d'une position
- de pouvoir calculer des distances entre 2 coordonnées GPS (la distance donnant l'indication de risque)

Le routing (retrouver le plus court chemin entre 2 positions selon divers critères) n'est pas considéré ici comme fonctionnalité élémentaire du système, cette fonctionnalité étant depuis de nombreuses années proposée dans les systèmes commerciaux du marché. Mais il serait aisé d'en développer les fonctions dans l'API.

### *b) Map-matching*

En ce qui concerne le map-matching, fonction qui est la base d'un capteur cartographique, la littérature foisonne de recherches en la matière et pourrait constituer une thèse entière. En effet, le positionnement GPS est très important pour les applications ADAS, or cette technique requiert un ciel dégagé (pour la visibilité des satellites) et ne fournit une position correcte qu'à 10 mètres. Cependant, dans les grandes métropoles, en raison par exemple de la présence des hauts immeubles, la visibilité des satellites est très compromise. (Meng, et al., 2002) rapportent par exemple une disponibilité du système GPS de seulement 30% à Hong Kong. C'est pourquoi, il n'est pas rare de coupler un capteur GPS avec un odomètre en amont d'un algorithme de map-matching (Abuhadrous, 2005) ou de gyromètres (Jabbour, et al., 2008) . Un état de l'art complet du map-matching est proposé dans la thèse (Quddus, 2006). Ces algorithmes vont d'un matching assez simple (position du nœud le plus proche ou du segment le plus proche – matching géométrique), en passant par un matching un peu plus intelligent (calculer l'orientation du véhicule à partir de plusieurs points successifs et matcher le segment le plus proche ayant la bonne orientation – matching topologique) jusqu'à utiliser des concepts plus avancés en se basant sur des modèles de filtrage des données comme le filtre de Kalman, la théorie de Dempster-Shaper, la logique floue,… Cependant, (White, et al., 2000) rapporte que la plupart des algorithmes fonctionnent correctement sur autoroute et que toute l'intention doit se porter sur les problèmes d'intersection.

L'utilisation d'un GPS non « conventionel » au sein du laboratoire de type DGPS (notamment utilisé par l'équipe de numérisation 3D), permet de s'affranchir quelque peu des problèmes de positionnement GPS dont la précision théorique est de 1 mètre (en urbain on note plutôt une précision de 2 à 3 mètres). C'est pourquoi, le map-matching proposé ici dans la CAORTO se base sur un matching géométrique retournant le segment le plus proche. Cette opération est basée sur une fonction de calcul de distance par rapport à tous les segments situés dans la zone courante (dans l'arbre de stockage) de la position actuelle ainsi que ses 8 zones voisines. Basée sur la même fonction, l'API propose de retrouver tous les segments situés dans un certain périmètre d'une position donnée. Bien entendu, il est possible de récupérer la position GPS projetée (corrigée ?) sur le segment map-matché.

L'API calcule en interne, et ce de façon automatique, l'orientation du véhicule (par différentiation des positions GPS successives). Cela permet de proposer la fonctionnalité de récupérer la coordonnée de la prochaine intersection en parcourant le segment courant (map-matchée) dans la bonne direction jusqu'à atteindre un nouveau nœud (et non un shape-point, la différentiation se faisant sur le nombre de segments connectés à ce nœud, i.e. le degré du sommet dans la théorie des graphes).

### c) *Calcul de distances*

La fonctionnalité la plus importante d'un capteur cartographique est de pouvoir calculer la distance entre 2 coordonnées GPS. Le problème réside dans le fait que la terre n'est pas plate mais ronde. Ainsi la distance orthodométrique (aussi appelé distance du grand cercle) est souvent utilisée pour calculer la distance entre 2 points à la surface de la terre (Équation 2). Cette équation est cependant sujette à des erreurs d'arrondis dans le cas de calculs de petites distances. A la place de celle-ci, on lui préfère donc la distance d'Haversine (Équation 3) qui est plus juste en cas de petites distances.

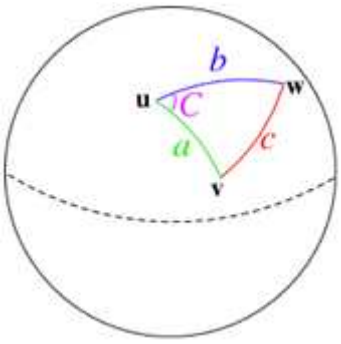



Figure 115 - Sur une sphère, les arrètes d'un triangle sont des arcs - Trigonométrie sphérique

$$d = \text{acos}(\sin(lat1) * \sin(lat2) + \cos(lat1) * \cos(lat2) * \cos(lon2 - lon1)) * R$$

Équation 2 - Distance du grand cercle

$$d = 2 * \text{asin}\left(\sqrt{\sin^2\left(\frac{\text{lat2} - \text{lat1}}{2}\right) + \cos(\text{lat1}) * \cos(\text{lat2}) * \sin^2\left(\frac{\text{lon2} - \text{lon1}}{2}\right)}\right) * \text{R}$$

Équation 3 - Formule d'Haversine

Cependant, l'affirmation qui dit que la « Terre est ronde » est fausse. Elle est en fait plus proche d'un ellipsoïde que d'une sphère, la Terre étant aplatie au niveau des pôles. Ainsi le rayon de la Terre au niveau de l'équateur est de 6378km tandis qu'aux pôles elle n'est que de 6357km (il y a donc 21km de différence). La Terre peut donc être représentée par un ellipsoïde. Un ellipsoïde est défini par deux paramètres : les longueurs du demi petit (r) et du demi grand (R) axe définissant un coefficient d'aplatissement f = (R-r)/R. Il est aussi courant de représenter un ellipsoïde par le facteur d'aplatissement et la longueur du demi grand axe. On aimerait que cet ellipsoïde représentant la Terre colle au niveau des mers. Cependant, la forme de la Terre au niveau des mers (représenté par le géoïde) n'est pas symétrique et ne correspond pas à un ellipsoïde (Figure 116).

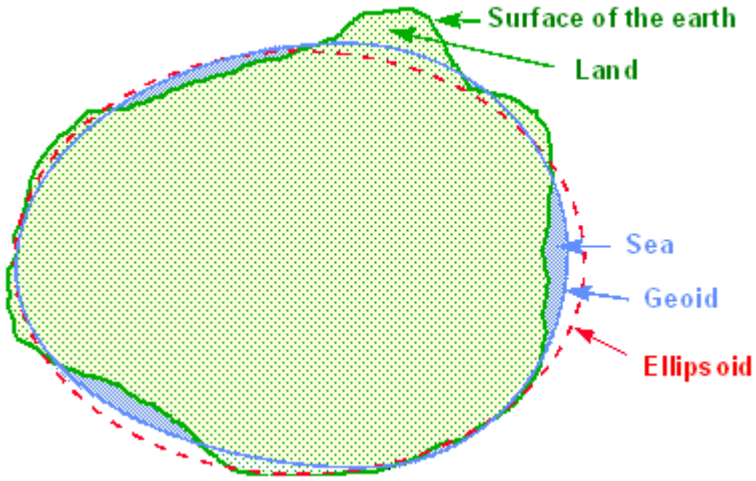



Figure 116- Ellipsoïde et Géoïde

Le géoïde de la Terre étant irrégulier (du fait de la répartition non uniforme de la gravité terrestre), il est trop complexe pour être utilisé directement en cartographie. La Terre est donc représentée en utilisant des ellipsoïdes approximatives. Ainsi, il est possible d'approximer localement le géoïde par un ellipsoïde local (au niveau d'une région / d'un pays) ou par une ellipsoïde global (Figure 117). Le modèle global retenu actuellement pour une représentation totale du globe est l'ellipsoide WGS84 (World Geodesic System 1984) qui a été revu en 2004 et dont la validité court jusqu'en 2010 (les ellispoides devant être corrigés avec le temps, dû à différents facteurs comme l'érosion, la dérive des continents,…). En France est utilisé l'ellipsoïde de Clarke 1880 dont le point fondamental se situe au niveau du Panthéon.

Le calcul des distances dans le modèle ellipsoïdal se fait généralement avec la formule de Thaddeus Vincenty (Vincenty, 1975). Cette formule a été récemment validée par des chercheurs australiens qui ont montré qu'elle était précise à 0,115mm (Thomas, et al., 2005).

Ainsi, l'API de la CAORTO propose le calcul des distances via la formule de Vincenty (cf. Annexe A) en utilisant par défaut l'ellipsoïde WGS84, mais propose aussi l'approximation du calcul par la distance des grands cercles et par la formule d'Haversine.

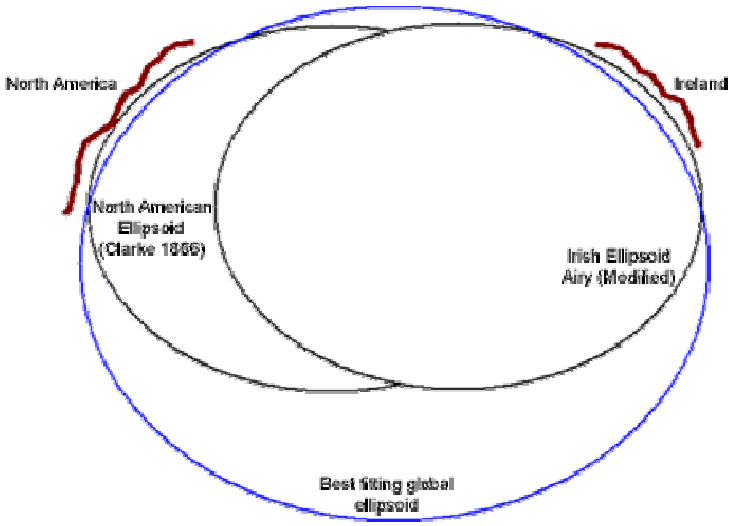



Figure 117 - Ellipsoïdes locales et globale pour représenter la Terre

### d) *Données administratives*

Enfin, il est possible de récupérer les coordonnées des zones administratives gratuitement sur le site Geonames (toujours sous licence Creative Commons). Ces données sont elles aussi intégrées dans la CAORTO et il est donc possible de retrouver les noms du pays, région, ville, arrondissement,… à une position donnée. Cela peut permettre d'établir des règles génériques dans un système ADAS qui ne seraient applicables que dans certaines zones et ou pays. Par exemple, en France la limitation de vitesse sur autoroute est de 110 ou 130km/h, alors qu'en Allemagne il n'y a pas de limitation imposée sur ce type de route.

### e) *Affichage et projections*

D'autres outils sont proposés par l'API qui sont plus d'ordre cosmétique que fonctionnel. Par exemple, il est possible de récupérer une image du graphe routier de la zone en cours et de pouvoir y dessiner facilement des informations directement en coordonnées GPS (et non pixélique), de pouvoir y sur-imprimer des images,… Cela dit, il n'est pas si évident d'afficher une carte correctement : afficher (projeter) une sphère en 3 dimensions sur une surface plane en 2 dimensions.

Il existe plusieurs systèmes de projection pour représenter la Terre, mais une projection ne peut jamais se faire sans qu'il y ait déformation. Pour s'en convaincre, il suffit d'essayer d'aplatir la peau d'une orange. Néanmoins, par le calcul il est possible de définir le type et les paramètres d'une projection dans le but de minimiser certaines déformations. Il est alors possible de :

- soit conserver les surfaces (projections équivalentes)
- soit conserver les angles (projections conformes)
- soit d'opter pour une représentation ne conservant ni les angles ni les surfaces (projections aphylactiques)

Une projection dite équidistante conserve les distances à partir d'un point donné. Dans tous les cas, aucune projection ne peut conserver toutes les distances.

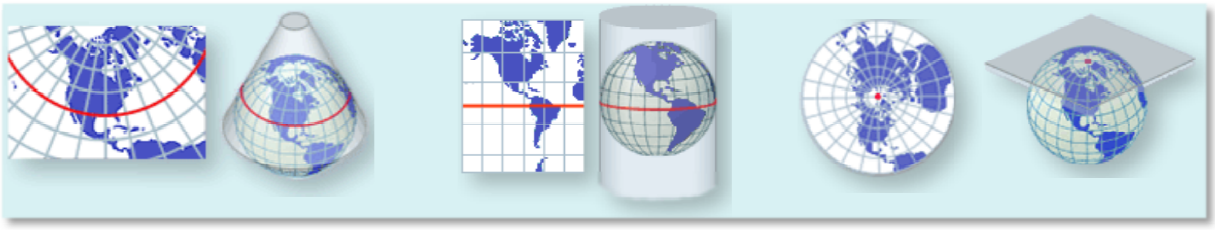


Figure 118 - Projections conformes conique, cylindrique et azimutale

En France, la projection usuellement utilisé est une projection conique (Figure 118) : la projection Lambert. Afin de rendre la projection plus conforme à la réalité (i.e. réduire les déformations), 4 cônes de projection sont utilisés (Figure 119). Une 5$^{ème}$ zone Lambert a été définie, lorsque la cartographie porte sur plusieurs zones et est utilisée pour représenter toute la France : le Lambert 2 étendu.

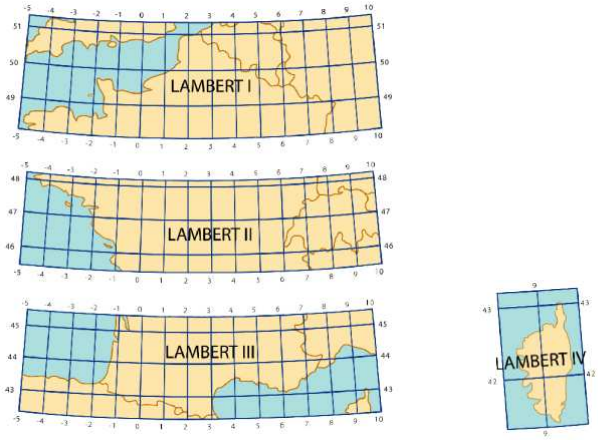



Figure 119 - Projection conique Lambert d'usage en France

La plupart des cartes dans le monde sont affichées avec une projection Mercator (Figure 120) qui est une projection cylindrique directe (Figure 118) facile à calculer (Équation 4). Cette projection

déforme la taille des surfaces qui sont loin de l'équateur. Par exemple, le Groenland est représenté aussi large que l'Afrique alors que l'Afrique est en réalité 14 fois plus grande. Pour pallier ce problème il est possible d'utiliser une projection cylindrique transverse (Figure 118) où il est clair que la déformation sera constante quelque soit l'éloignement à l'équateur. C'est ainsi qu'est défini la projection UTM - Universal Transverse Mercator qui est maintenant plus couramment utilisée. En fait, il n'y a pas 1 mais 60 projections en UTM. En effet, afin que le cylindre soit toujours tangent à la Terre quelque soit la zone cartographiée, la Terre est divisée en 60 zones verticales où en chacune est effectuée une projection cylindrique transverse (Figure 121). Ainsi, trois zones UTM parcours la France.

Le lecteur désirant plus de renseignement sur les systèmes de projection et de représentation de la Terre pourra se référer à (Robinson, et al., 1995).

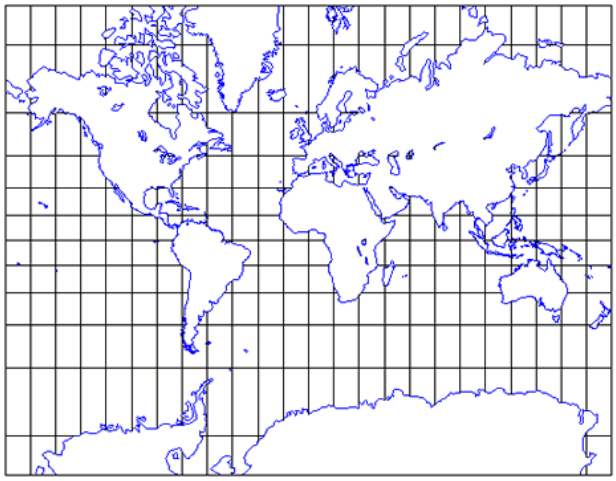


Figure 120 - Projection Mercator – Plus on s'éloigne de l'équateur, plus les aires sont étirées

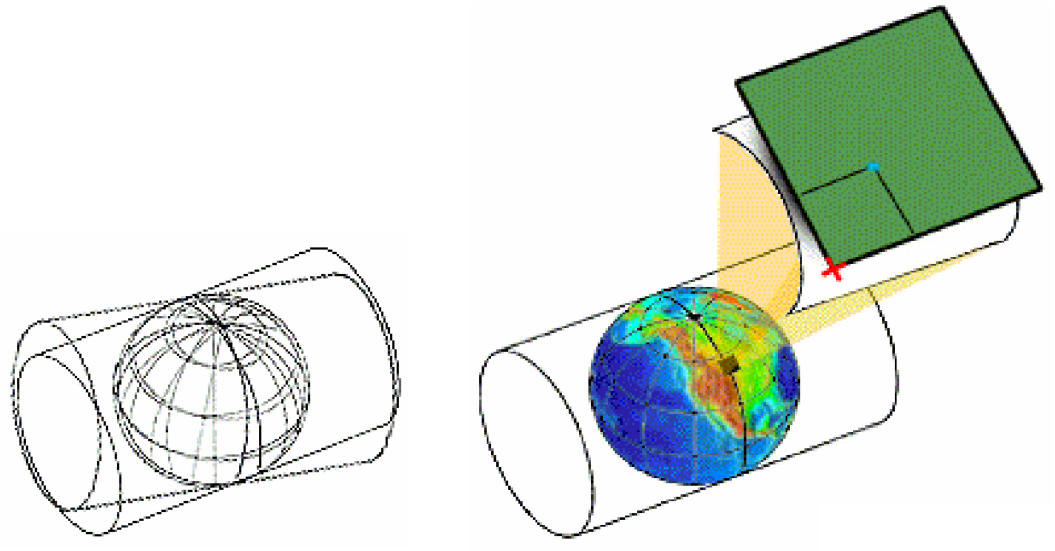


Figure 121 - La projection UTM consiste en plusieurs projections séparées pour chaque méridien tous les 6 degrés

Il est possible au sein de l'éditeur, et ce pour faciliter la création à la main du graphe routier, d'afficher les images satellitaire de Google. Google utilise pour ses cartes le système de projection Mercator (Équation 4). C'est pourquoi, l'API et l'éditeur de la CAORTO utilisent ce système de projection afin de rendre plus « réaliste » (i.e. conforme à ce que nous avons l'habitude de voir en représentation de carte) l'affichage de la cartographie et de ne pas déformer les images satellitaire.

$$x = lon$$
$$y = \ln\left(\tan(\text{lat}) + \frac{1}{\cos(lat)}\right)$$

Équation 4 - Projection Mercator

## 2.5. Intégration et résultats de travaux futurs

Afin de valider l'implémentation des algorithmes de calcul des distances de la CAORTO, ceux-ci ont été comparés avec ceux développés par des systèmes commerciaux. Ainsi, deux points GPS ont été relevés sur Google Maps, la distance entre ces deux points a été calculée avec Google Maps et avec FizzyCalc ; puis avec l'algorithme de Vincenty en utilisant l'ellipsoïde WGS84, le formule d'Haversine et le calcul des grands cercles de la CAORTO. L'API de la CAORTO permet aussi de convertir les coordonnées GPS en coordonnées dans le système de projection Lambert 2 étendu (utilisé en France) qui fournit des coordonnées dans un repère cartésien en mètre. La conversion des deux points GPS référence ici en Lambert permet ainsi de calculer directement leur distance. Le Tableau 14 montre que tous ces calculs sont conformes à 30 cm près, ce qui gage d'une bonne implémentation et de bons résultats pour la CAORTO.

| | Latitude | Longitude |
|---|---|---|
| **Point A** | 48.845579 | 2.339834 |
| **Point B** | 48.843855 | 2.338878 |

| Logiciel | Distance (m) |
|---|---|
| **Google Maps** | 204.29 |
| **FizzyCalc (Vincenty WGS84)** | 204.15 |
| **FizzyCalc (Haversine)** | 203.92 |
| **FizzyCalc (Grand Cercle)** | 203.92 |

| Formule (CAORTO) | Distance (m) |
|---|---|
| **Lambert2e** | 204.26 |
| **Vincenty WGS84** | 204.15 |
| **Haversine** | 204.06 |
| **Grand Cercle** | 204.06 |

Tableau 14 - Comparaison du calcul des distances des Points A et B avec la CAORTO et d'autres logiciels

En ce qui concerne la procédure de map-matching, comme prévu du fait de l'utilisation d'un DGPS assez précis (d'une précision théorique d'1 mètre), le calcul du segment le plus proche au segment en cours utilisé dans la CAORTO, ne pose de problème qu'au niveau des intersections. Ainsi sur un carrefour par exemple, la notion de segment le plus proche est trop ambigüe est peut très bien se retrouver être n'importe lequel des segments joignant le carrefour. Par exemple, Figure 122, si le GPS se « trompe » et retourne la position rouge, alors le segment le plus proche sera le A, si c'est la position bleue qui est fournie, alors le segment le plus proche calculé sera le C,… n'importe quelle route à ce carrefour peut être le segment le plus proche quand le véhicule s'y trouve au centre. Avec un GPS classique (d'une imprécision de 10 mètres), ce phénomène serait encore plus grand et l'erreur de map-matching, qui est ici très courte, du fait qu'un véhicule ne reste généralement pas longtemps sur un carrefour, s'en ressentirait fortement. Il serait bon, dans des travaux futurs d'incorporer un processus de map-matching beaucoup plus robuste fusionnant des informations proprioceptives (comme l'odométrie, l'angle au volant,…) afin de savoir si le véhicule tourne et sélectionner le bon segment.

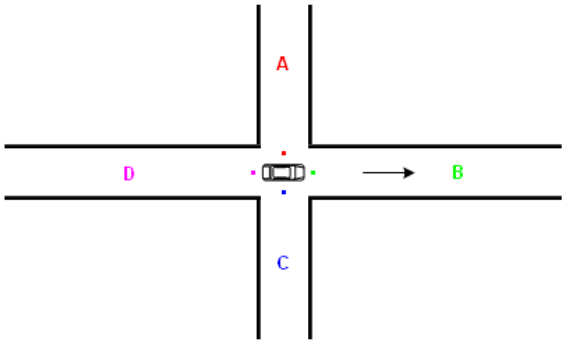



Figure 122 - Imprécision du GPS et choix du segment le plus proche (map-matching)

L'intégration de l'API au sein de notre logiciel de prototypage ($^{rt}$Maps) est aisée et ne pose guère de problème. Ainsi, comme le montre la Figure 123, l'utilisation des coordonnées GPS permet bien d'afficher la cartographie de l'endroit où le véhicule se trouve. Tout d'abord les noms de la rue, de la ville, du département, de la région et du pays sont corrects. L'information administrative semble donc être correctement sauvegardée et retrouvée. Les fonctionnalités de dessins offertes par la CAORTO permettent ensuite de visualiser et de valider ses diverses fonctionnalités. Ainsi l'affichage de la limitation de vitesse en cours attribuée au segment courant (sous forme d'une image en sur-impression du graphe routier), l'affichage en rouge du segment sélectionné par le map-matching, de la position GPS en verte, de la position rétroprojetée en bleue et de la position du prochain carrefour en violet, permet de valider les calculs mis en jeux par ces diverses fonctionnalités. L'utilisation d'une caméra valide les résultats de map-matching et de projection de la position GPS en comparant la position visuelle (par rapport aux batiments) de la position cartographique. Enfin, cette validation « visuelle » a eu lieu sur plusieurs séquences dans Paris, et n'a pas révélé de problème particulier (hormis le map-matching aux carrefours). Elle a en outre prouvé le bon comportement en temps réel de la CAORTO (pour la récupération des données routières dans la base et l'affichage de ces informations).

En ce qui concerne l'éditeur de donnée, c'est celui-ci qui a permis d'importer (et de traduire) les données d'OpenStreetMap et de GeoNames. Au vu des précédents résultats, il est clair que cette importation fonctionne et sauvegarde correctement dans le bon format les données dans la base de la CAORTO. L'affichage et l'édition des données dans une même interface (Figure 124) permet aussi de valider ces deux fonctionnalités : en effet, la sélection d'un segment permet d'afficher ses attributs, la modification de l'un deux, comme le sens de circulation entraine la sauvegarde de la nouvelle donnée et un changement d'affichage (après relecture des données). Enfin, l'affichage en fond des images satellitaires Google permet de valider l'affichage en projection Mercator du graphe routier ainsi que les coordonnées GPS (i.e. la position) des routes (Figure 125).

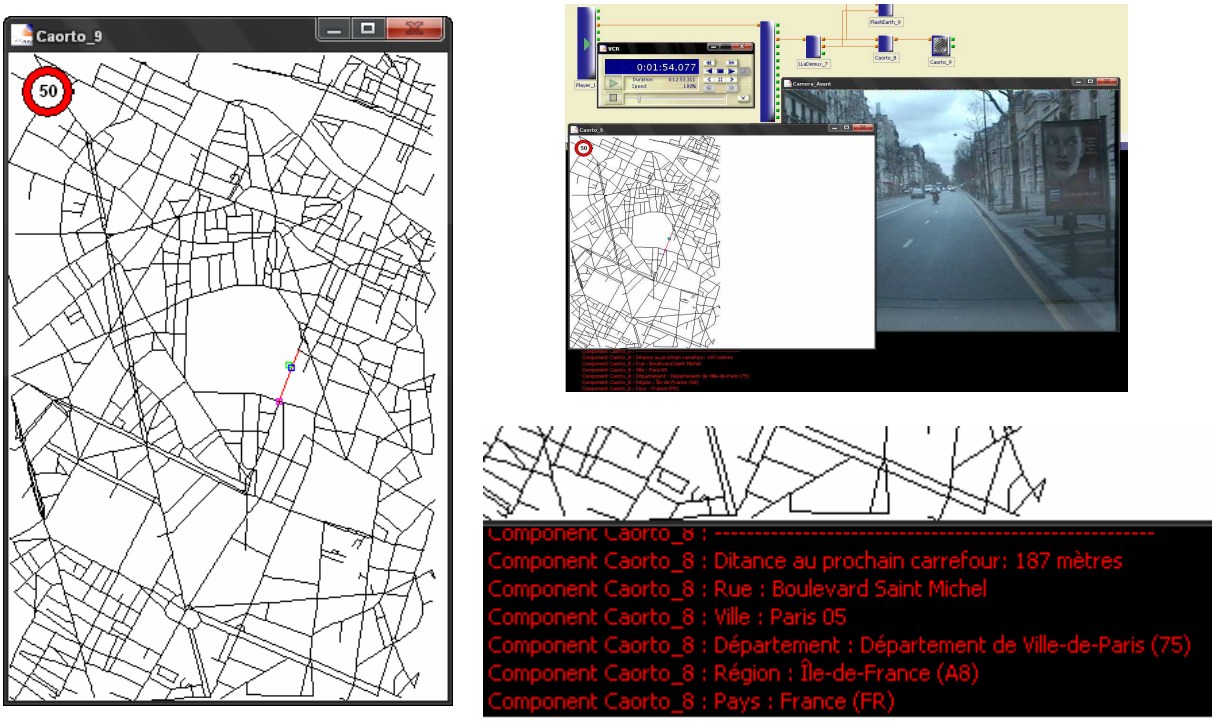



**Figure 123 - Intégration de la CAORTO dans rtMaps (à gauche le graphe routier avec diverses informations projetées, en haut à droite une vue d'ensemble de l'intégration, en bas à droite un zoom sur l'affichage des informations administratives)**

L'une des équipes du laboratoire a mis au point une technique de numérisation et modélisation 3D d'environnements extérieurs, utilisant un système mobile appelé LARA-3D. Il s'agit d'une voiture équipée d'un système de localisation géographique précis (GPS, Centrale Inertielle), d'un télémètre laser fixé à l'arrière du véhicule, et de caméras. Ce dispositif, utilisé en environnement urbain et routier, permet de recueillir des nuages de points 3D décrivant avec une bonne précision les éléments présents le long des trajets effectués (routes, ronds-points, façades, arbres, voitures...), et

après traitement d'avoir des modèles par facettes des scènes numérisées, ainsi que des couleurs sur les points et des textures photo-réalistes sur les facettes (Brun, 2007).

Une des idées émises étaient de pouvoir sauvegarder ces informations 3D. Or, extraire les informations (largeur de la route, rayon de courbure vertical,…) d'un modèle 3D en vue d'être intégrées dans une base cartographique et surtout utilisées n'est pas chose aisée et n'est pas encore d'actualité bien que des travaux sont en cours sur ces problématiques. Cependant, les modèles 3D issus de ces scans sont géo-référencés. C'est pourquoi, dans un premier temps, ces modèles bruts ont été intégrés afin de pouvoir être affichés les uns à coté des autres sur une carte, telle la CAORTO (Figure 126). Cet affichage ne constitue pour l'instant qu'une « démonstration technologique » et ne prend pas en compte par exemple l'élévation du terrain. Nous n'avons pas encore réfléchi sur la façon de stocker intelligemment les modèles 3D (s'il faut par exemple les segmenter par zone et les stocker de façon hiérarchique ou non, comme pour le graphe routier,…) ?

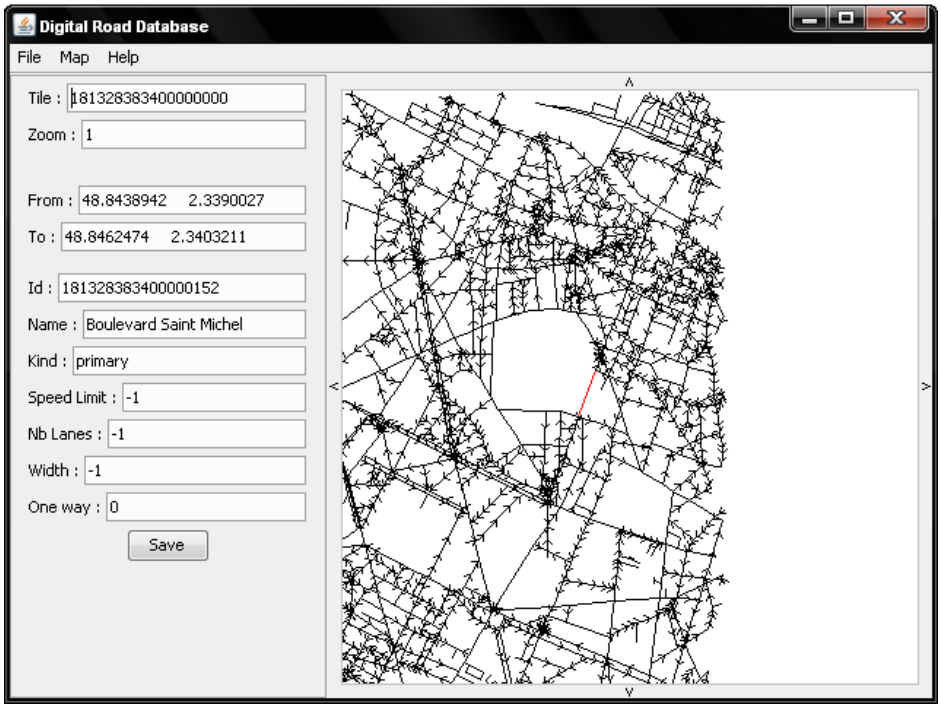



Figure 124 - L'éditeur de la CAORTO permet l'affichage et l'édition des données

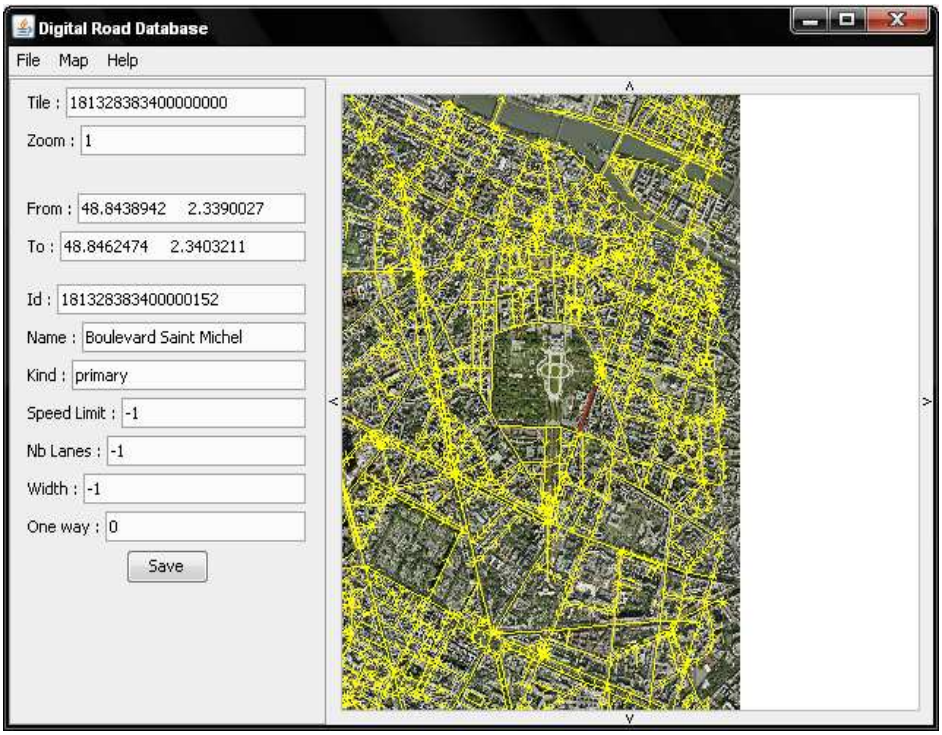



Figure 125 - L'éditeur de la CAORTO avec affichage des images satellitaire Google

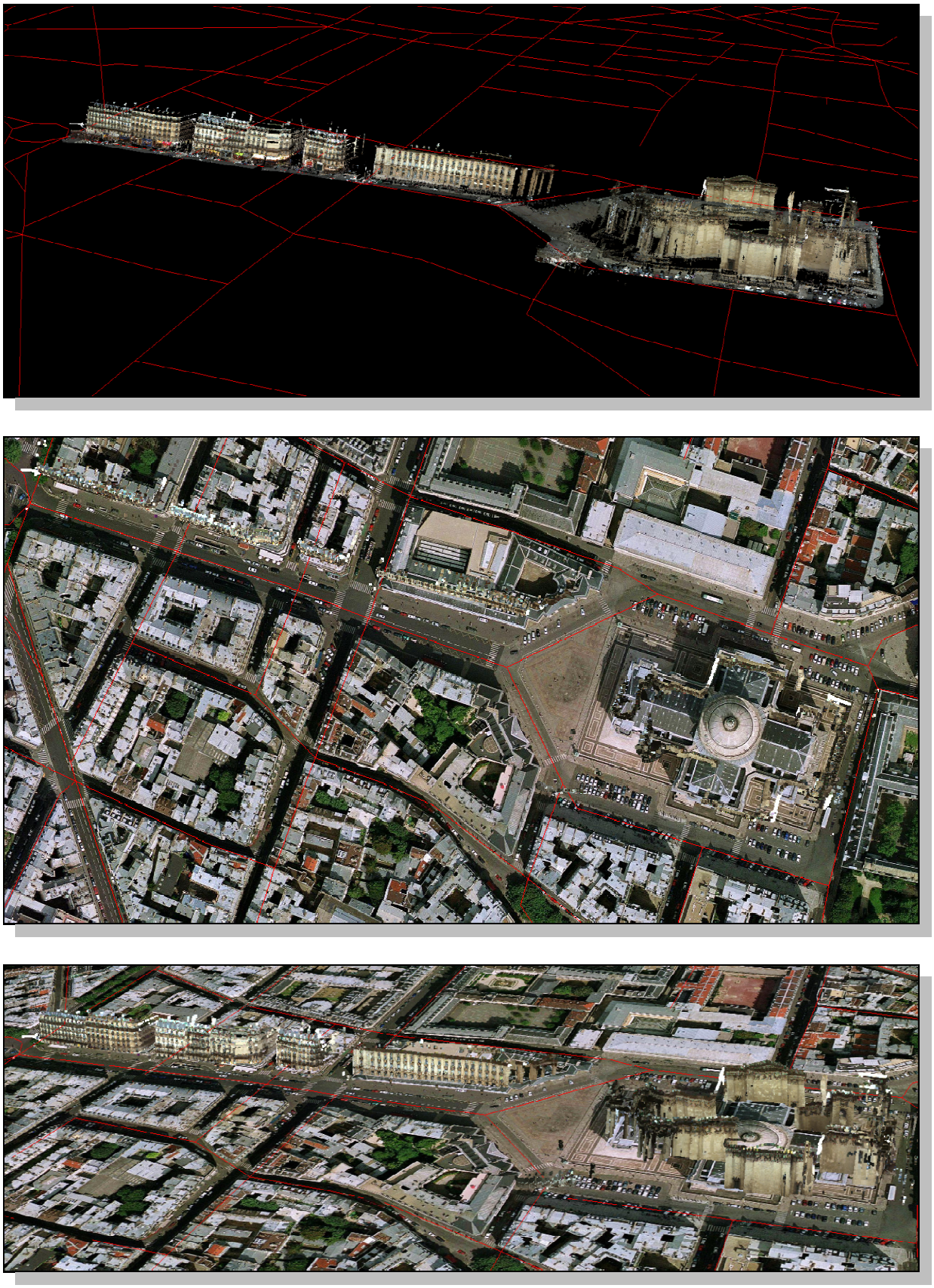


Figure 126 - Intégration et affichage 3D de 3 modèles 3D juxtaposés sur la carte (avec ou sans affichage des données satellitaires Google)

Enfin la CAORTO est déjà utilisée dans d'autres projets du laboratoire comme :

- SLA en collaboration avec Valeo, qui vise à effectuer une détection intelligente de la limitation de vitesse
- SpeedCam, pour la création d'un ACC (Advanced Cruise Control) légal et intelligent
- DIVAS, pour le stockage de données de l'infrastructure (ex : l'adhérence des routes,…)

La CAORTO sera aussi sans doute utilisée dans les travaux de thèse futurs du laboratoire, notamment les travaux ADAS où il sera possible par exemple de déclencher la détection des feux tricolores aux approchent des carrefours et d'orienter les zones de recherche dans l'image.

Enfin, il n'est pas exclu d'utiliser nos divers algorithmes de détection visuelle (ex : détection de panneaux de limitation de vitesse, des feux tricolores, ….) afin d'enrichir la CAORTO en y enregistrant de façon automatique et en temps réel ses « attributs » de route.

L'utilisation de la CAORTO peut être aisément exploitée afin de connaître la limitation en cours (Figure 123) au cas où le système visuel n'ait pas encore détecté et reconnu de panneau de limitation de vitesse. Cependant, dans le cas schématisé Figure 127, la position GPS qui souffre d'une certaine imprécision (10 mètre en cas normal, cercle bleu sur la figure), fait qu'il n'est pas possible de savoir sur quelle voie le véhicule se trouve (sur l'autoroute ou sur la voie de sortie). Le capteur cartographique seul ne peut donc pas déterminer la limitation de vitesse même si celle-ci est renseignée comme attribut des segments. Or, le système de reconnaissance visuel peut très bien détecter les 2 panneaux de vitesse (70 et 110 de la voie de sortie et de l'autoroute) qui sont à proximité et être lui aussi dans l'incapacité de prendre une décision quant à la limitation de vitesse en cours ! L'utilisation de la théorie de Dempster Shafer (comme il est fait dans l'état de l'art) ne résoudrait rien. En effet, les 2 systèmes renvoyant chacun 70 et 110 en guise d'éléments focaux (de détection), il serait impossible de fusionner ces résultats pour déterminer la limitation qui est la bonne sans renseignement extérieur (la voiture pouvant très bien être positionnée sur la voie de sortie).

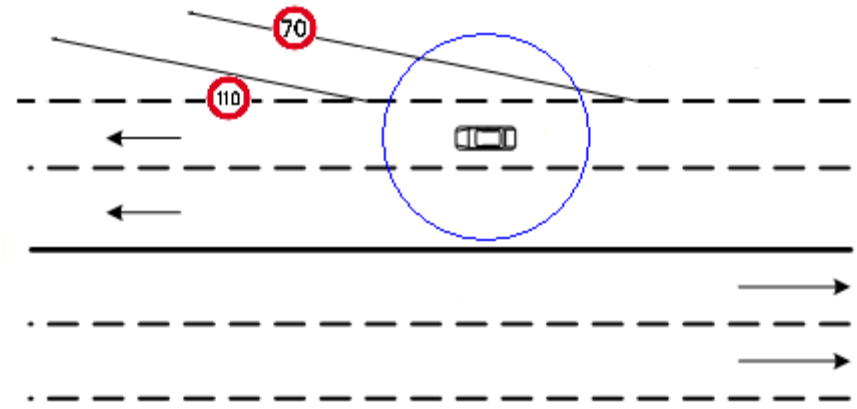



**Figure 127 - Cas où l'utilisation d'un capteur cartographique seul ne suffit pas à déterminer la limitation de vitesse (ellipse d'incertitude de positionnement GPS en bleu)**

Afin de résoudre ce problème, on se propose d'utiliser l'information issue de la détection des lignes de marquage au sol décrite dans le chapitre suivant.

# Détection des lignes de marquage au sol

## 3.1. Motivations

Les situations de collisions et de sorties de route sont celles qui précèdent immédiatement la plus grande partie des accidents. En Allemagne par exemple, les sorties de voie sont la cause d'un des six types d'accident les plus sérieux et plus d'un tiers des morts sur la route est dû à ce type d'accidents (Daimler).

Ainsi, la détection des lignes de marquage au sol est l'un des composants fondamentaux dans un système d'aide à la conduite automobile. Détecter ces lignes blanches (ou jaunes) peut permettre entre autre de prévenir automatiquement le conducteur en cas de sortie de voie involontaire, quand le conducteur n'active par ses clignotants par exemple, de contrôler l'attention du conducteur par rapport à sa ligne de conduite et de le prévenir en cas d'éventuelle fatigue, de faire du contrôle automatique du véhicule pour éviter la sortie de voie (Figure 128).

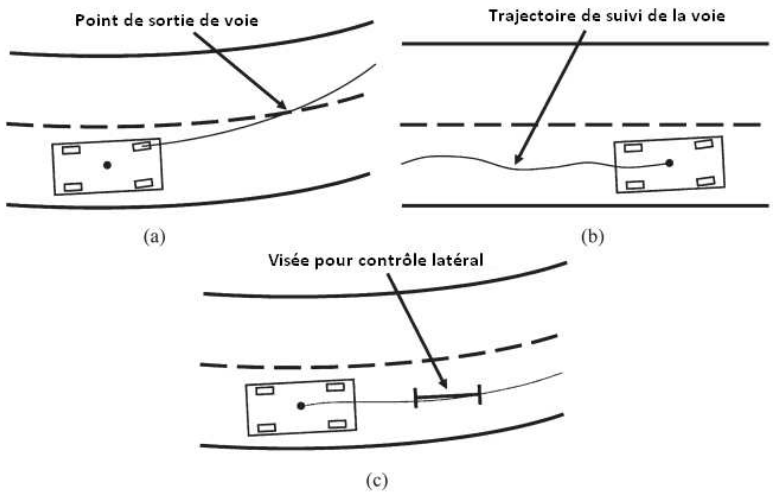



**Figure 128 – Illustrations de systèmes requérant une détection des lignes de marquage au sol : (a) Prévention de sortie de voie, (b) Surveillance de l'attention du conducteur, (c) Contrôle du véhicule**

Le problème de prévention de sortie de voie (Lane Departure System – LDS) n'est pas nouveau et intéresse depuis une dizaine d'années les constructeurs automobiles qui le commercialisent déjà en option sur certaines voitures haut de gamme (Wikipedia). Ainsi le premier à avoir vendu un tel système sur une de ses voitures fut **Nissan** en 2001 avec la Cima. **Toyota** l'a suivi en 2002 avec ses voitures Cardina et Alphard vendus au Japon. Puis **Honda** en 2003 avec l'Inspire. Tous ces systèmes

intègrent une caméra derrière le rétroviseur central pour effectuer la détection des lignes de marquage au sol. En 2004, **Citroën** a développé ce système en utilisant des diodes et capteurs infrarouge placés sous le bouclier avant du véhicule et en équipe depuis 2005 ses véhicules C3, C4 et maintenant C6. En 2006, **Lexus** a commercialisé la LS460 qui utilise une paire de caméras stéréoscopiques pour la détection des lignes. Depuis 2008, **General Motors** propose ce système sur ses véhicules Cadillac STS et DTS et sur ses Buick Lucerne ; la même année **Volvo** le propose sur ses modèles S80 et V70 ; ainsi que **BMW** avec ses Série 5 et Série 6. Le système utilisé par ces 3 derniers constructeurs est développé par la société Mobileye. Enfin, depuis 2009, **Merces-Benz** propose aussi ce système sur sa nouvelle classe E. Il est donc clair que tous les constructeurs portent un grand intérêt à ce problème, gage d'un enjeu majeur des systèmes d'aide à la conduite automobile.

La détection des lignes de marquage au sol, si elle estime la courbure de la route, peut permettre de réaliser un grand nombre d'applications ADAS de plus haut niveau. Il est par exemple possible d'alerter le conducteur d'un éventuel risque si sa vitesse est trop importante face à un prochain virage. Couplée à un système de détection de véhicule, celle-ci permet de prévenir des collisions entre véhicules empruntant la même voie. Par exemple, dans la Figure 129, un système de détection de véhicule dirait que la voiture n'est pas en « face » de nous et qu'il n'y aurait pas de risque. En estimant la courbure de la route, le système saurait que ce véhicule est en fait dans notre voie, et il pourrait prévenir d'un risque éventuel en cas d'accélération de notre véhicule ou de freinage brutal du véhicule en amont.


**Figure 129 - Cas où il est nécessaire de coupler un système de détection des lignes de marquage au sol à un système de détection de véhicule pour créer un système, de plus haut niveau, d'évitement de collision**

Il peut aussi être déduit d'une détection des lignes de marquage au sol le nombre de voies que comporte la route et surtout sur quelle voie le véhicule roule. Cependant cela n'est possible que lorsque la route est assez dégagée et donc que les marquages sont apparents (non occultés par d'autres véhicules). Un tel système sur le cas de la Figure 129 pourrait aisément déduire que cette route comporte au moins 3 voies et que le véhicule roule sur la voie centrale. Couplé à un système de détection de sortie de voie, il serait envisageable de détecter un changement de voie et de savoir si le véhicule emprunte, par exemple, une voie de sortie. Cela serait, dans le cas du problème de la

prise en compte des détections des panneaux de limitation de vitesse, un atout considérable quand à la prise de décision de notre système.

## 3.2. Etat de l'art

Les travaux sur la détection des lignes de marquage au sol en utilisant une caméra remontent à plusieurs dizaines d'années (Dickmanns, et al., 1992) et continue de nos jours à être approfondis (Ślot, et al., 2009). Cependant la majorité de ces travaux peut être décomposée en trois ou quatre étapes comme le relève (McCall, et al., 2006), qui comporte un état de l'art complet sur ce sujet :

- La définition d'un modèle de route, qui est généralement basé sur une représentation polynomiale du premier ou du second ordre (des clothoïdes) ;
- L'extraction des caractéristiques des lignes de marquage au sol, dont la majorité des travaux utilisent l'extraction des contours des lignes ;
- L'analyse des ses caractéristiques, qui se fait généralement en utilisant l'algorithme de Hough ;
- Et optionnellement d'opérations de post-traitement.

A ces méthodes, il faut ajouter d'autres approches basées sur la segmentation de la route (segmentation de région).

Plus concrètement la détection des lignes de marquage au sol peut se faire, comme pour la détection des panneaux de limitation de vitesse, soit par détection des contours de forme (recherche de ligne) soit par segmentation couleur (les lignes étant blanches).

L'extraction des caractéristiques des lignes de marquage au sol par extraction par contour de forme (donc sur des images en niveau de gris) se fait en utilisant des algorithmes bien connus de traitement d'images. Ainsi, est généralement utilisé le filtre de Canny - Deriche pour obtenir une image des contours de forme (dont les contours ne sont représentés que par 1 pixel) très peu sujette au bruit comme l'utilisent (Macek, et al., 2004), (Nedevschi, et al., 2004), (Wang, et al., 2004), (Tian, et al., 2006) et (Hetrick, et al., 2007). D'autres types de filtres du domaine du traitement d'image sont aussi utilisés, par exemple un filtre Gaussien dans (Aly, 2008) ou le filtre LoG (Laplacian of Gaussian) dans (Macek, et al., 2004) et (Ślot, et al., 2009), qui est plus sujet au bruit que le filtre de Canny (Figure 130), mais beaucoup plus rapide à appliquer sur une image. Les techniques issues du domaine de la morphologie mathématique sont utilisées dans (Tsai, et al., 2008) où par différenciation des images résultantes d'ouvertures et de fermetures ils arrivent à trouver, après seuillage, les contours des lignes de marquages au sol. Enfin, certains travaux, sur la base de l'hypothèse qu'une ligne de marquage au sol constitue un changement de luminosité horizontale (et donc de contraste) sur le sol de type foncée / claire / foncée, recherche soit directement ce type de changement dans l'image (Southall, et al., 2001) et (Liu, et al., 2008) ou par une convolution de l'image en utilisant un noyau unidimensionnel comme dans le système ARCADE de (Kluge, 1994).

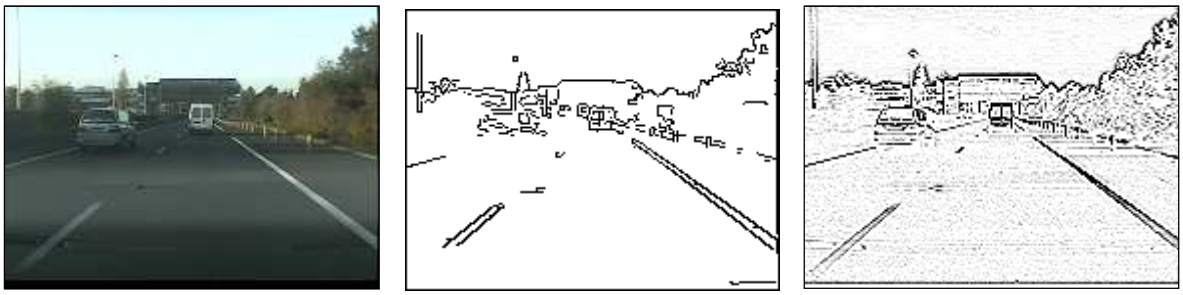


Figure 130 - Exemple de résultat d'extraction de contour sur l'image témoin (à gauche) via filtre de Canny (au centre) et le filtre Laplacian of Gaussian (à droite)

L'extraction via segmentation par couleur fait l'hypothèse que les lignes sont blanches (ou jaunes) sur un sol plutôt foncé. Ainsi dans leurs travaux, (Macek, et al., 2004) comparent chaque composante couleur RGB de chaque pixel par rapport à la valeur moyenne des composantes de la route afin de classifier les pixels en type « route » ou « ligne ». Si la différence de ces deux composantes est supérieure au seuil défini par l'écart-type de la couleur de la route, alors le pixel est considéré comme ne faisant pas partie de la route. Cependant, comme stipulé au chapitre 1 sur la détection des panneaux de limitation de vitesse, une segmentation par couleur souffre forcément des variations de condition lumineuse. Pour pallier ce problème, (Macek, et al., 2004) recalcule constamment la couleur moyenne et la variance des pixels de la route (sur une zone prédéfinie dans l'image). Ce type de calcul de seuil dynamique pour répondre aux changements de conditions lumineuses est aussi effectué par (Southall, et al., 2001). D'autres effectuent de la segmentation couleur en se basant sur un autre espace de couleur, moins sensible aux variations lumineuses, comme (Ślot, et al., 2009) qui effectuent un seuillage sur la composante Hue de la représentation HSV. Enfin, certains définissent leur propre espace de couleur où la route et les lignes de marquages au sol auraient toujours les mêmes valeurs quel que soient les variations lumineuses. Ainsi (Cheng, et al., 2006) définisse un espace de couleur $D_1D_2D_3$ résultant de la différentiation des canaux du modèle RGB de la caméra (D1 = rouge - bleu / D2 = bleu - vert / D3 = rouge - vert), la segmentation se faisant ensuite par un seuillage sur ce modèle.

Certains travaux fusionnent les résultats de différents types de segmentation afin d'obtenir un résultat global amélioré. Par exemple (Macek, et al., 2004) utilisent 3 segmentations (Canny, LoG et couleur) et attribue à chaque segmentation un niveau de confiance se basant sur le résultat des détections précédentes ; la segmentation ayant le plus grand niveau de confiance est retenue. (Ślot, et al., 2009) superposent simplement les images de contours obtenues par leurs différentes segmentations (couleur et contour de forme).

Enfin, certains travaux s'essayent à utiliser les techniques issues de l'intelligence artificielle afin de détecter les lignes de marquage au sol dans les images. Par exemple, (Gopalan, et al., 2009) utilisent la technique du boosting (via l'algorithme d'Adaboost) avec des sous-classificateurs faibles de type Haar. Une comparaison de différents algorithmes (SVM, Réseaux de neurones, Bayes) appliqués à la segmentation des lignes de marquage au sol, a été proposée par (Kim, 2008). Il a montré que ces

algorithmes présentent de meilleurs résultats que l'approche couramment utilisé pour la détection des lignes de marquage au sol (i.e. la segmentation par contraste), les SVM donnant les meilleures détections.

Dans tous les cas, l'image résultante de la segmentation doit être traitée afin d'y reconnaître et de modéliser les lignes de marquage au sol. Il est possible de modéliser ces lignes par des droites (en faisant donc l'hypothèse que la route est plate ou localement plate), par des courbes ou par des modèles plus complexes comme une modélisation en 3 dimensions de la route. L'approche la plus simple étant de modéliser les lignes de marquage au sol par une ligne droite, il suffit d'appliquer le filtre de Hough qui est capable de trouver les lignes dans une image même bruitée comme le font (Macek, et al., 2004), (Hetrick, et al., 2007) et (Aly, 2008). D'une manière similaire, (Wang, et al., 2004) et (Tian, et al., 2006) utilisent le filtre de Hough, mais au lieu de l'appliquer sur toute l'image issue de la segmentation, ils divisent cette image en 5 zones horizontales puis appliquent le filtre sur chacune de ces 5 zones. Ceci leur permet de détecter la courbure de la route en retenant les points trouvés par Hough comme appartenant aux lignes de marquage au sol, et en utilisant un algorithme de contour actif (Snake) sur ces points afin de trouver les paramètres d'une courbe de type B-Spline. Dans (Aly, 2008), ils utilisent une détection par Hough, comme zone de recherche pour l'algorithme RANSAC (RANdom SAmple Consensus) afin de trouver les Splines (courbes) qui collent au mieux aux points segmentés. RANSAC est une méthode itérative permettant d'estimer les paramètres d'un modèle mathématique (une ligne, une courbe,…) à partir d'un ensemble de données observées qui contiennent des données aberrantes (ici les point segmentés de l'image originale). Enfin certains travaux essayent de modéliser la route en 3 dimensions (i.e. en calculant les courbures horizontale et verticale). L'utilisation d'une paire de caméra permet à (Nedevschi, et al., 2004) de calculer la hauteur des points détectés (appartement aux lignes de marquage au sol) dans l'image. L'utilisation de l'algorithme de Hough, sur plusieurs zones, leur permet de trouver dans ce nuage de points, la pente de la route (Figure 131).

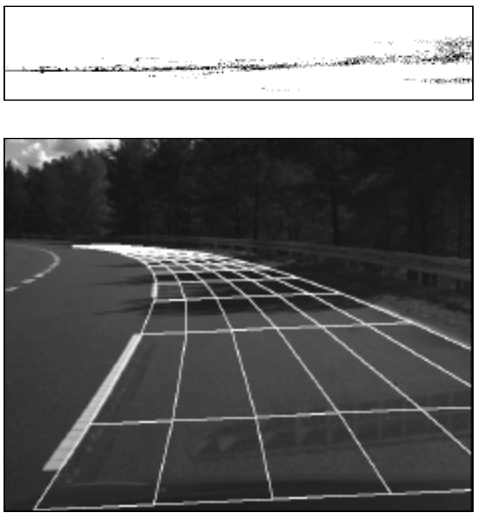


**Figure 131 - Détection et modélisation en 3 dimensions de la route (Nedevschi, et al., 2004) – En haut, hauteur des points des lignes de marquage au sol, en bas résultat d'une détection.**

Quelque soit la nature de la modélisation, de fausses détections sont à prévoir. Ainsi tous les travaux filtrent leurs reconnaissances afin de supprimer ces fausses détections. De par la nature très contrainte des lignes de marquage au sol, il est facile de supprimer ces fausses détections. En effet, tout comme l'hypothèse émises dans l'ensemble des travaux, ces lignes convergent à l'horizon et ne peuvent être représentées que d'une certaine manière dans l'image (par exemple, il ne peut y avoir une ligne de marquage au sol détecté verticalement sur un des côtés de l'image). De plus, la largeur des voies étant assez normalisée, certains travaux valident les détections par paire de ligne (Bertozzi, et al., 1997).

La dernière étape, qui n'est pas toujours effectuée dans les travaux de la littérature, est celle du suivi temporel. L'intérêt de cette étape est multiple : elle permet de définir une zone de recherche des lignes de marquage au sol dans l'image courante par rapport aux détections précédentes afin de réduire le temps de calcul ; elle permet aussi de lisser les résultats et pallier les non détections à certains moments (en cas d'occultation d'une ligne de marquage au sol, quand un véhicule change de voie par exemple). Ainsi, (LIM, et al., 2009) définissent une simple zone de recherche d'une largeur fixe autour des lignes (droites) détectés à l'image précédente. (Nedevschi, et al., 2004) quant à eux définissent une zone de recherche un peu plus complexe. Tout d'abord ils prédisent la position 3d de la ligne de marquage au sol via un filtre de Kalman, ce qui leur permet de pallier aux problèmes éventuels de non détection dans une image. Ensuite cette prédiction est couplée aux informations fournies par la centrale inertielle (qui détermine les angles de tangage et de roulis de la voiture) afin de définir des zones de recherche « 3d » des lignes projetées sur la nouvelle image (Figure 132). La prédiction par filtre de Kalman (simple ou étendu) est couramment utilisée dans les divers algorithmes des systèmes ADAS. Ainsi, (McCall, et al., 2006) l'utilisent aussi afin de prédire la position des lignes de marquage au sol. L'utilisation de la détection par contour actif permet à (Wang, et al., 2004) d'utiliser les paramètres des courbes précédentes, qui doivent être sensiblement équivalentes à ceux des lignes courantes, comme initialisation à leur algorithme de contour actif. Enfin, l'utilisation des filtres particulaires via l'algorithme Condensation (Conditional Density Propagation) est aussi envisagé dans (Southall, et al., 2001) et (Liu, et al., 2008). En effet cet algorithme a initialement été développé pour répondre au problème du suivi temporel de courbes dans des images (Isard, et al., 1998) et est plus robuste au bruit que le filtre de Kalman (Southall, et al., 2001).

**Figure 132 - Résultat de la prédiction de la position des lignes de marquage au sol ainsi que les zones de recherche**

Enfin d'autres méthodes plus hétéroclites ont été proposées dans la littérature. Par exemple, (Bertozzi, et al., 2002) utilisent l'algorithme évolutionnaire des Fourmis pour reconnaître les lignes de marquage au sol dans l'image des contours segmentés. Le système RALPH (Pomerleau, 1995) quant à lui, propose une approche assez originale : l'utilisation d'un processus du type « prédiction / vérification ». Dans un premier temps l'image en perspective inversée (inverse perspective mapping – IPM) de la route (Figure 133) est transformée en 5 images possibles selon 5 hypothèses statiques de courbures (forte courbure gauche, faible courbure gauche, pas de courbure, faible courbure droite, forte courbure droite). La bonne hypothèse de courbure (l'étape de vérification) est représentée par une image de ligne de marquage au sol rectiligne qui est facilement détectable parmi ces 5 images via une analyse d'histogramme. L'utilisation d'image IPM dans les travaux de détection des lignes de marquage au sol est souvent utilisés afin de simplifier la détection (Bertozzi, et al., 2002), (Bertozzi, et al., 1997), (Pomerleau, 1995), (Cheng, et al., 2008) et (Aly, 2008), mais elle implique de connaître le tangage de la voiture.

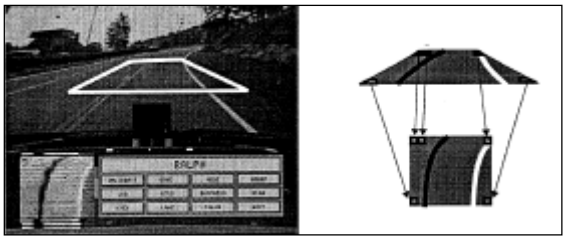


Figure 133 – Image originale avec sa région analysée à gauche, Image en perspective inversée à droite

## 3.3. Système développé

L'application visée ici par le système de détection de lignes de marquage au sol est de pouvoir détecter les changements de voie du véhicule et optionnellement pouvoir compter le nombre de voies afin de désambigüiser certaines situations complexes pour la reconnaissance des panneaux de limitation de vitesse.

L'algorithme développé suit les 4 grandes étapes définies dans la littérature :

- Le modèle de route défini est celui d'une route plane sans courbe (i.e. une ligne dans l'image), le modèle le plus simple qu'on puisse utiliser, mais qui répond à notre problématique ;
- L'extraction des caractéristiques des lignes de marquage au sol, se fait, comme la majorité des travaux de la littérature, via une extraction des contours des lignes ;
- L'analyse des caractéristiques extraites, se fait là aussi comme la plus grande partie des travaux, en utilisant l'algorithme de Hough ;

- Une dernière étape de suivi spatio-temporel des lignes afin de gérer les cas d'occultation des lignes.

### 3.3.1. Extraction de contours

#### *a) Le filtre*

Cette première étape se base sur l'étude effectuée par un ancien doctorant du laboratoire, Pierre Coulombeau, réalisée en 2002.

Les lignes de marquage au sol montrent toujours un contraste (i.e. un passage brut d'une zone foncée à une zone claire ou inversement) plus ou moins important avec la couleur de la route. Dans la littérature, plusieurs filtres linéaires ont été utilisés afin de détecter ces changements de contraste, comme les filtres [-1 … -1 0 1 … 1] ou [-1 … -1 1 … 1] (Pomerleau, 1995). Les efficacités respectives de ces deux filtres sont assez proches, ainsi le second filtre a été retenu arbitrairement comme base d'étude au module d'extraction de contours mis en place ici.

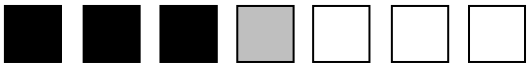
L'application du filtre [-1 … -1 1 … 1], de longueur 2L, est définie par l'Équation 5, où NegSum est la somme des valeurs des L pixels à gauche de la position courante dans l'image, et PosSum celles des L pixels de droite.

$$v = 128 + \left\lfloor \frac{PosSum - NegSum}{2 * L} \right\rfloor$$

Équation 5 - Application du filtre linéaire [-1 ... -1 1 ... 1]

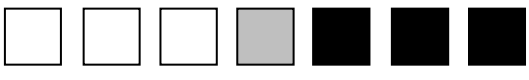
Appliquons ce filtre de longueur 7 à deux cas extrêmes :

- Dans le cas d'une hausse brutale du niveau de gris, le pixel dans la nouvelle image est blanc :




$$v = 128 + \left\lfloor \frac{255 * 3 - 0 * 3}{2 * 3} \right\rfloor = 255$$

- Dans le cas d'une chute brutale du niveau de gris, le pixel dans la nouvelle image est noir :




$$v = 128 + \left\lfloor \frac{0 * 3 - 255 * 3}{2 * 3} \right\rfloor = 0$$

Ces deux cas ne sont bien entendus jamais rencontrés dans la réalité. En effet la route est plutôt grise que noire et les lignes blanches ne sont pas toujours à un niveau de gris de 255 (ombres, peinture effacée, etc.) Le profil d'une ligne de l'image avec un marquage après application de ce filtre

est donc de la forme idéale décrite par la Figure 134. Le premier pic (vers le haut) décrit le premier changement de contraste (i.e. de la route sombre vers la ligne blanche claire), le second pic (vers le bas) décrit le second changement de contraste (i.e. de la ligne blanche claire vers la route sombre).

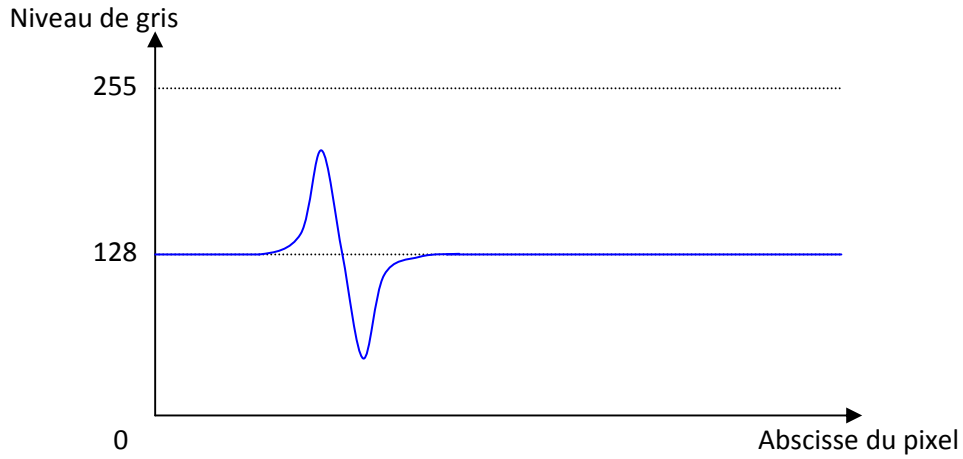



Figure 134 - Profil d'une ligne de l'image traité avec le fitre [-1 ... -1 1 ... 1]

### b) *Taille du filtre et réduction de bruit*

Un filtre efficace de détection de marquages doit être capable de faire ressortir le marquage tout en atténuant le bruit. Il apparaît que si le bruit sur l'image est uniforme, le bruit contenu dans les deux termes (PosSum et NegSum) de notre filtre [-1 … -1 0 1 … 1] a tendance à se compenser. On observe aussi que plus la moyenne spatiale est grande (c'est à dire plus L est grand), plus l'effet du bruit est atténué. Il y aurait donc intérêt à prendre une largeur de filtre L aussi grande que possible pour éliminer les effets du bruit. Voici quelques résultats de simulation qui montrent cet effet sur un marquage au sol de largeur 21 pixels dans l'image avec du bruit uniforme d'une amplitude de 30 (Figure 135 à Figure 138).

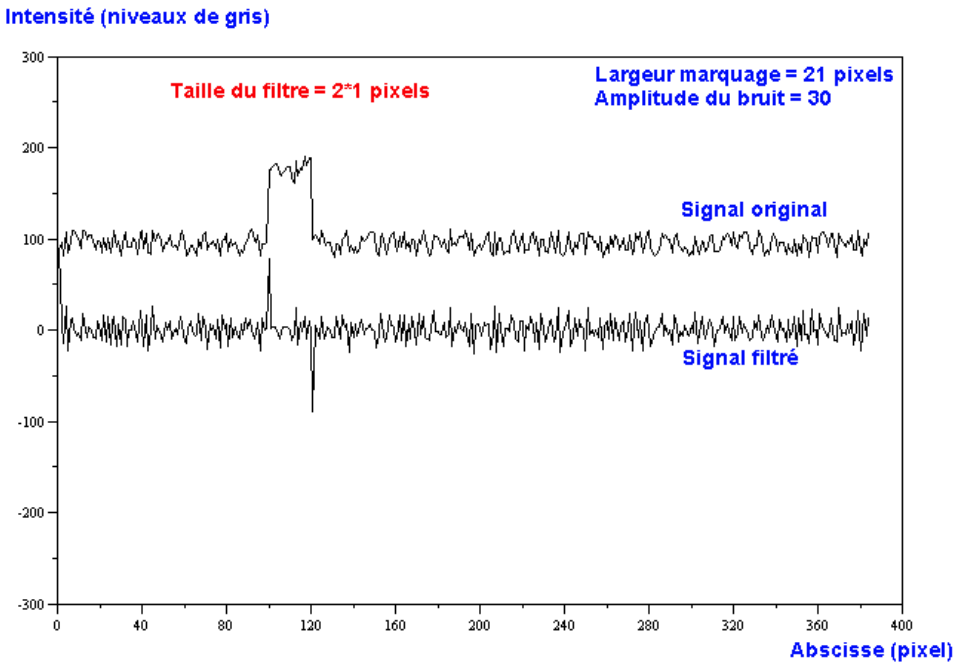



Figure 135 - Filtre de demi-longueur 1 sur un marquage de longueur 21 pixels. Le signal original représente une ligne dans l'image.

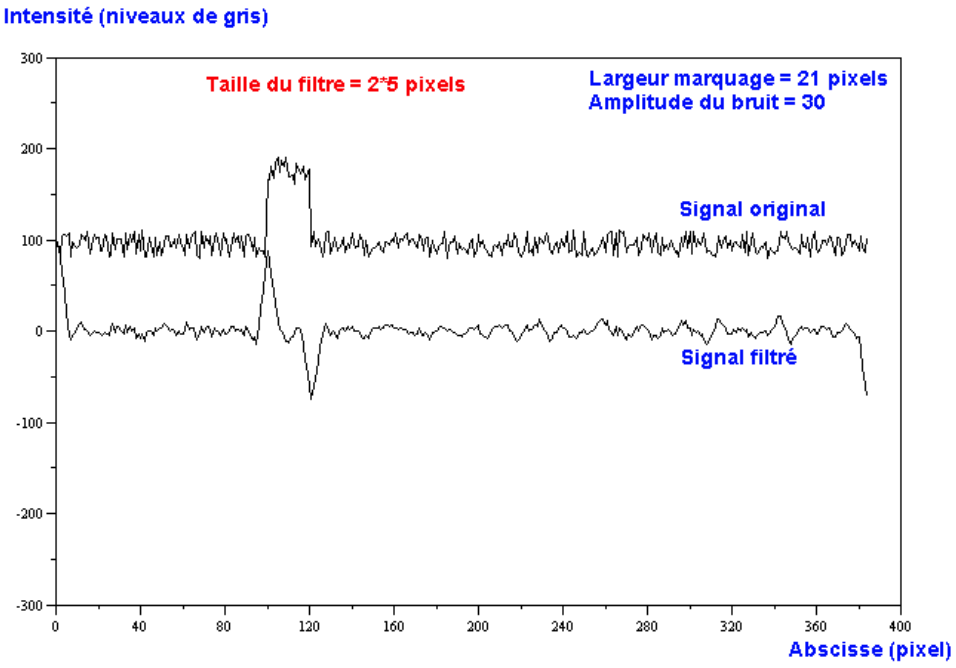



Figure 136 - Filtre de demi-longueur 5 sur un marquage de longueur 21 pixels

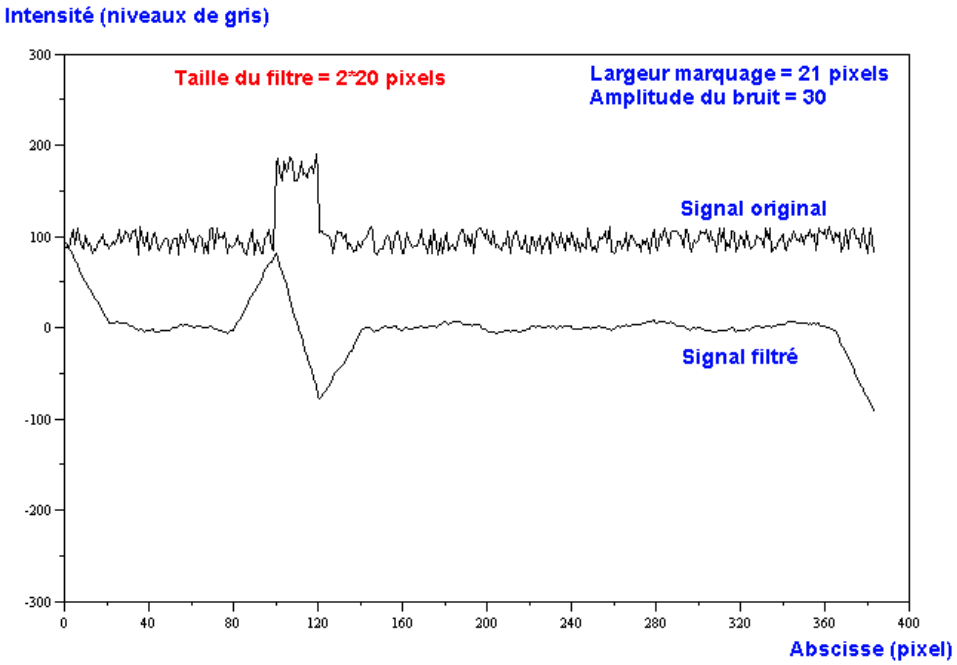



Figure 137 - Filtres de demi-longueur 20 sur un marquage de longueur 21 pixels

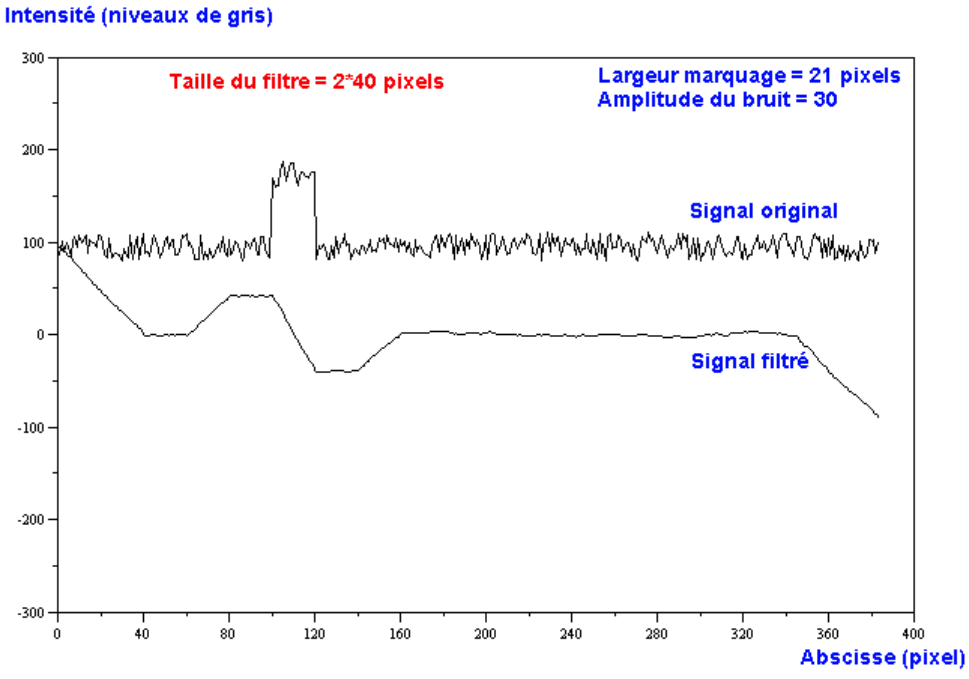



Figure 138 - Filtre de demi-longueur 40 sur un marquage de longueur 21 pixels

Observons maintenant quelques résultats de simulation qui montrent cet effet sur un marquage au sol idéal (i.e. sans bruit) de largeur 21 pixels dans l'image (Figure 139 à Figure 141).

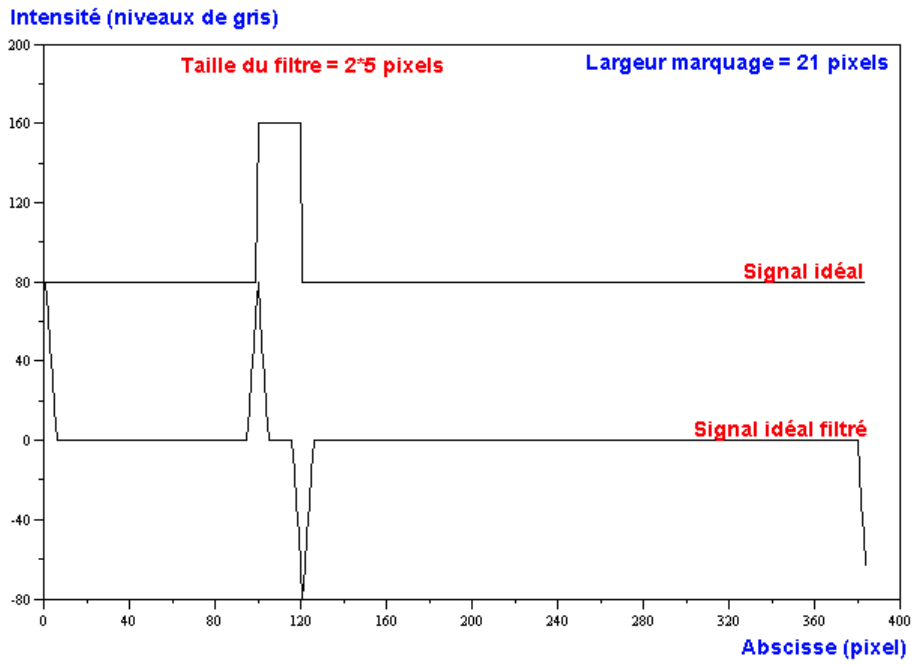



Figure 139 - Effet du filtre de demi-longueur 5 pixels sur un créneau idéal

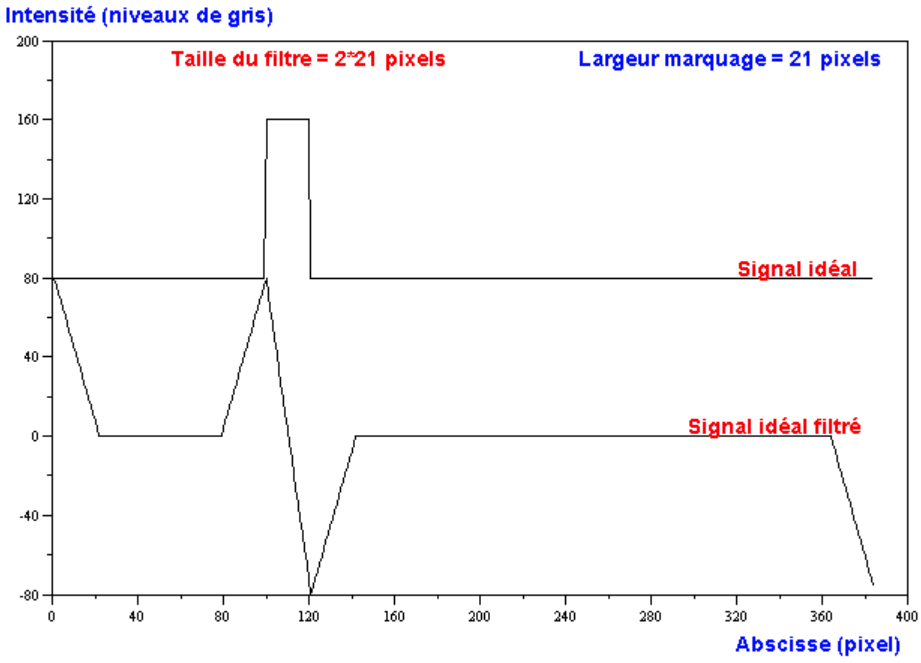



Figure 140 - Effet du filtre de demi-longueur 21 pixels sur un créneau idéal

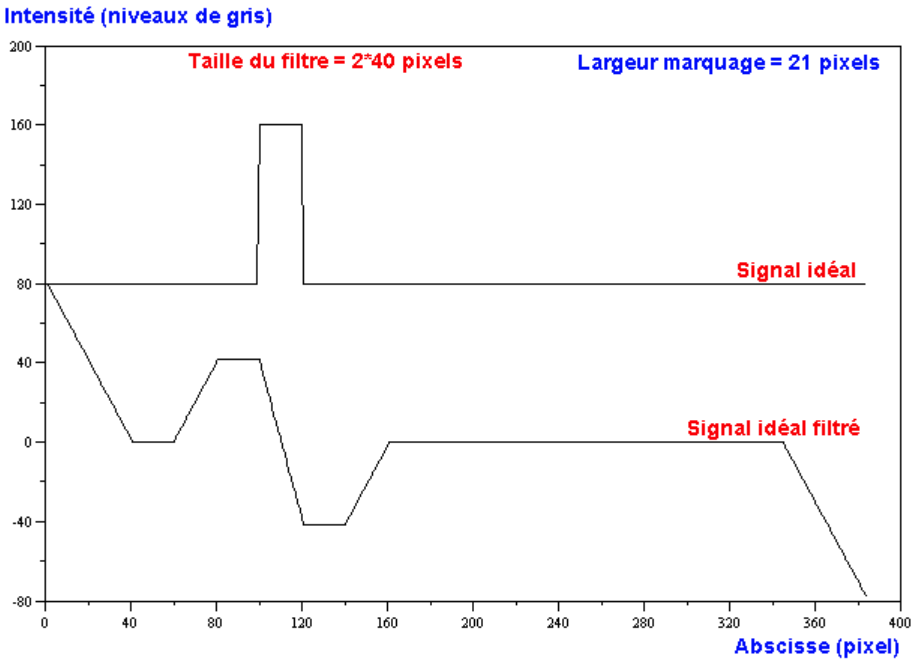



Figure 141 - Effet du filtre de demi-longueur 40 pixels sur un créneau idéal

Ces résultats montrent que tant que la largeur du filtre ne dépasse pas celle du marquage, il y a création d'une dent de scie positive centrée sur le début du marquage suivi d'une dent de scie négative centrée sur la fin du marquage. La largeur de chaque dent de scie est égale à 2*L (cela correspond à la largeur du filtre), et l'amplitude entre le maximum et le minimum est égale au double de celle du marquage idéal non filtré. En revanche, lorsque la demi-largeur du filtre devient supérieure à la taille du marquage, les pics sont d'autant plus érodés que L augmente.

Optimiser le rapport Signal/Bruit demande donc d'utiliser la demi-largeur de filtre la plus grande possible sans dépasser la largeur (ou taille) du marquage.

### c) *Filtre adaptatif*

Les différentes approches bibliographiques utilisant un filtre de convolution de type [-1…-1 1…1] ont toujours proposé un filtre de taille constante sur l'ensemble de l'image. Ces approches ne tiennent pas compte du fait que l'effet de ces filtres est différent suivant la hauteur à laquelle on se trouve dans l'image, c'est-à-dire suivant la largeur du marquage dans l'image.

En effet, pour une taille de marquage donnée, la taille de filtre la mieux adaptée (c'est-à-dire qui gomme au mieux le bruit sans éroder les pics correspondants aux bords des marquages gauches et droits) est celle qui est égale à la largeur du marquage en pixels, pour la ligne de l'image considérée. La largeur de marquage variant pratiquement linéairement suivant l'ordonnée dans l'image, il en est de même pour la largeur du filtre. Par exemple, Figure 142, le filtre a été appliqué sans variation de

sa taille ; en revanche, Figure 143, la taille du filtre a été adaptée en fonction de la largeur du marquage attendu. Le bruit s'en trouve fortement atténué et les marquages ressortent clairement sur cette dernière image.

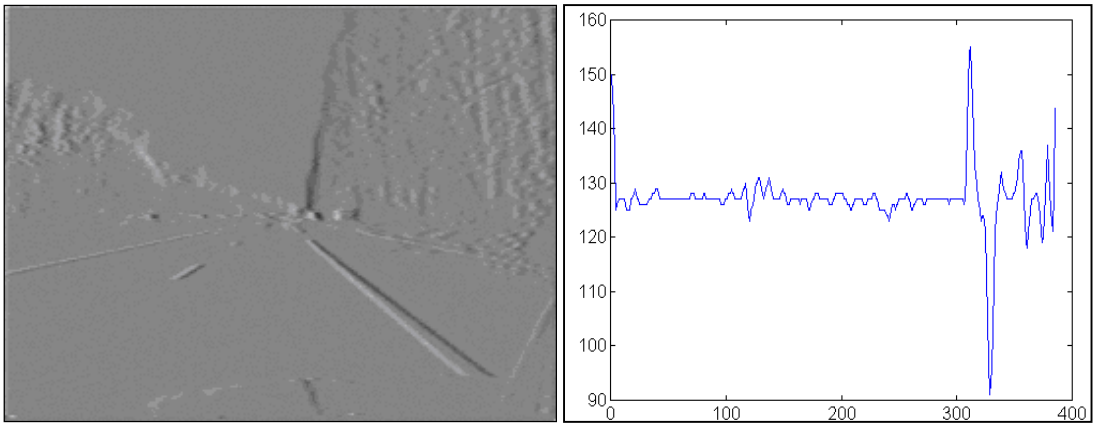



Figure 142 - Application d'un filtre de taille constante sur l'image

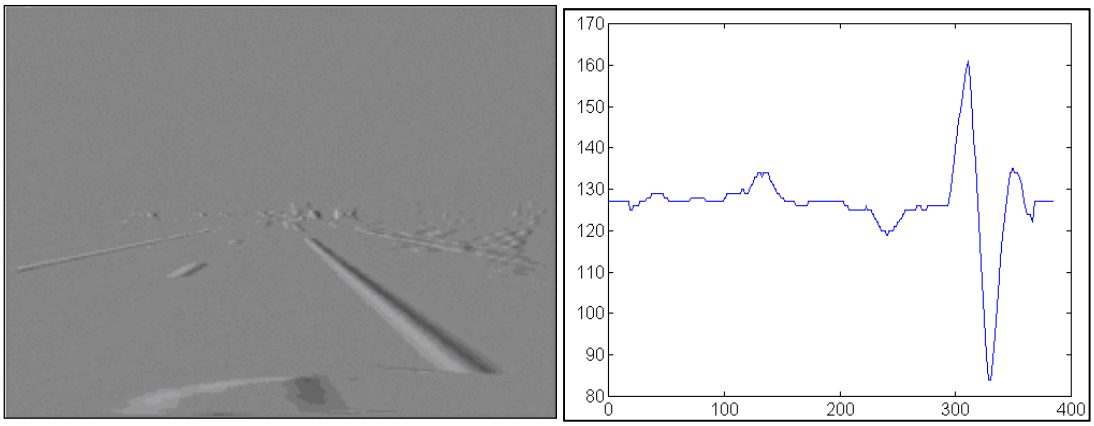



Figure 143 - Application d'un filtre de taille variable sur l'image

Ainsi, pour obtenir une détection optimale, les paramètres du filtre devraient être adaptés en fonction de la taille du marquage attendu dans l'image qui varie selon de multiples facteurs (variations d'assiette du véhicule d'une image à la suivante, inclinaison de la caméra, position latérale sur la voie…).

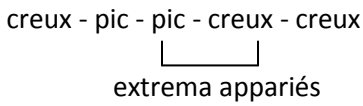
### *d) De l'image des contours aux centres des marquages*

Après application du filtre adaptatif de convolution on dispose d'une nouvelle image en niveaux de gris dont il est désormais possible d'extraire les lignes. La présence d'un marquage sur la route se traduit dans l'image filtrée par la présence d'un maximum d'intensité suivi d'un minimum. Cette

propriété nous permet d'utiliser un algorithme relativement simple pour détecter les marquages en recherchant la présence sur chaque ligne de cet enchaînement maximum local – minimum local.

Pour ce faire, l'algorithme recense tous les minima et maxima locaux qui sont situés au-delà (pour les maxima) ou en deçà (pour les minima) d'un seuil défini à l'avance (Figure 144).

Une fois les extrema détectés, l'algorithme les apparie (il fait correspondre un maximum avec un minimum). Pour être appariés, un pic et un creux doivent forcément se trouver dans cet ordre (il ne peut y avoir un creux puis un pic, car un marquage est une succession foncé – clair – foncé), et tel que la distance entre les deux soit la plus faible possible (et dans tous les cas inférieure à un seuil fixé à l'avance : la largeur maximale de marquage attendue). Les extrema qui ne sont pas appariés ne sont pas considérés comme étant des bords de marquage (Exemple 3).




Exemple 3 - Appariement des extrema pour la détection des marquages

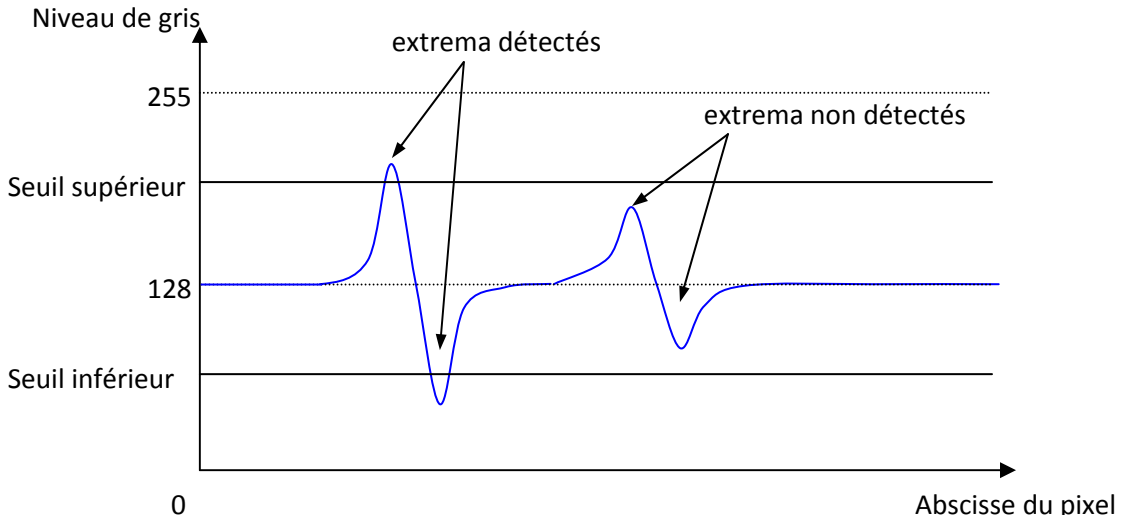



Figure 144 - Filtrage des extrema dans l'image filtrée avec le filtre [-1 … -1 1 … 1]

L'un des principaux problèmes soulevé dans la littérature est la détection de ligne quand des ombres sont présentes sur la chaussée (ombres des arbres sur une route boisée par exemple) qui font apparaître de forts contrastes (alternance ombre – soleil) entre la route et les marquages au sol d'une zone à l'autre de l'image. L'utilisation ici d'une approche locale pour la recherche des transitions maximum – minimum nous permet de nous affranchir des problèmes de changement de contraste dans l'image. Enfin, la connaissance a priori sur la largeur d'un marquage au sol selon son

ordonnée dans l'image est utilisée pour ne conserver que les transitions maxima – minima de largeurs comprises dans l'intervalle de largeur de marquage souhaité.

Ensuite est calculé le milieu entre le pic et le creux, qui est enfin enregistré comme étant le centre du marquage. Les points aberrants sont ensuite éliminés en considérant le fait que les marquages s'étalent sur plusieurs lignes de l'image tout en gardant une continuité au niveau de leur largeur et de leur positionnement. La Figure 145 montre un exemple de détection des centres des marquages.

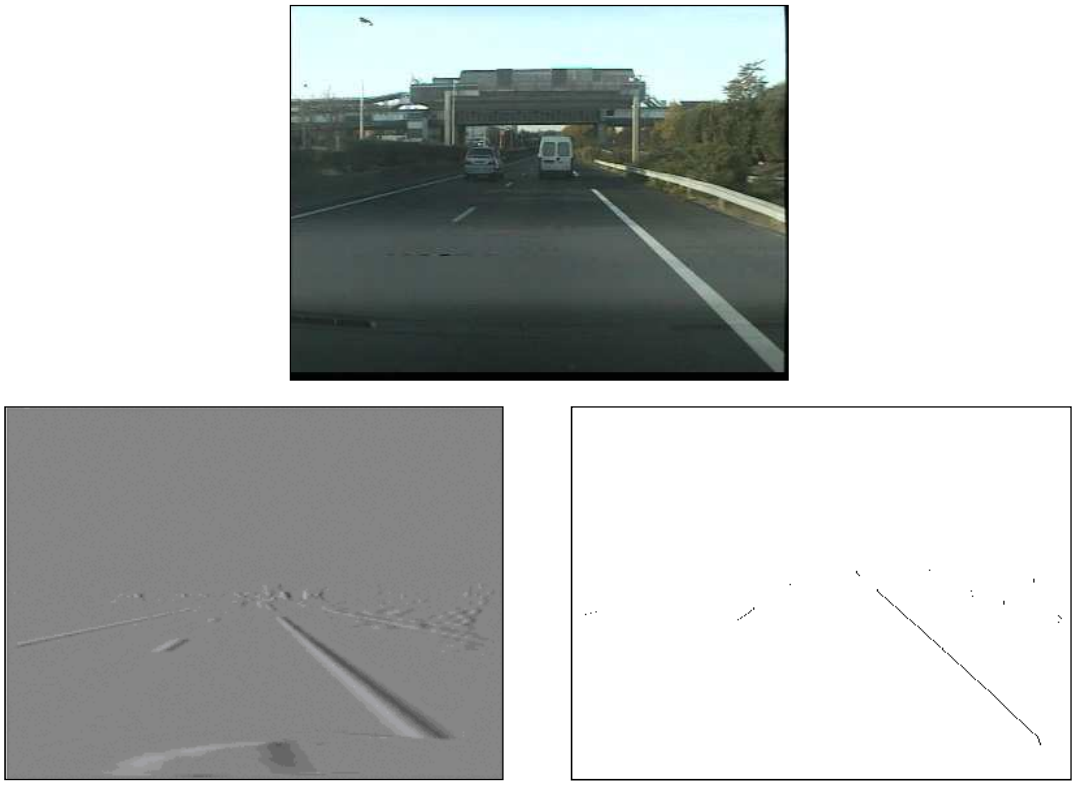


Figure 145 - Exemple de détection des centres de marquage

## 3.3.2. Détection des lignes

Après la phase d'extraction nous disposons d'un ensemble de points correspondant aux centres des différents marquages présents sur la route. Dans cet ensemble sont aussi présents des points qui n'appartiennent à aucun marquage et qui n'ont pas été éliminés par le premier filtrage (Figure 145).

Pour éliminer le bruit de cet ensemble et regrouper les points correspondant aux différents marquages entre eux, l'algorithme de Hough est utilisé. En effet, comme mentionné précédemment dans l'état de l'art, la transformation de Hough est une technique optimale pour détecter les droites

dans une image même très bruitées car cette technique ne dépend pas de la continuité des droites dans l'image.

Idéalement l'algorithme de Hough devrait trouver les lignes de marquage au sol (Figure 146). Malheureusement, cet algorithme n'est pas magique, le cas idéal ne se présente jamais et de nombreux problèmes apparaissent lors de l'utilisation de cette technique.

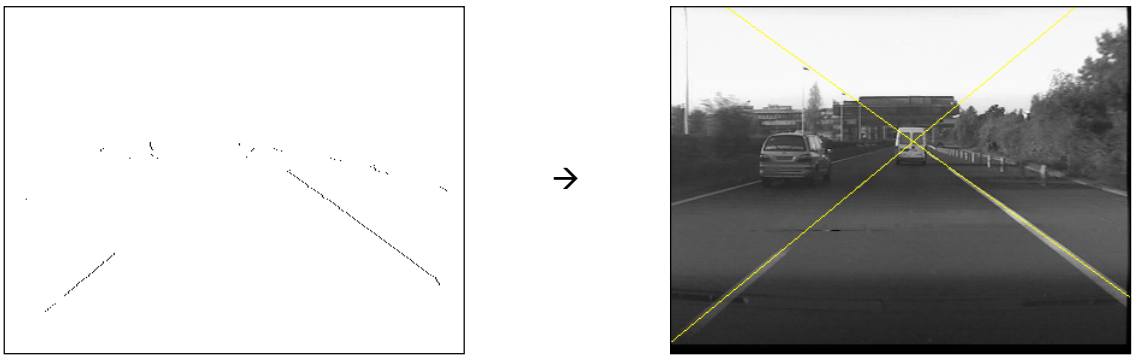


Figure 146 - Cas idéal de l'application de la transformée de Hough

### a) *Plusieurs droites détectées pour une même ligne*

La transformation de Hough permet de détecter la présence de droites dans une image mais parfois propose plusieurs droites pour représenter en fait la même ligne. Ce problème vient du fait que les pixels représentant une droite sont rarement alignés. Il est donc nécessaire de filtrer les résultats de l'algorithme en regroupant les droites détectées qui sont similaires (i.e. ayant sensiblement les mêmes paramètres). Chaque groupe est ensuite représenté par une droite résultant de la moyenne de toutes les droites du groupe (Figure 147).

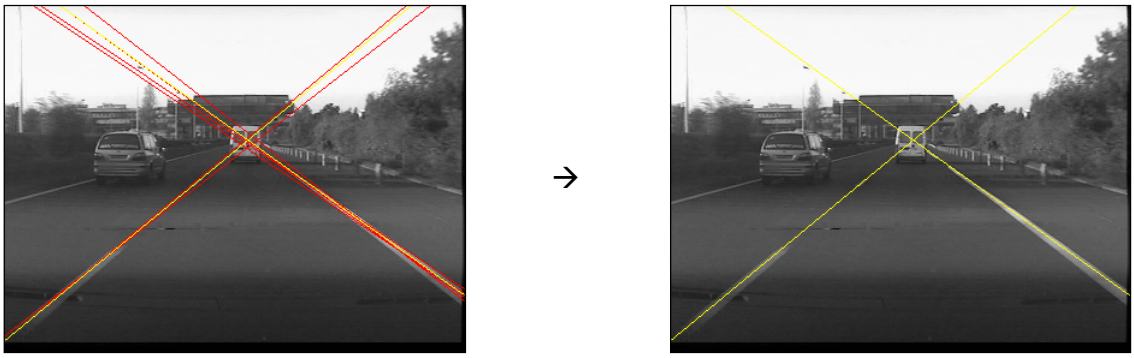


Figure 147 - La transformée de Hough détecte plusieurs droite pour une même ligne

### b) *Fausses détections*

Pour de nombreuses raisons, l'application de la transformée de Hough pour détecter les lignes de marquage au sol peut détecter des lignes qui ne sont pas celles des marquages. Par exemple, dans la Figure 148, l'application du filtre fait apparaître des lignes au niveau des poteaux qui forment dans l'image une alternance foncé – clair – foncé comme le ferait une ligne de marquage au sol.

Pour pallier à ce problème, plusieurs processus entrent en jeu.

Premièrement, la détection de contour n'a lieu que sur la partie inférieure de l'image. En effet, comme décrit précédemment, l'application du filtre adaptatif d'extraction de contour, recherche ligne par ligne dans l'image des alternances foncé – clair – foncé de taille variable corrélée à la hauteur dans l'image. Ainsi au dessus de l'horizon, aucune alternance n'est cherchée puisque la largeur hypothétique d'une ligne serait en dessous de 0 pixel.

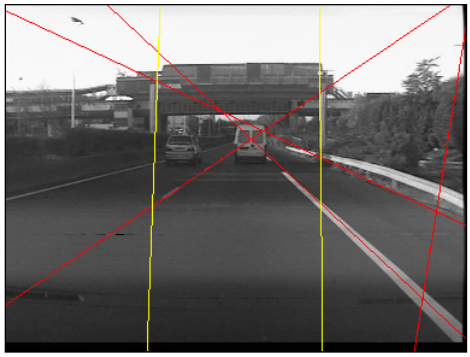


Figure 148 - Cas de mauvaises détections par la transformée de Hough

Deuxièmement les droites dont la position dans l'image est telle qu'elles ne peuvent pas provenir de lignes de marquage au sol sont éliminées. En particulier, toutes les droites horizontales, ou faisant un angle trop faible avec l'horizontale, sont considérées comme non admissibles. Les droites « trop verticales » et situées vers le bord de l'écran sont elles aussi déclarées comme non admissibles, étant donné le fait que la voiture possède toujours un angle de lacet limité par rapport à sa voie (Figure 149).

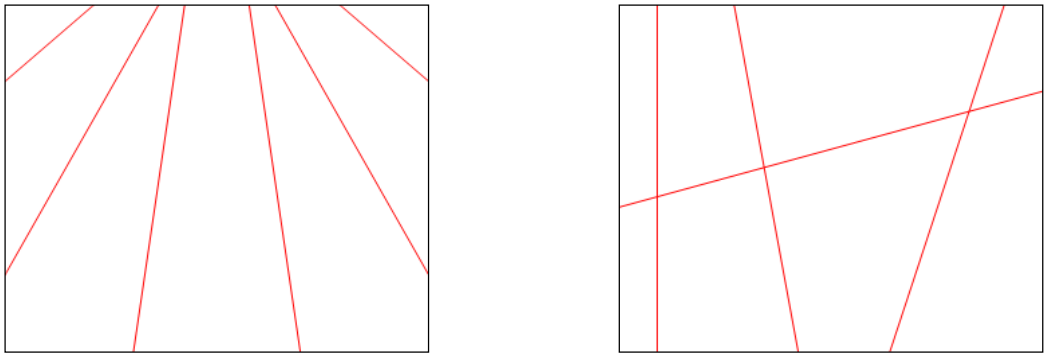


Figure 149 - Exemples de droites admissibles (à gauche) et non admissibles (à droite)

Enfin, la dernière méthode utilisée pour filtrer les fausses détections dans l'image est l'utilisation d'un filtre temporel. En effet, les fausses détections sont habituellement très ponctuelles. Ainsi, si une ligne est détectée plus de X images à la suite, cette ligne est considérée comme ligne de marquage au sol et non plus comme du bruit de détection. Le nombre 'X' d'images est un nombre très faible de l'ordre de 5, c'est-à-dire qu'une ligne doit être détectée sur plus d'un cinquième de seconde à la suite.

### c) *Obstructions et lignes pointillées*

Le problème qui apparaît lors de l'étape de détection des lignes de marquage au sol est celui de l'absence de ligne à détecter. Ce problème intervient dans deux situations :

- Dans le cas des lignes pointillées (milieu de route) qui produisent une alternance régulière dans le flux vidéo de la présence / absence d'un marquage (Figure 150).

Figure 150 - Ligne pointillé visible (à gauche) et 'non' visible (à droite)

- Dans le cas d'une obstruction, quand la ligne de marquage au sol est cachée, généralement par une voiture qui change de voie (Figure 151).

Figure 151 - Cas d'une ligne de marquage au sol occultée

Il est clair que ces deux cas engendrent des problèmes de lignes non visibles temporairement, sur un laps de temps assez court. La solution la plus simple du problème apparaît alors comme l'utilisation d'un filtre de suivi spatio-temporel.

Tout d'abord ce filtre utilise un tampon de stockage des lignes, c'est-à-dire d'une zone mémoire des lignes présentes à l'image précédente dans le flux vidéo. En effet, si une ligne est présente à l'instant 't-1' elle sera forcément présente à l'instant 't' : une ligne de marquage au sol ne disparaît pas de l'image instantanément. La longévité de cette mémoire tampon est intimement liée à la vitesse de la voiture. En effet, prenons le cas des lignes pointillées, l'absence temporaire du marquage saura d'autant plus long que la voiture roule lentement (et inversement).

Ensuite ce filtre apparie les lignes de sa mémoire tampon, avec celles détectés à l'instant courant. Cet appariement se fait de la même façon que lors du regroupement de toutes les droites détectées pour une même ligne.

Une dernière étape de l'algorithme consiste à calculer l'intersection des lignes détectés (qui correspond normalement au point de fuite dans l'image) afin d'utiliser la hauteur de ce point comme hauteur de l'horizon en paramètre de l'application du filtre d'extraction de contour, pour qu'il adapte dynamiquement la taille du filtre utilisé.

## 3.4. Expérimentation et résultats

Voici pour résumer comment fonctionne l'algorithme de détection des lignes de marquage au sol mis en place (Figure 152):

1. A partir d'une image en niveau de gris
2. Un filtre [-1 … -1 1 … 1] de largeur adaptative d'extraction de contour est appliqué
3. De l'image filtrée sont extraits les centres des marquages
4. L'application de l'algorithme de Hough permet d'y trouver les lignes
5. Un filtrage des résultats frame par frame et temporellement permet de ne conserver que la détection des lignes de marquage au sol

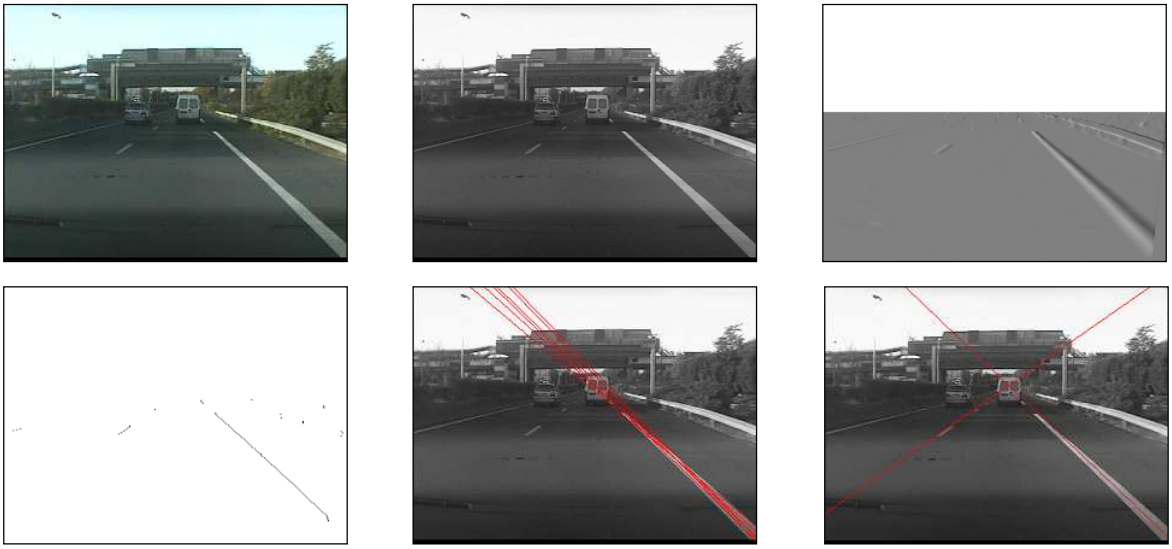


Figure 152 - Etapes successives de la détection des lignes de marquage au sol

Cet algorithme a été soumis à de nombreux cas de figures différents et répond aux problèmes exposés par l'ensemble des auteurs des algorithmes vus dans l'état de l'art, dont voici quelques résultats.

### *a) Cas d'ombres sur la route*

Les ombres peuvent produire des artefacts à la surface de la route pour l'extraction de contours et détériorer les résultats de la détection des lignes de marquage au sol soit par la présence d'une alternance uniforme de grosses zones ombragées ou non, soit par la présence de multiples petites

« taches » d'ombres. L'utilisation couplée du filtre adaptatif (qui supprime les alternances non voulues des zones foncé – clair – foncé) et de l'algorithme de Hough (qui arrive à retrouver des lignes dans un nuage de point même bruité) permet de bien distinguer les lignes dans ces cas de figure (Figure 153).

Figure 153 - Détection des lignes de marquage au sol avec des ombres présentes sur la route

### b) *Cas des lignes partiellement occultées*

Le passage d'une voiture d'une voie à une autre occulte temporairement la ligne de marquage au sol qui sépare les deux voies. Le filtre de suivi spatio-temporel permet de palier à ce problème (Figure 155). Il est notamment intéressant de voir que sur une frame, d'après l'image des contours, l'algorithme arrive à « reconnaître » les lignes de marquage au sol dans un tel cas (Figure 154).

**Figure 154 - Détection des lignes de marquage au sol (à droite) et image des contours (à gauche) lors d'occultation des lignes**

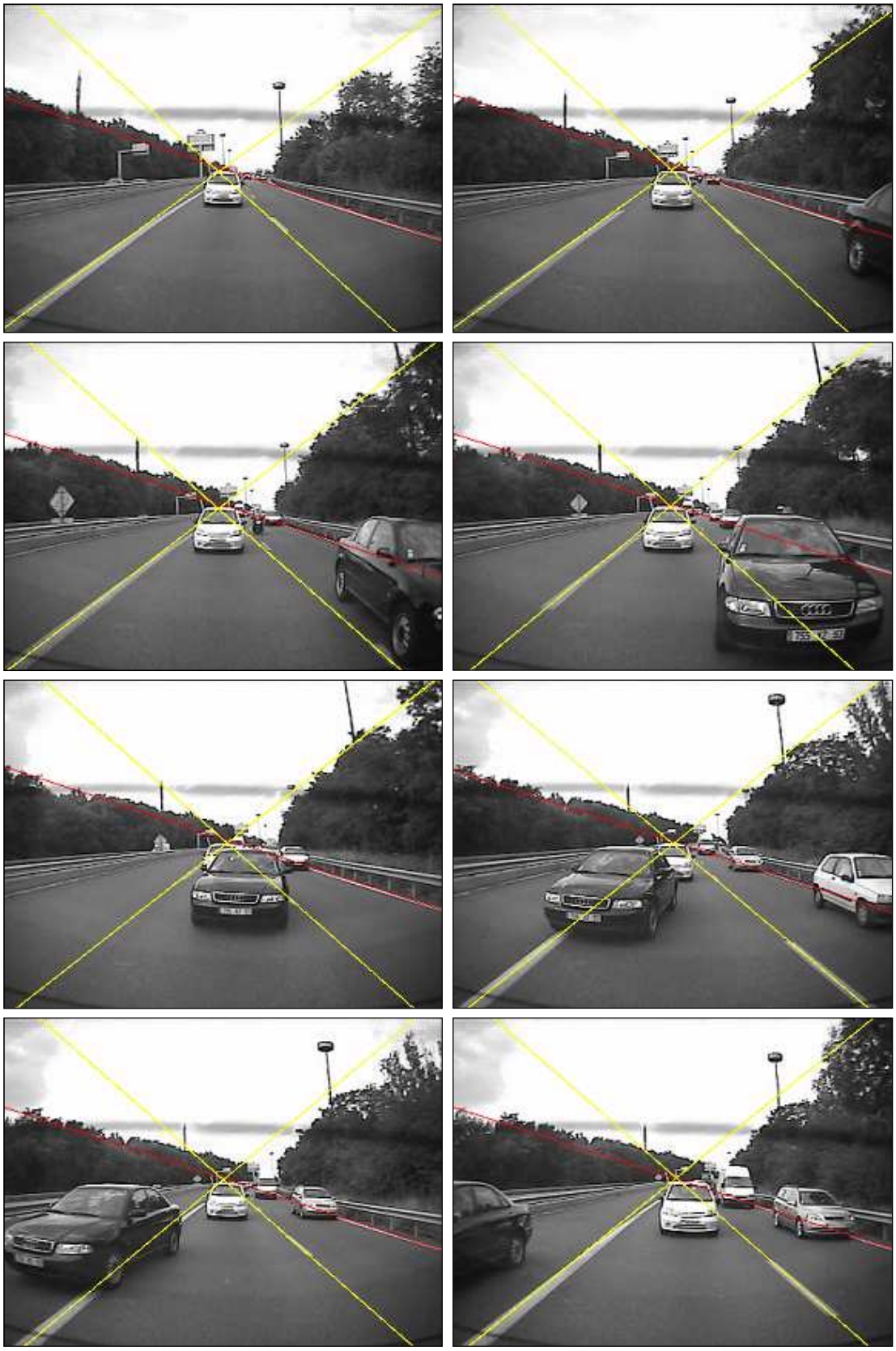


Figure 155 – Détection des lignes de marquage au sol lors d'un changement de voie d'une voiture

### c) *Cas des différentes conditions de luminosités*

Les changements de conditions lumineuses, qui peuvent arriver par exemple lors de passage sous un pont, font drastiquement changer les contrastes des lignes de marquage au sol par rapport à la route dans l'image qui devient beaucoup plus clair ou beaucoup plus foncé. L'utilisation d'un seuil local lors de la sélection des extrema dans l'image des contours permet d'outre passer ce problème relevé dans la littérature (Figure 156).

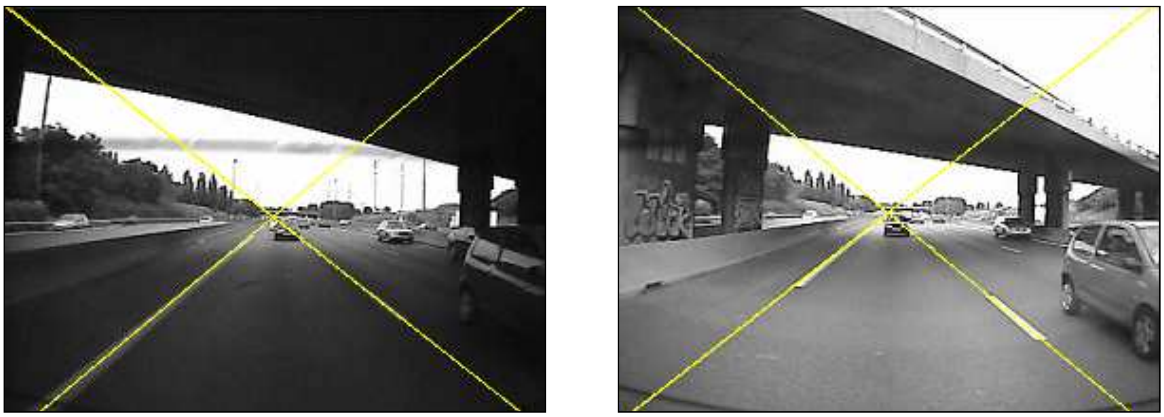


**Figure 156 - Détection des lignes de marquage au sol sous différentes conditions lumineuses**

### d) *Cas de changement de voie*

Le cas des changements de voie est très peu décrit dans la littérature. Or, l'utilisation d'une mémoire pour pallier aux problèmes d'occultation peut faire apparaître des problèmes de détection lors des changements de voie de son propre véhicule. En effet, comme le véhicule se déplace latéralement, les lignes détectées se déplaceront d'un coté à l'autre de l'image (en fait elles subiront une rotation autour du point de fuite de l'image) plus ou moins rapidement selon le temps mis pour effectuer le changement de voie par le véhicule (Figure 157). Un appariement spatio-temporel qui ne tiendrait pas compte de cette problématique ferait apparaître des artefacts de détections entre la détection précédente et la détection courante. Ici, cette problématique a été étudiée (Figure 158).

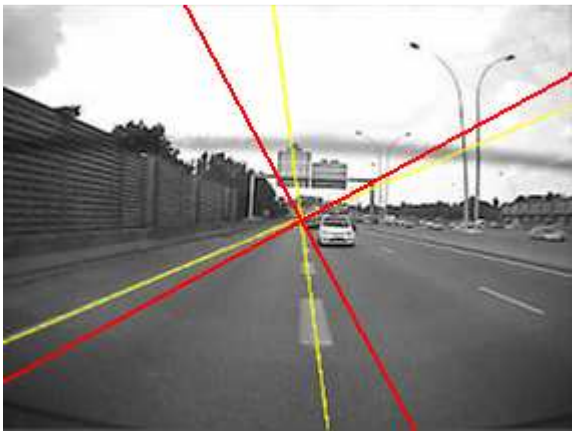


**Figure 157 - Détection précédente (en rouge) et courante (en jaune) des lignes de marquage au sol lors d'un changement de voie du véhicule (de la gauche vers la droite)**

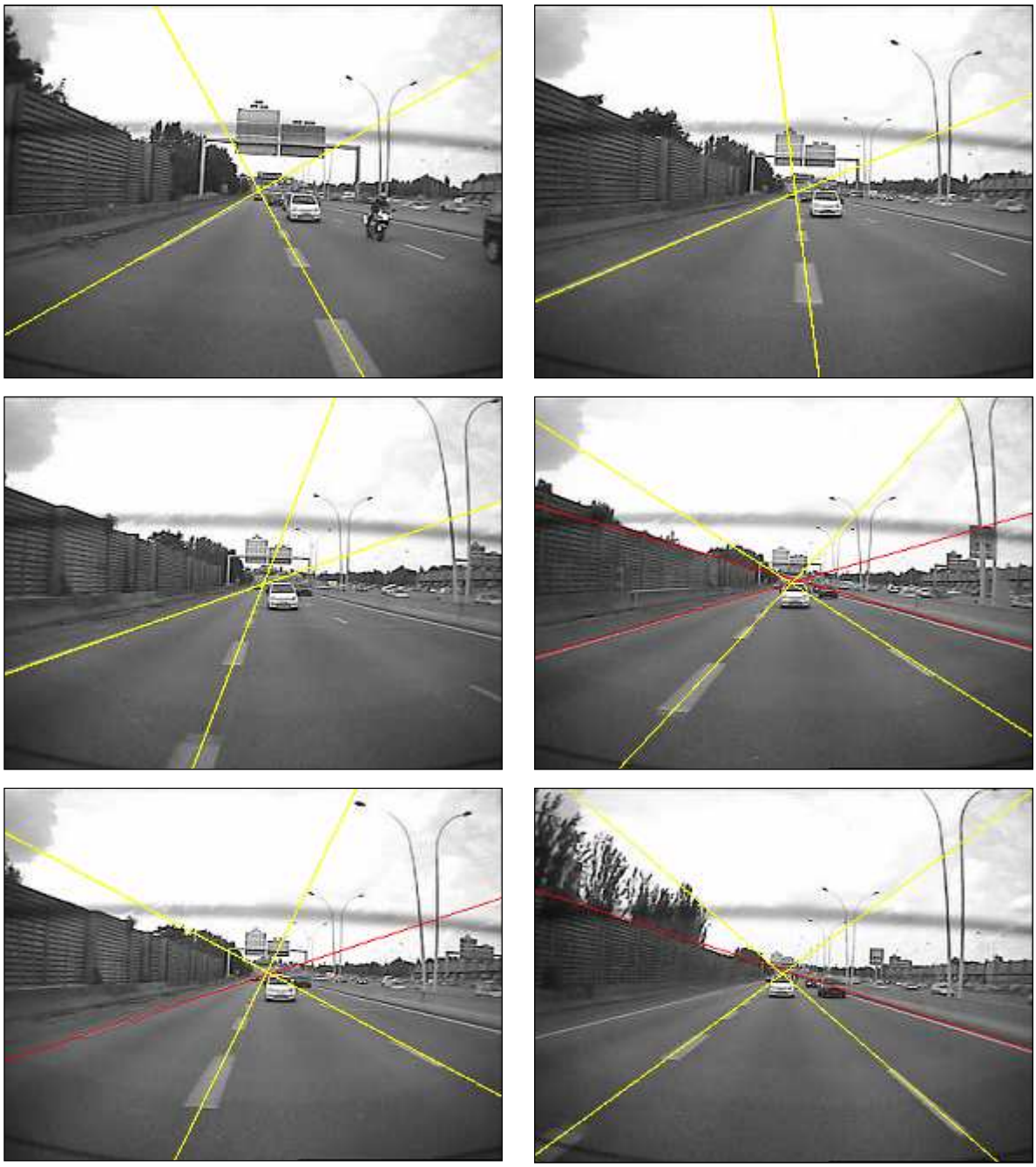


Figure 158 - Détection des lignes de marquage au sol lors d'un changement de voie de son propre véhicule

## 3.5. Perspectives et applications

Un point difficile soulevé à plusieurs reprises par les évaluateurs industriels et par des observateurs extérieurs, est l'existence pendant les périodes de travaux de zones comportant plusieurs marquages superposés, l'un temporaire et valide, l'autre non valide, existant au préalable et restant visible au sol. Face à cette situation, l'être humain fait naturellement appel à des éléments extérieurs pour analyser la scène de travaux. Aucun algorithme de vision ne sait traiter ce problème de manière systématique, ce qui est aussi le cas de l'algorithme présenté ici, incapable de faire cette différence.

L'utilisation de l'information de couleur, si une image en couleur est disponible, pourrait aider à résoudre ce problème en distinguant les lignes temporaires (jaunes) des lignes officielles (blanches).

La modélisation de la route sous forme de ligne droite, permet d'envisager de nombreuses applications et notamment celle qui nous intéresse ici : la détection des changements des voie de notre véhicule. Cependant, améliorer cette modélisation afin d'estimer la courbure de la route, ne devrait pas poser beaucoup de problème au vu de la méthode couramment utilisé dans la littérature : effectuer une détection de lignes par transformée de Hough, non pas comme il est fait ici sur toute l'image, mais sur 5 zones horizontales prédéfinies de l'image, puis calculer les paramètres d'une courbe de Bézier / B-Spline correspondant au mieux à ces séries de 5 segments.

L'utilisation de cet algorithme en vue d'implémenter des algorithmes ADAS de plus haut niveau est envisageable. Par exemple il est possible de calculer dynamiquement le tangage, le roulis et le lacet du véhicule en comparant la position du point de fuite détecté (point de croisement des lignes) avec la position du centre optique de la caméra dans l'image (position idéale du point de fuite sans mouvement du véhicule). L'utilisation de cette information peut permettre de stabiliser le flux vidéo en corrigeant chacune des images (par le calcul d'une rotation en 3 dimensions des pixels de l'image) par rapport à ces paramètres comme le font (Liang, et al., 2004). Cette méthode a été implémentée dans le cadre de mon stage (Bargeton, 2005).

Plusieurs autres applications décrites dans la première partie de ce chapitre peuvent être facilement mis en place à partir de cet algorithme de détection de ligne de marquage au sol, comme par exemple :

- La prévention de sortie de voie en détectant si une ligne se rapproche dangereusement du centre de l'image ;

- Coupler à une détection d'obstacle, il serait aisé, en ligne de droite, de déterminer si cet obstacle est présent dans notre voie ou non ;

- Créer un système auto cruise plus intelligent qui garderait le véhicule dans sa voie ; et qui, couplé à un système de détection des limitations de vitesse, adapterait automatiquement la vitesse du véhicule à la vitesse légale.

# Vers un système complet

## 4.1. Motivations

L'objectif du travail présenté dans ce mémoire est la réalisation d'un système de détection des limitations de vitesse afin d'aider le conducteur à gérer au mieux sa conduite (i.e. ne pas commettre d'infraction en roulant trop ou trop peu vite, ne pas se mettre dans des situations dangereuses,…).

Nous avons vu dans les chapitres précédents qu'un capteur seul ne suffit pas afin de répondre à cette problématique.

Un capteur cartographique par exemple, ne peut connaître avec précision la vitesse limite concernant un véhicule. Cela est dû à l'imprécision physique du positionnement GPS, mais aussi au caractère statique de sa base de connaissance qui ne peut donc connaître les changements durables du système routier (nouvelle route, changement de limitation de vitesse d'une rue,…), et encore moins les changements temporaires (travaux sur une route, limites de vitesse variables,…).

Un système de détection et de reconnaissance visuelles des panneaux de limitation de vitesse souffre quant à lui de l'impossibilité de détecter toujours les panneaux et d'un manque d'interprétation. En effet, il y aura toujours des cas où un panneau reste partiellement ou totalement occulté par un autre véhicule durant la totalité du moment où on aurait pu le voir. Par ailleurs, tous les panneaux présents sur la route ne sont pas dédiés à tous les véhicules (certains réglementent la vitesse pour les camions uniquement, d'autres pour les voies de sortie,…).

Il n'est donc pas envisageable de se baser uniquement sur l'un ou l'autre de ces deux capteurs. Il est donc nécessaire d'utiliser plusieurs sources d'informations afin de compléter la connaissance sur l'environnement routier pour que le système développé puisse prendre une décision quant à la valeur réelle de la limitation de vitesse à laquelle est soumis le véhicule

C'est donc dans ce chapitre que le titre de ce mémoire, *Fusion multi sources pour l'interprétation de l'environnement routier*, prend tout son intérêt. Les briques élémentaires (détection des panneaux, détection des marquages au sol et capteur cartographique) nécessaires à la réalisation du système de détection des limitations de vitesse ont été présentées dans les chapitres précédents. Nous allons voir ici comment les fusionner pour augmenter la connaissance du système sur l'environnement routier qui entoure le véhicule et ainsi trancher quant à la limitation à laquelle est soumis le véhicule.

## 4.2. Fusion de données multi sources

### 4.2.1. Définition

La fusion de données multi sensorielle n'est pas une idée nouvelle. Les humains et les animaux ont naturellement développé la combinaison de leurs divers sens pour assurer leur survie : lorsque la vision est réduite la nuit, l'ouïe et l'odorat apportent des informations complémentaires qui améliorent la « perception » de la scène ambiante.

Un exemple plus proche de la thématique de ce mémoire est celui se ramenant à la conduite automobile lors de laquelle l'homme corrèle les différentes informations lui provenant de :

- sa vue (frontale, latérale, arrière (rétroviseur))
- son ouïe (bruit du moteur, klaxon, clignotant, frein)
- son toucher (retour d'effort volant, pédales)
- son équilibre (accélération, virage,…)
- ses connaissances (trajet, règles de conduite, autres usagers,…)

La définition de la fusion d'informations communément admise dans la littérature est la suivante :

> « La fusion d'informations consiste à combiner des informations issues de plusieurs sources afin d'améliorer la prise de décision. » (Bloch, 2003)

La fusion de données multi-sensorielle homogène ou hétérogène peut donc être vue comme l'utilisation de données fournies par plusieurs sources d'informations pour construire une source de connaissance, plus précise et plus riche que les informations initiales prises indépendamment les unes des autres et ceci dans le but de prendre une décision ou d'effectuer un diagnostic ou de réaliser une meilleur estimation.

### 4.2.2. Problématique

#### *a) Incertitude et imprécision*

La difficulté du problème de la fusion de données multi sensorielle réside dans le fait que les connaissances apportées par les différents capteurs peuvent être :

- incomplètes : certaines données ne sont pas disponibles
- imprécises : le contenu de l'information peut faire paraître plusieurs options (valeurs)
- floues : le contenu de certaines propositions est flou, énoncé de façon vague
- incertaines : la valeur de vérité d'une connaissance (existence) est connue avec plus ou moins de précision
- inconsistantes : les informations peuvent être contradictoires

Voyons ces notions dans une série d'exemples simples confrontant la distance d'un véhicule se situant devant nous.

Exemple 1 : Le véhicule se situe à une distance de 1,72m.
Proposition nette et propriété précise.

Exemple 2 : Le véhicule se situe à une distance entre 1,70m et 1,75m.
Proposition nette et propriété imprécise (appartenance à un intervalle).

Exemple 3 : Le véhicule se situe à une distance aux alentours de 1,72m.
Proposition floue (quel est l'intervalle de donnée ?).

Exemple 4 : Le véhicule se situe probablement à une distance de 1,72m.
Proposition incertaine.

Exemple 5 : Le véhicule se situe sans doute à une distance entre 1,70m et 1,75m.
Proposition incertaine et imprécise.

Exemple 6 : Le véhicule se situe probablement à une distance de 1,80m.
Le véhicule se situe certainement à une distance de plus de 1,60m.
On peut diminuer l'incertitude d'une proposition en augmentant son imprécision.

### b) *Redondance et complémentarité*

Ces propriétés concernent les sources d'informations. Celles-ci sont redondantes quand elles donnent des informations de même nature sur le même phénomène ou le même objet. Cette propriété est généralement exploitée pour améliorer la qualité des informations en termes de précision et d'incertitude.

Les sources d'informations sont complémentaires lorsqu'elles fournissent des informations sur des caractéristiques différentes du phénomène observé ou sur des objets différents. Cela permet d'obtenir une vision plus complète ou plus générale sur le phénomène.

Exemple 1 : Estimation d'une distance *d* par capteur ultrason et télémètre laser.
Les données sont redondantes.

Exemple2 : Détection des objets par leur forme et leur position.
Les données sont complémentaires.

### c) *Concordance et conflit*

Ces propriétés concernent les informations et donc les sources dont elles proviennent. Les informations sont en conflit lorsque leurs affirmations ne sont pas compatibles, c'est à dire qu'elles ne peuvent être vraies simultanément. Elles sont en concordance quand rien n'empêche qu'elles

soient vraies simultanément. Le conflit peut être partiel et ne porter que sur une partie des informations. Quand il n'y a aucun conflit, les informations sont concordantes.

Exemple 1 : Source 1 : la mesure $d \in [12,15]$
Source 2 : la mesure $d \in [10,13]$
Les informations données par les deux sources sont concordantes.
Si les données sont fusionnées, la qualité des informations est améliorée.

Exemple 2 : Source 1 : la mesure $d \in [12,15]$
Source 2 : la mesure $d \in [9,11]$
Les informations données par les deux sources sont en conflit.
Si les données sont fusionnées, la qualité du résultat est dégradée.

#### *d) Combinaison et décision*

La combinaison est l'opération fondamentale de la fusion de données. Elle permet d'obtenir une information qu'il ne serait pas possible d'avoir en n'utilisant qu'une seule source. Quand les sources sont concordantes, la qualité des informations en termes de précision et de certitude ne peut se dégrader, et la quantité d'information reste constante. Quand les sources sont discordantes, il est parfois possible de combiner quand même les informations, mais la qualité des informations est dégradée, et il y a souvent une perte d'information.

Lorsque tous les traitements et les combinaisons ont été réalisés, il s'agit de déterminer le résultat en fonction des informations et d'un ou de plusieurs critères de décision.

### 4.2.3. Topologies, niveaux et techniques de fusion

#### *a) Topologies*

Il existe de nombreux modèles d'architecture, souvent très liés au type d'application concerné. Ils sont dans ce cas assez précis mais peuvent difficilement être utilisés dans d'autres domaines d'application. Les modèles plus génériques sont, eux, très peu précis et n'ont donc pas toujours une grande utilité. Néanmoins, voici quelques exemples de modèles génériques les plus connus :

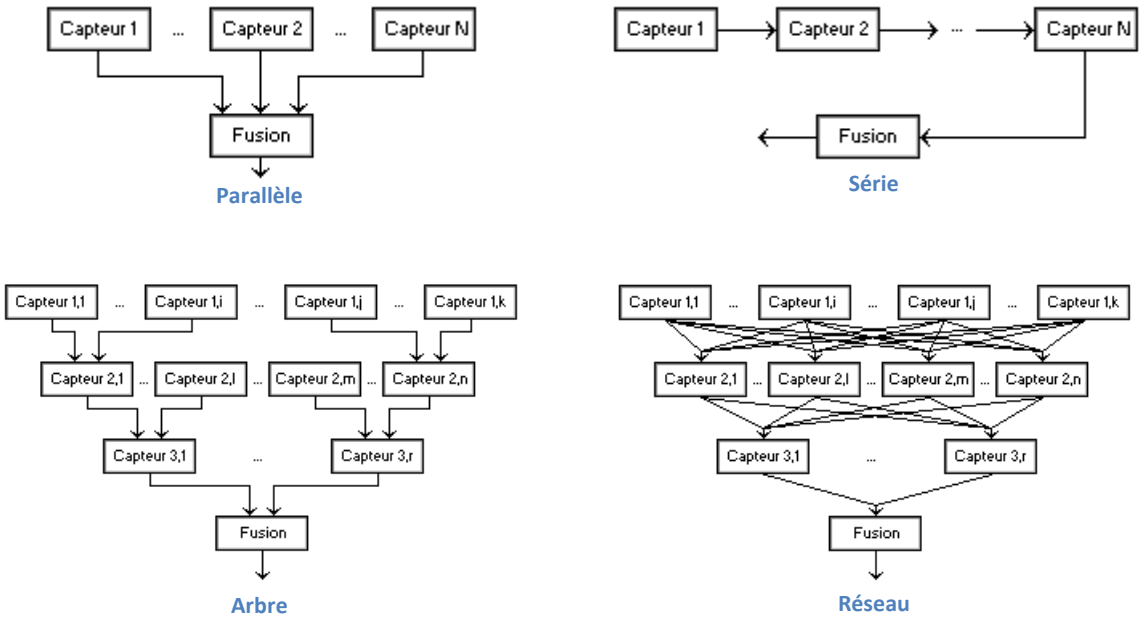



Figure 159 - Exemple de topologies de fusion de données

### b) *Niveaux de fusion*

Le niveau de fusion définit le niveau dans lequel les informations provenant des différents capteurs sont fusionnées. Généralement une distinction est réalisée entre 3 niveaux :

1. Niveau signal
   A ce niveau, la fusion consiste à combiner des signaux de même nature pour obtenir un signal de meilleure qualité (par exemple la combinaison de plusieurs images originales).

2. Niveau attribut
   A ce niveau, la fusion permet d'augmenter la confiance et la précision sur les attributs des objets (par exemple la taille), de créer de nouveaux attributs. Il s'agit du premier niveau d'abstraction, qui nécessite une première interprétation des informations.

3. Niveau symbolique
   Ce niveau correspond au plus haut niveau d'abstraction. Les informations fusionnées sont en général très hétérogènes, et correspondent à des zones d'observation, des objets ou des instants différents. La fusion permet alors d'avoir une vision beaucoup plus globale de la situation. A ce niveau, beaucoup d'informations a priori sont généralement utilisées.

Le choix d'un niveau de fusion dépend des types de capteurs disponibles :

- Quand les capteurs sont de même nature, il est possible d'effectuer la fusion au niveau signal afin de tenir compte de toutes les données.

- Quand les capteurs sont très différents, une fusion au niveau symbolique est plus adaptée et les calculs effectués sont plus efficaces.
- La fusion au niveau attribut est le niveau approprié quand les attributs trouvés par le traitement des différents capteurs peuvent être convenablement associés.

Dans la pratique, il ne semble pas toujours adapté de partitionner la fusion de données, car les interactions entre les niveaux, les traitements et les informations sont et doivent être nombreuses.

### c) *Techniques de fusion*

Les techniques utilisées pour la fusion d'informations sont nombreuses et dépendent du niveau de fusion choisi. (Linn, et al., 1991) ont relevé 5 types de fusion de données orientés par le but recherché : association des données, estimation, identification, reconnaissance de forme, et l'intelligence artificielle. De ces 5 catégories, 10 techniques de fusions peuvent être identifiées comme le montre le Tableau 15. Ce découpage en 3 niveaux avec ces différentes techniques a depuis été communément admis par la communauté scientifique avec quelques raffinements, dont (Klausner, et al., 2006) qui parle de niveau décisionnel pour le dernier niveau.

| Niveau de fusion | Méthode générale | Technique spécifique |
|---|---|---|
| **Niveau 1 Signal** | Association de données | Figure of merit (FOM)<br>Gating techniques |
| | Estimation | Filtres de Kalman |
| **Niveau 2 Attributs** | Identification | Réseaux bayésiens<br>Théorie de Dempster Shafer<br>Inférence classique |
| | Reconnaissance de forme | Méthodes à base de clusters<br>Réseau de neurones |
| **Niveau 3 Symbolique** | Intelligence artificielle | Systèmes experts<br>Blackboard<br>Logique floue |

**Tableau 15 - Techniques et niveaux de fusion**

Le choix d'une ou de plusieurs approches pour traiter une application particulière laisse souvent perplexe. Comme le suggère (Rombaut, 2001), ce choix doit être dicté en fonction d'un certain nombre de critères qui caractérisent le cadre d'application de chacune des techniques :

- Le type de connaissances dont on dispose sur le phénomène observé (statistiques, expertes, règles) ;
- La définition de l'objectif à atteindre (précision, incertitude, contraintes temporelles) ;
- La culture scientifique de l'utilisateur.

En fait, il ne faut pas vouloir comparer les performances des différentes approches sur une application particulière, car les résultats sont essentiellement fonction de la modélisation des

informations qui diffèrent suivant le formalisme choisi. Chacune des techniques est performante quand elle est utilisée dans le cadre qui lui convient.

Un état de l'art complet des techniques les plus couramment utilisés en fusion de données peut être trouvé dans (Rombaut, 2001), ainsi que dans (Smith, et al., 2006) qui fait un tour d'horizon plus complet en discutant des dernières techniques / théories développées ces dernières années.

## 4.3. Thématique

La connaissance de la limitation de vitesse par le conducteur n'est pas quelque chose de mystique qu'il doit deviner. Tous les éléments sont présents sur la route pour l'informer de la vitesse limite à laquelle il est soumis : soit par la présence d'un panneau de limitation de vitesse, valide jusqu'au prochain panneau, soit parce qu'il sait que sur le type de route où il roule, il existe une vitesse limite imposée (limitation à 130km/h sur autoroute, 50km/h en ville,…), sauf en cas de réglementation temporaire où là encore, un panneau serait présent pour l'en informer (par exemple une zone de travaux, une zone 30 en ville,…).

L'idée naturelle qui découle de ce constat serait de proposer ce même type de raisonnement pour répondre à la problématique de la détection des limitations de vitesse : détecter la limitation par capteur visuel, et si aucun panneau n'a jamais été vu, utiliser le capteur cartographique pour déduire cette information. Cela peut sembler simple, mais c'est typiquement ce que fait l'être humain.

Or, nous avons vu (cf. chapitre 2, section 2.2) que tous les travaux de la littérature effectuant de la fusion de données multi-sources pour la détection de la limitation de vitesse se basent sur le fait que les sources (capteur cartographique et capteur visuel) sont redondantes au niveau attribut (valeur de la limitation de vitesse). Cette redondance leur permet, dans le cas où les deux sources sont en accord, de confirmer la détection de limitation de vitesse. Cependant, il n'est pas rare qu'un capteur cartographique se trompe quant à la limitation imposée (imprécision du positionnement et base de connaissance statique). Que se passe-t-il donc dans le cas où les sources sont en désaccord ? L'expertise leur permet de répondre à ce problème en accordant plus ou moins de crédibilité / croyance / probabilité à l'un ou l'autre des capteurs mais cela peut amener à une situation d'ignorance où il ne pourrait être décidé de la limitation de vitesse en vigueur. Et, encore plus problématique, que se passe-t-il si les deux capteurs fournissent la même information erronée ?

Nous disposons ici d'un capteur visuel (caméra + algorithme) de détection des panneaux de limitation de vitesse presque parfait, i.e. qui ne se trompe rarement (dans le chapitre 1 section 1.6.2, il a été relevé une seule fausse détection) et qui ne voit pas un panneau que dans de rare cas (94% de taux de bonne détection, chapitre 1 section 1.6.2). Sauf évidemment en cas d'occultation totale et durable par un autre véhicule, nous avons vu que, outre les panneaux ratés, la principale difficulté qui découle de l'utilisation de ce capteur est le problème de l'interprétation de la détection des panneaux qu'un être humain fait naturellement.

Sur ce dernier aspect, il ne faut pas envisager ici le processus de fusion comme de la fusion de niveau attribut (niveau 2) de sources redondantes qu'il faudrait départager quand elles sont en conflit via des théories de fusion bien établies dans ce domaine (inférence bayésiene, logique floue, théorie des croyances,…) mais utiliser le fait que les sources d'information disponibles sont complémentaires afin de réaliser une fusion de dernier niveau, de niveau décisionnel : décider si le panneau concerne ou non le véhicule.

Pour ce faire, la fusion par complémentarité des sources est envisagée ici via un processus à base de règles comme les Systèmes Experts. Le principe fondamental des systèmes experts est d'introduire la connaissance des experts humains dans les ordinateurs afin de les rendre « intelligents » et de permettre de faire de l'aide à la décision. L'expertise se traduit souvent par un ensemble de règles déductives qui reflètent par leur enchainement le raisonnement des experts eux-mêmes.

Les règles « expertes » mises en place pour cette fusion de données sont décrites dans la section suivante pas à pas en même temps que la réponse au scénario de test réel qui y est présenté.

## 4.4. Mise en œuvre

### 4.4.1. Scénario de test

Plutôt que de présenter de nombreux scénarios pour différents cas de figure du problème d'interprétation des panneaux de limitations de vitesse, un seulement est exposé ici. Ce scénario regroupe l'ensemble des difficultés les plus communes pouvant être rencontrées. Il s'agit d'un cas concret typiquement croisé sur le périphérique parisien (Figure 160) :

- Une voie de sortie présente avec son panonceau un panneau de limitation de vitesse à 50km/h.

- Après cette voie de sortie est rappelée, via un panneau, la limitation de vitesse réglementaire sur le périphérique : 80 km/h.

Ce cas est, pour rappel, le même que celui décrit par la Figure 127 à propos de l'impossibilité d'utiliser seul un capteur cartographique, dû à l'imprécision, pour y déterminer la limitation de vitesse d'une voiture se déplaçant sur cette route.

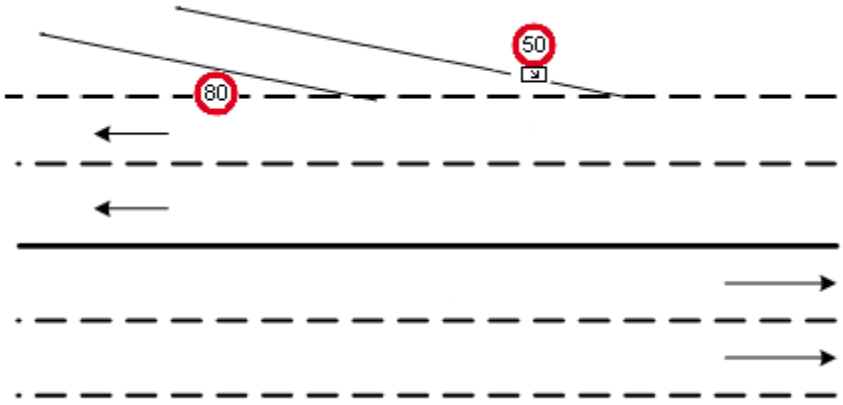



Figure 160 - Scénario de test

Les difficultés de ce cas typique sont multiples :

- Le panneau de limitation de sortie à 50km/h ne doit pas être pris en compte par le système si le véhicule ne quitte pas le périphérique. Il est clair que cette difficulté regroupe aussi l'ensemble des problèmes d'interprétation de sortie d'autoroute, ainsi que celui de la reconnaissance des panonceaux.

- Le panneau de limitation de 80km/h ne doit pas être pris en compte si le véhicule a emprunté la voie de sortie. Cette difficulté n'arrive que très rarement, mais ne doit pas pour autant être négligée afin de réaliser un système fiable.

Paradoxalement, comme nous le verrons dans la suite de ce chapitre, d'autres difficultés surgiront du fait de l'utilisation des différents capteurs dans le processus de fusion de données mis en place, fusion qui doit permettre de trancher quant à la prise en compte de la détection d'un panneau ou non.

Finalement ce scénario est décomposé en 3 cas de figure présentés ci-après dans lesquels seront établie une à une les règles qui régiront le processus de fusion de nos 3 capteurs :

- Caméra + détection des panneaux et panonceaux de limitation de vitesse
- Caméra + détection des lignes de marquage au sol
- GPS + Base de données cartographiques

### 4.4.2. Règles de fusion

#### *a) Cas 0 : Le véhicule n'a pas vu de panneau*

Il peut arriver dans de nombreux cas, que le véhicule n'ait pas croisé de panneau de limitation de vitesse. Cela se produit par exemple quand le conducteur démarre sa voiture (d'un parking, du bas coté d'une route ou de la voie d'arrêt d'urgence d'une autoroute) ou si il se déplace en ville (lieu où la réglementation générale imposant une limite de 50km/h rend les panneaux obsolètes sauf en cas de zone 30).

Dans ces cas, la seule information de limitation disponible est celle provenant du capteur cartographique. Or, comme il a déjà été stipulé précédemment, ce capteur souffre d'une *grande* imprécision quant au positionnement GPS et peut se tromper de route avec une adjacente. Mais est-ce ici vraiment un problème ?

En effet, la voiture ne se « réveille » pas au milieu de nulle part, elle est généralement au même endroit que là où elle s'était arrêtée. Pour s'en assurer, il suffit d'enregistrer la dernière position GPS du véhicule avant son arrêt et la nouvelle position GPS à son redémarrage. Une mémoire permet d'affecter la limitation de vitesse à la dernière valeur enregistrée (i.e. panneau vu par le véhicule).

Néanmoins, la question peut se poser dans de rares cas où il serait nécessaire de retrouver la limitation de vitesse via le capteur cartographique. Rappelons-le, un capteur cartographique ne permet pas de connaître avec *précision* la limitation de vitesse, son défaut étant notamment lors des intersections où il pourrait se tromper quant à savoir sur quelle route le véhicule se trouve (les 2 routes pouvant ne pas avoir la même limitation de vitesse).

Un capteur cartographique connaît soit la limitation réglementaire du segment sur lequel évolue la voiture, soit connaît le type d'environnement où il se situe (zone urbaine, autoroute,…). Si l'environnement est constant en termes de limitation de vitesse (i.e. si toutes les routes dans un faible périmètre ont la même limitation) alors il ne peut y avoir de confusion.

Le seul cas ambigu reste alors celui où le véhicule est déplacé alors qu'il est éteint (enlèvement par la fourrière par exemple) et qu'il est stationné au niveau d'un carrefour. Cela n'arrive pour ainsi dire jamais.

Concrètement, prenons l'exemple de la Figure 161 de notre scénario. Le véhicule se trouve en amont du premier panneau, celui de la voie de sortie. Supposons qu'il n'ait pas encore croisé de panneau de limitation de vitesse. Le cercle bleu schématise l'imprécision de positionnement du GPS qui peut donc seulement ici se tromper de voie. Le cône rouge schématise quant à lui la vue du véhicule. L'obtention de la limitation de vitesse par le capteur cartographique n'est ici aucunement ambiguë et fournira la vitesse escomptée de 80km/h.

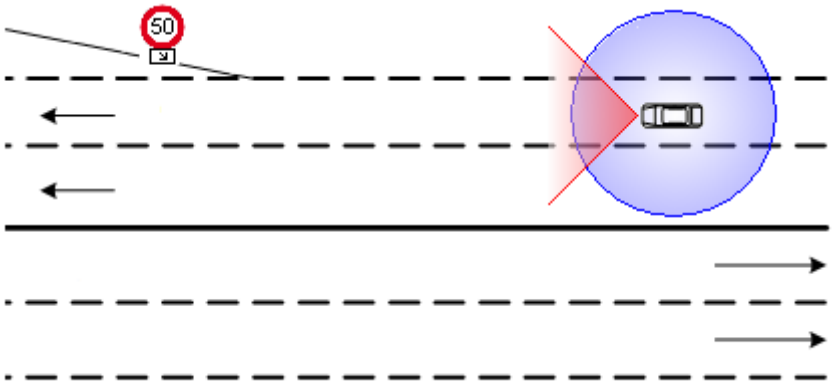



Figure 161 - Scénario de test, cas où le véhicule n'a pas encore croisé de panneau de limitation de vitesse

Dans le cas où il y aurait un pont à proximité avec une zone urbaine au dessus, un peu de logique dans le processus de map-matching résoudrait se léger problème bien connu et bien maitrisé des capteurs cartographiques.

Le seul cas problématique en termes de détermination de la limitation de vitesse est en fait celui des embranchements (carrefour, voie de sortie,…) qui est débattu dans les 2 prochaines sous sections.

### b) *Cas 1 : Le véhicule n'emprunte pas la voie de sortie*

Dans ce premier cas de figure, le véhicule roule et n'emprunte pas la voie de sortie. Il détecte donc successivement les panneaux de limitation de 50km/h avec son panonceau de sortie et le panneau de limitation de 80km/h.

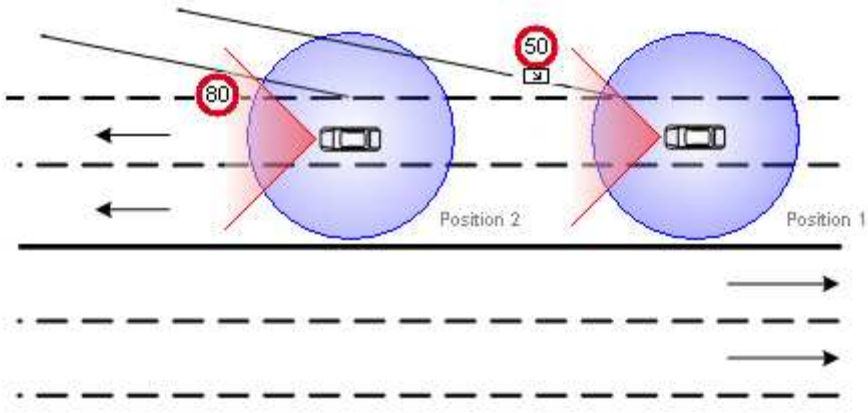



Figure 162 - Scénario de test, cas où le véhicule n'emprunte pas la voie de sortie

On voit ici que, du fait de l'imprécision de positionnement, l'information du GPS peut aussi bien fournir une limitation de 50km/h que de 80 à tout moment aux abords de la voie de sortie. Ainsi quand le véhicule a vu la limitation de 50km/h et si le GPS donne aussi cette information, que se passerait-il avec un processus de fusion classique (Dempster-Shafer). Serait-on dans un cas d'incertitude totale quand à la limitation en cours ? Ou est-ce que le système donnerait une solution, au hasard, de 50 ou de 80 km/h ? Dans les 2 cas, un tel système est à proscrire dans un système ADAS qui est là pour aider le conducteur et donc auquel le conducteur doit pouvoir se fier.

Le problème est en fait ici d'interpréter la détection du premier panneau, qui ne doit pas être pris en compte. Dans le chapitre 3 a été présenté l'algorithme de détection des lignes de marquage au sol. L'utilisation de cette détection permet de savoir quand le véhicule change de voie : l'une des lignes, au lieu de rester sur l'un des cotés de l'image, la traversera par un mouvement de translation latérale, elle aura changé de côté dans l'image.

Ainsi, si aucun changement de voie n'est détecté, alors le panneau de limitation à 50km/h affublé d'un panonceau de sortie, ne doit pas être pris en compte.

Dans notre scénario, et dans ce cas, le panneau suivant de limitation à 80km/h est classiquement pris en compte.

### c) *Cas 2 : Le véhicule emprunte la voie de sortie*

Dans ce second cas de figure, le véhicule roule et emprunte la voie de sortie. Il détecte successivement les panneaux de limitation de 50km/h avec son panonceau de sortie mais aussi celui de limitation de 80km/h qui est proche de la voie de sortie.

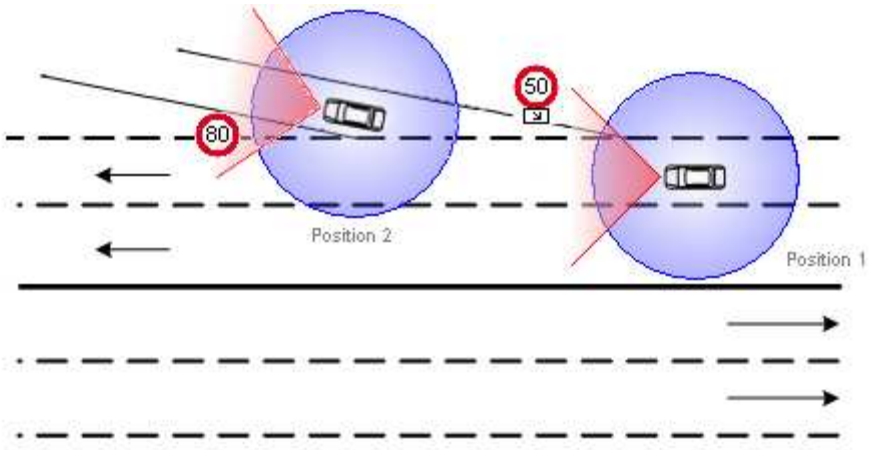



Figure 163 - Scénario de test, cas où le véhicule emprunte la voie de sortie

Le problème est ici double, il faut tout d'abord interpréter l'environnement routier afin de déterminer que le panneau de limitation à 50km/h est bien en vigueur pour le véhicule ; mais aussi et surtout, de comprendre que le panneau de limitation à 80km/h n'est pas une limitation pour la voie de sortie.

Le premier problème se traite facilement, avec la même méthode que dans le cas précédent : détecter le changement de voie. Ainsi, si un changement de voie est détecté, alors il faut tenir compte de cette limitation de vitesse.

On voit bien ici que pour le second problème, qu'un processus de fusion classique n'est pas du tout adapté. En effet, la caméra détecte le panneau de limitation à 80km/h. Le capteur cartographique a aussi de grandes chances de se tromper et fournir une vitesse de 80km/h. Ainsi, dans ce cas de figure, les 2 capteurs peuvent fournir des informations erronées (par manque d'interprétation des détections) et la redondance de ces 2 informations inexactes fusionnées entraine inévitablement une détection globale de 80km/h encore plus « certaine ». Dans ce cas de figure, un système basé sur une telle fusion est encore plus à proscrire dans un système ADAS que précédemment, le système indiquant au conducteur qu'il peut accélérer au lieu de freiner (ce qui mettrait en danger à la fois sa vie et celle des autres).

Néanmoins, ce second problème n'est pas beaucoup plus compliqué à résoudre que le premier si l'on reste dans une optique d'imiter le raisonnement humain. Le véhicule a emprunté une voie de sortie, sa vitesse limite ne peut alors que décroitre jusqu'au croisement d'une autre intersection (embranchement d'autoroute,…). Ainsi le panneau de limitation à 80km/h est une limite plus importante que celle de 50 à laquelle est déjà soumis le véhicule, alors il n'est pas pris en compte. Il ne reste qu'à utiliser le capteur cartographique en analysant l'horizon électronique afin de détecter l'embranchement avec une autre route et calculer la distance à la prochaine intersection.

## 4.5. Expérimentation et résultats

### *a) Prototype*

L'expérimentation issue des 2 cas du scénario de test s'est effectuée avec le prototype (véhicule + logiciel) décrit précédemment au chapitre 1 : une voiture Citroën C3 équipée entre autre d'une caméra à l'avant (derrière le rétroviseur central), d'un GPS, et de capteurs proprioceptifs (vitesse,…) et embarquant un PC à l'arrière du véhicule sur lequel est exécuté le logiciel $^{rt}$Maps se chargeant d'enregistrer de façon synchronisée les données issues de ces différents capteurs.

Les données ont ensuite pu être rejouées en temps réel et les scénarios validés. Ci-dessous, Figure 164, est montré à titre informatif, le diagramme $^{rt}$Maps de prototypage des algorithmes de détection et de fusion de données mis en œuvre dans ce mémoire.

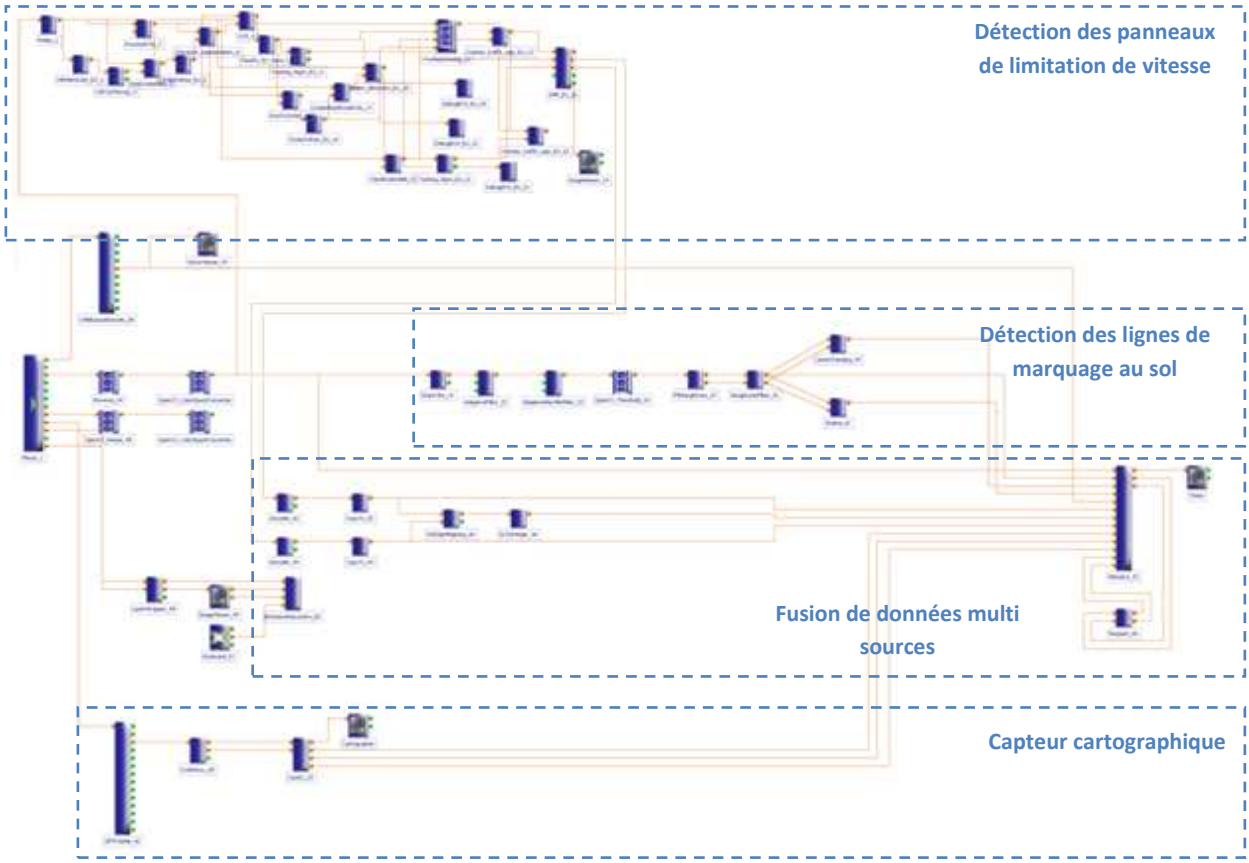



Figure 164 - Diagramme rtMaps de détection et d'interprétation des panneaux de limitation de vitesse

### b) *Cas 1 : Le véhicule n'emprunte pas la voie de sortie*

Quelques commentaires préliminaires s'imposent sur l'affichage des résultats avant même de décrire les différentes étapes du scénario. La fenêtre de gauche représente les données cartographiques. Celle de droite la vue de la caméra avec les différents résultats. Au centre de la vidéo est affichée la limitation de vitesse validée par le processus de fusion. En bas est indiqué par une flèche si le véhicule change de voie (flèche horizontale vers la gauche ou la droite) ou pas (flèche verticale). Dessous est indiqué le nom de la route (information cartographique). Les lignes de marquage au sol détectées sont affichées en rouge. En haut à gauche sont affichées les limitations de vitesse détectées par les 2 capteurs cartographique et visuel. En haut à droite est affichée la distance au prochain embranchement (information cartographique). Au centre, sur la limitation validée, est affichée la vitesse actuelle du véhicule.

Le système est tout d'abord initialisé alors que le véhicule roule déjà sur le périphérique. A cet instant, le récepteur GPS n'a pas encore eu le temps de se synchroniser avec les satellites et ne reçoit donc aucune position. Le capteur cartographique ne fournit donc aucune donnée. Le capteur visuel n'a lui aussi encore détecté aucun panneau. Le système ne peut donc déterminer la limitation de vitesse en cours (Figure 165). Un processus de mémoire comme décrit précédemment aurait pu permettre de retrouver la limitation de vitesse, mais il n'est pas mis en place ici.

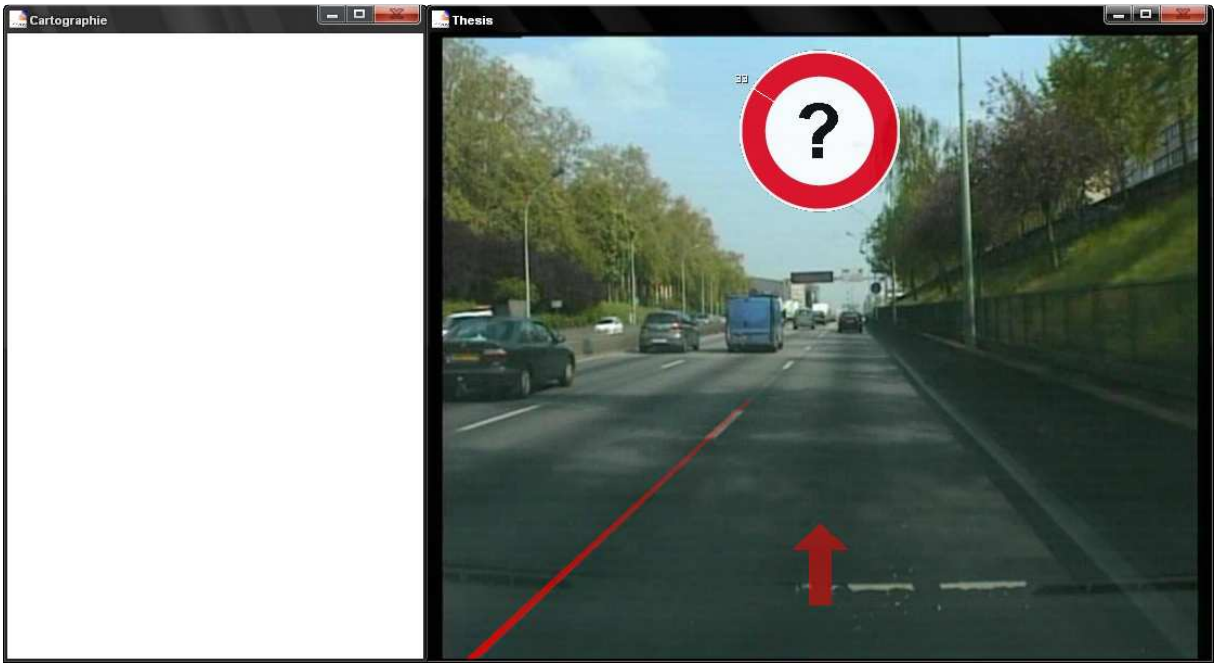



Figure 165 - Aucune information de limitation de vitesse n'est détectée (vision ou cartographie)

Au bout de quelques instants (Figure 166), le récepteur GPS reçoit les informations des satellites et fournit une position au système. Le capteur cartographique peut ainsi retrouver les informations dans sa base, afficher le graphe routier courant, et fournir les informations nécessaires au processus de fusion : la limitation actuelle de 80km/h. Sans autre information de limitation visuelle et n'étant pas proche d'une route connexe à autre limitation (cas qui serait ambigu), le processus de fusion valide cette limitation.

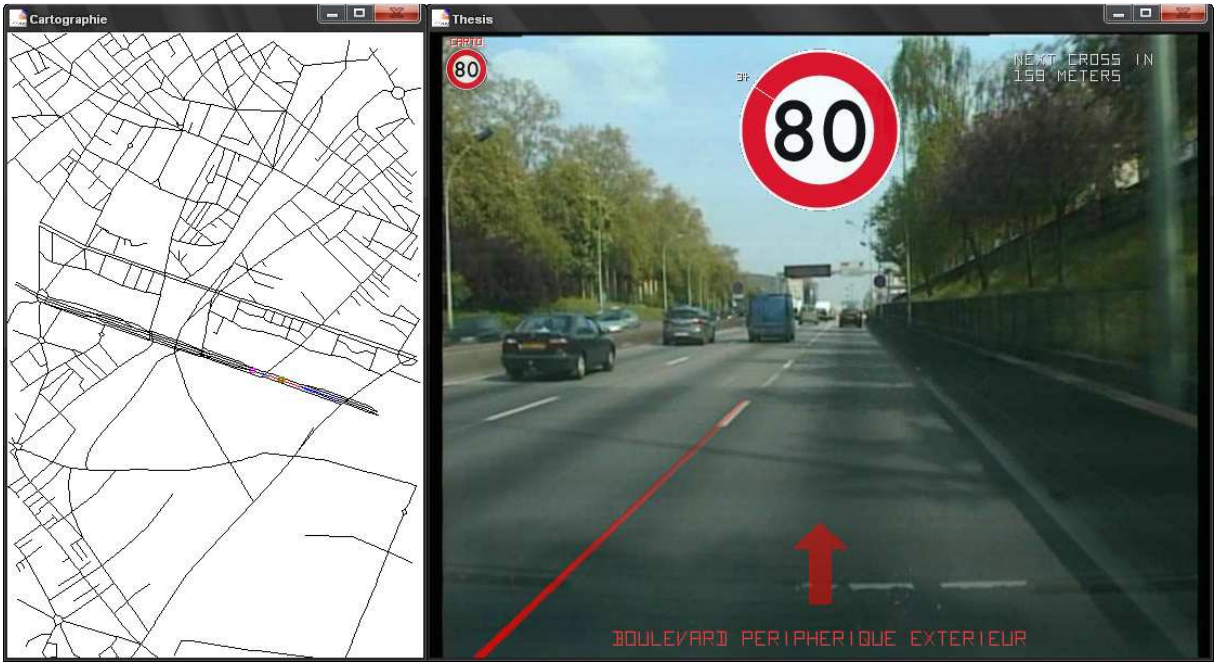



Figure 166 - Réception des données cartographiques : validation de la limitation cartographique à 80km/h

A l'approche de la voie de sortie, le capteur visuel détecte le panneau de limitation de vitesse ainsi que son panonceau associé stipulant qu'il s'agit d'une limitation pour voie de sortie (Figure 167). A l'entrée du processus de fusion réside donc la détection de 80km/h par le capteur cartographique et de 50km/h par la vision. Le système de fusion ne valide, pour l'instant, pas cette détection visuelle. Il attend un éventuel changement de voie pour cela.

A la fin de cette intersection, le panneau de limitation de 80km/h est détecté visuellement (Figure 168). Le système n'ayant pas détecté de changement voie, n'a pas validé la précédente détection à 50km/h. Ce panneau est validé et renforce la *croyance* du système dans la limitation actuelle à 80km/h.

Enfin, Figure 169, bien qu'il n'y ait plus aucun panneau, la dernière limitation validée reste active jusqu'à la prochaine détection qu'il faudra interpréter.

Il faut noter que dans ce cas (la voiture qui n'emprunte pas la voie de sortie), l'information cartographique n'est que très peu utilisée pour l'interprétation des résultats visuels. Il ne sert qu'à combler la première lacune de détection. Seule la détection des lignes de marquage au sol intervient dans le processus de fusion / désambigüisation.

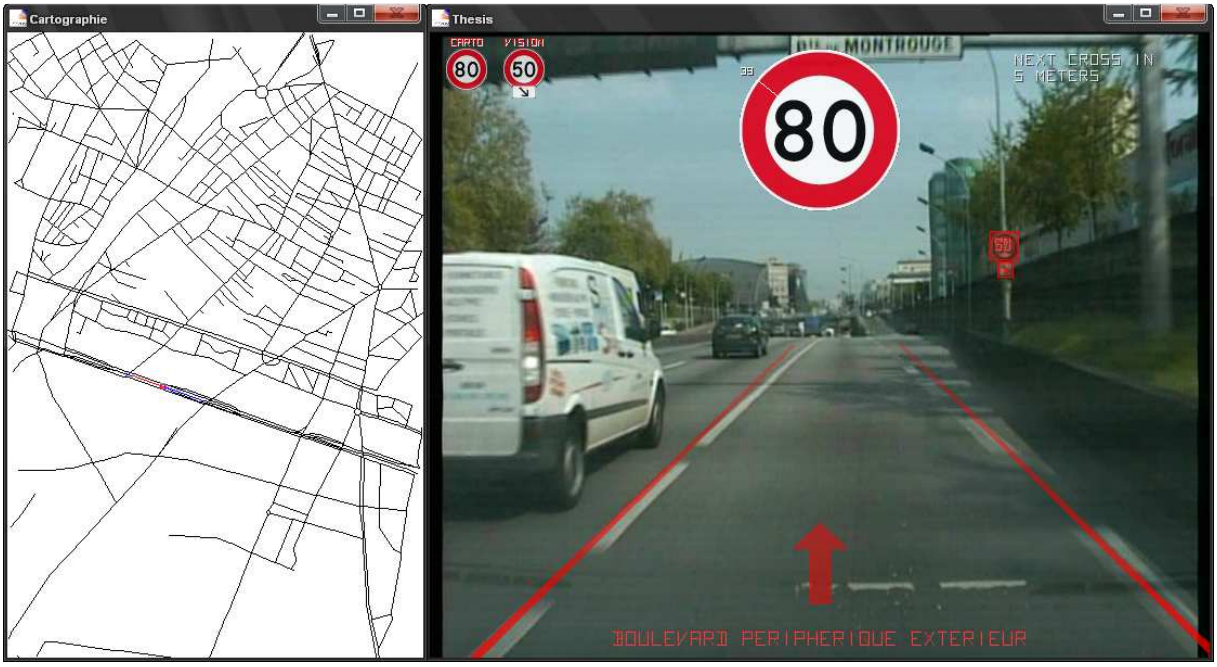



Figure 167 – Détection visuelle du panneau de limitation de vitesse de voie de sortie, qui n'est pas validé

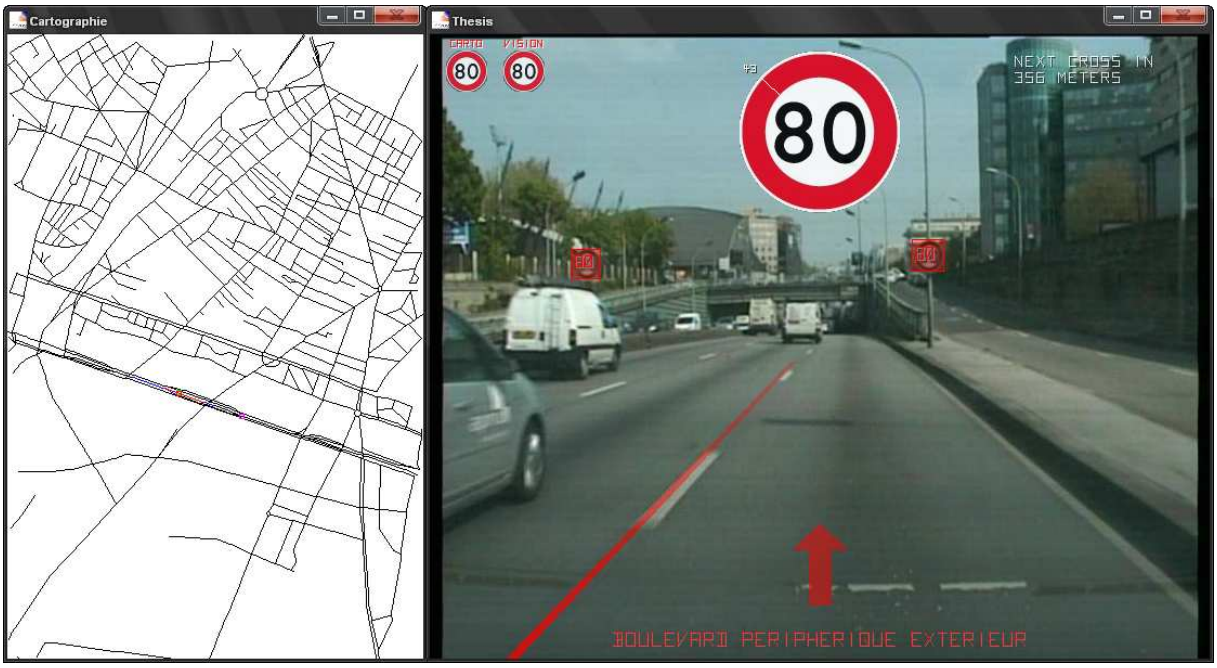



Figure 168 - Détection et validation du panneau de limitation à 80km/h

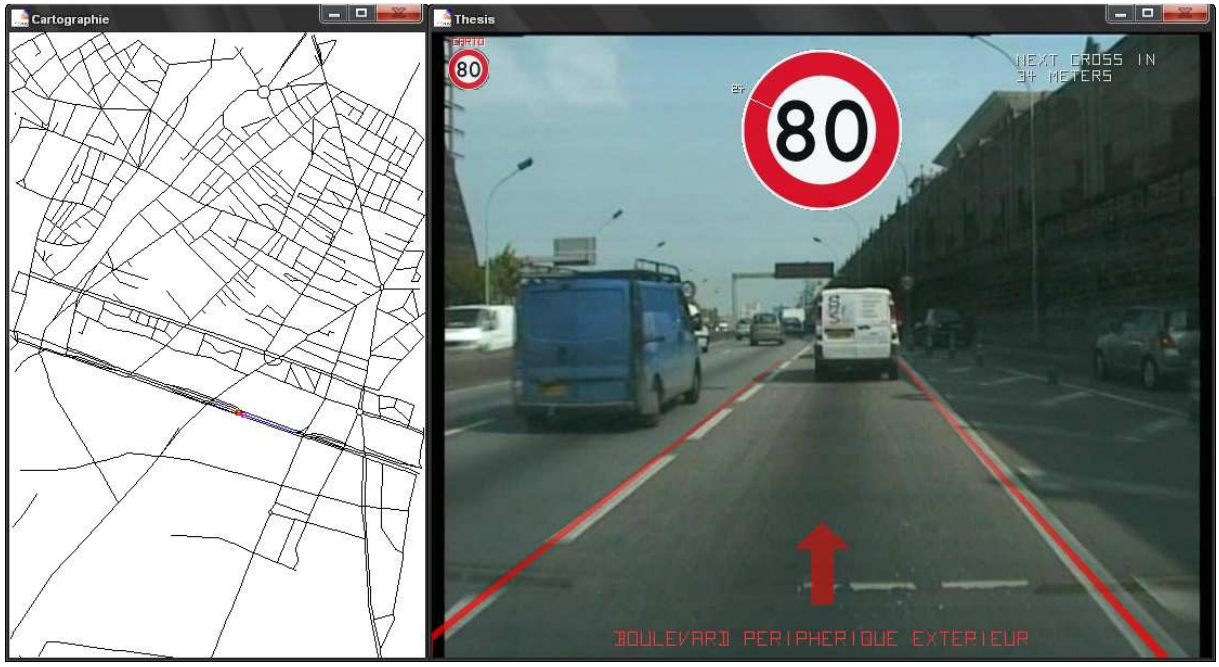



Figure 169 - Le dernier panneau validé reste actif

### c) *Cas 2 : Le véhicule emprunte la voie de sortie*

Ce cas de figure débute comme le précédent : le système commence par ne pas savoir déterminer la vitesse limite en l'absence d'information des deux capteurs visuel et cartographique (Figure 165), puis valide la détection du premier panneau à 80km/h (Figure 166) et attend pour valider ou non la détection du panneau de 50km/h de sortie de route (Figure 167).

La détection visuelle des lignes de marquage au sol permet de savoir si le véhicule a franchi l'une de ces lignes et donc de savoir si il change de voie (et dans quelle direction). Il est ainsi détecté ici et validé ici le changement de voie par le véhicule et la limitation à 50km/h est donc retenue (Figure 170).

Comme précisé plus haut, l'une des règles du système de fusion veut que cette détection de limitation pour voie de sortie ne soit valide que jusqu'à la prochaine intersection. Or, il est clair que le capteur cartographique se trompe ici de voie dans son processus de map-matching (ce qui est l'une des limitations majeurs du positionnement GPS) : la limitation fournie par le GPS reste, et ce même quand le véhicule est sur la voie de sortie, à 80km/h (Figure 171).

Afin de calculer la bonne distance, l'algorithme de fusion émet des requêtes à la base cartographique afin de retrouver la voie de sortie connectée au segment map-matché actuel. Il est alors possible de calculer la nouvelle distance au prochain carrefour à partir d'une projection de la position GPS sur le segment de la voie de sortie (comparer les deux distances calculées des Figure 168 et Figure 172 où le véhicule se retrouve à même hauteur du panneau 80, mais pas sur la même voie).

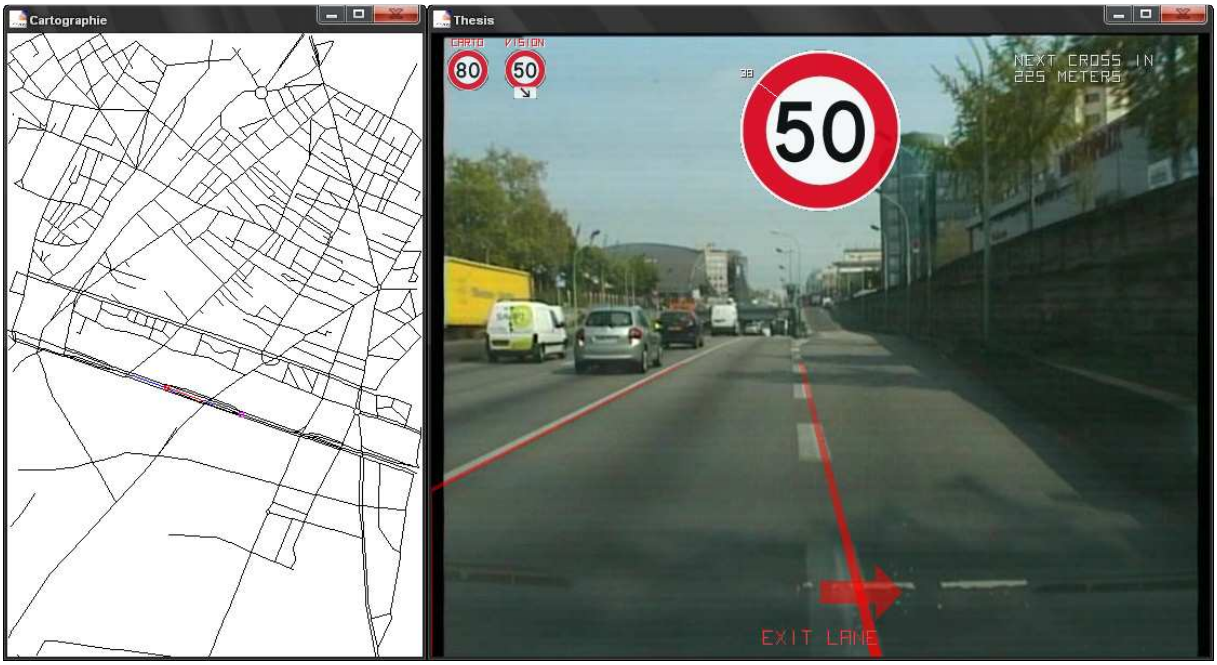



Figure 170 - Le véhicule change de voie pour emprunter la rampe de sortie, la limitation de 50km/h est validée

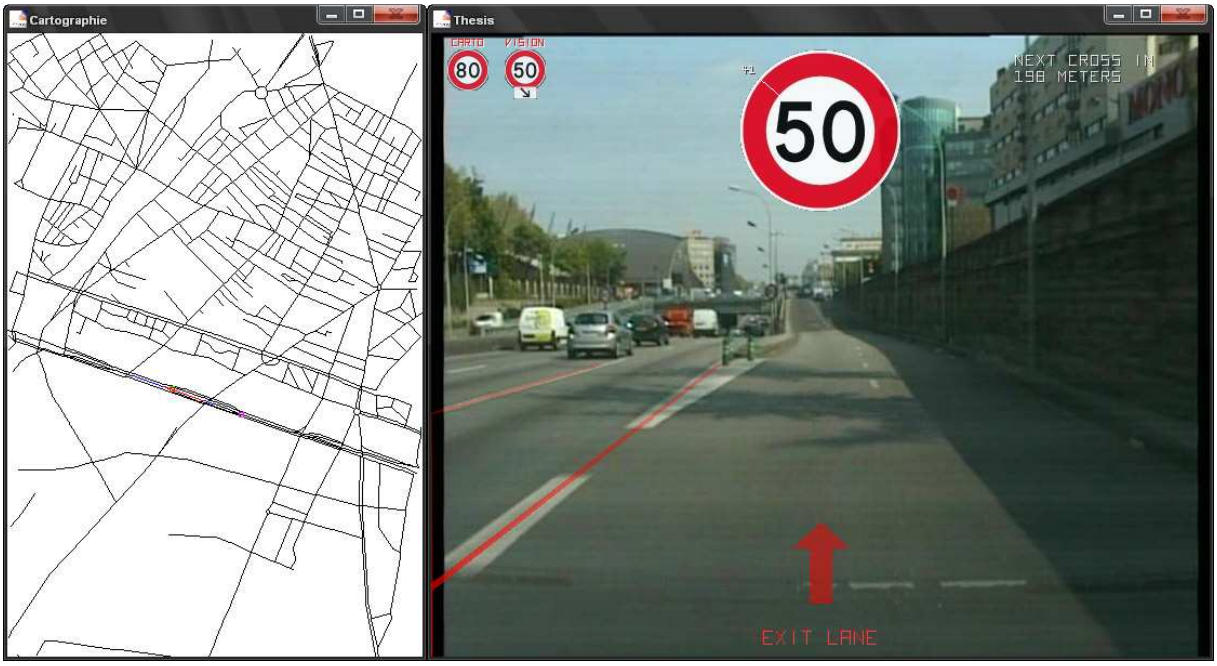



Figure 171 - Imprécision du GPS, le capteur cartographique détecte toujours une limitation à 80km/h

Enfin, le panneau de limitation à 80km/h est détecté. Celui-ci n'est pas validé (Figure 172) car le système de fusion sait que le véhicule se trouve sur une voie de sortie et que la limitation ne peut que décroitre par rapport à la première limitation validée (celle du panneau de 50km/h).

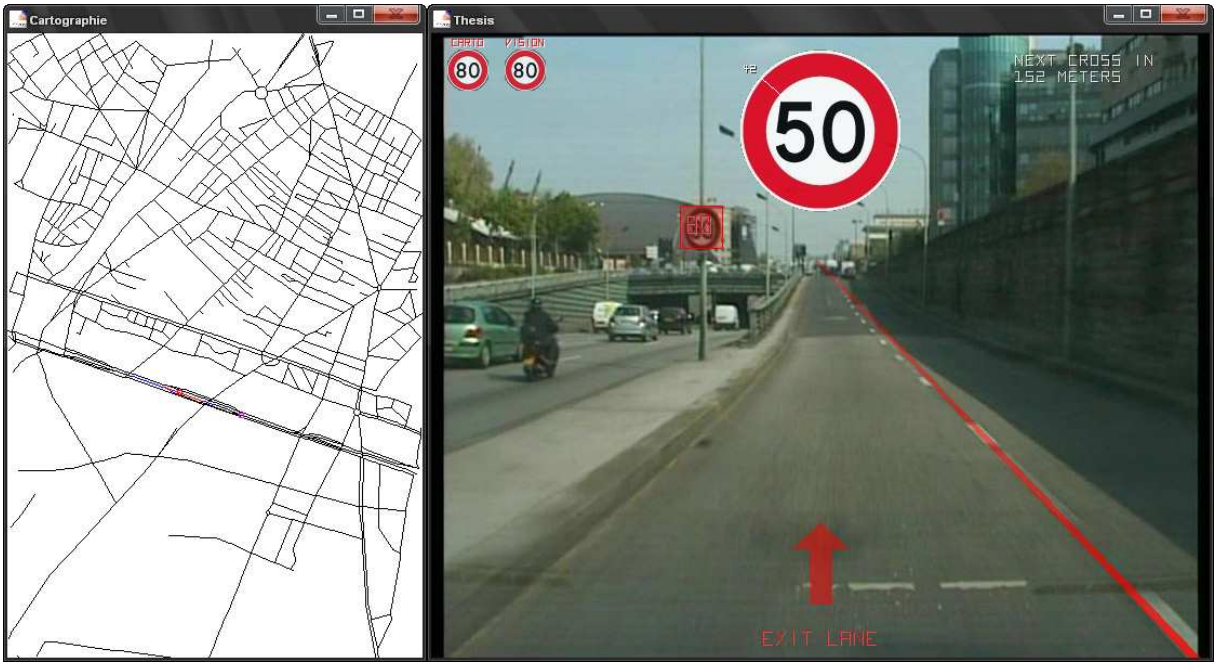



**Figure 172 - La limitation à 80km/h n'est pas validée car elle ne concerne pas la voie de sortie ; bien remarquer ici que les 2 capteurs cartographique et visuel concordent pour fournir la valeur erronée de 80 km/h.**

Il faut noter que dans ce cas (la voiture qui emprunte la voie de sortie), toutes les informations disponibles sont utilisées et sont nécessaires pour le processus de fusion servant à interpréter les résultats de la détection visuelle.

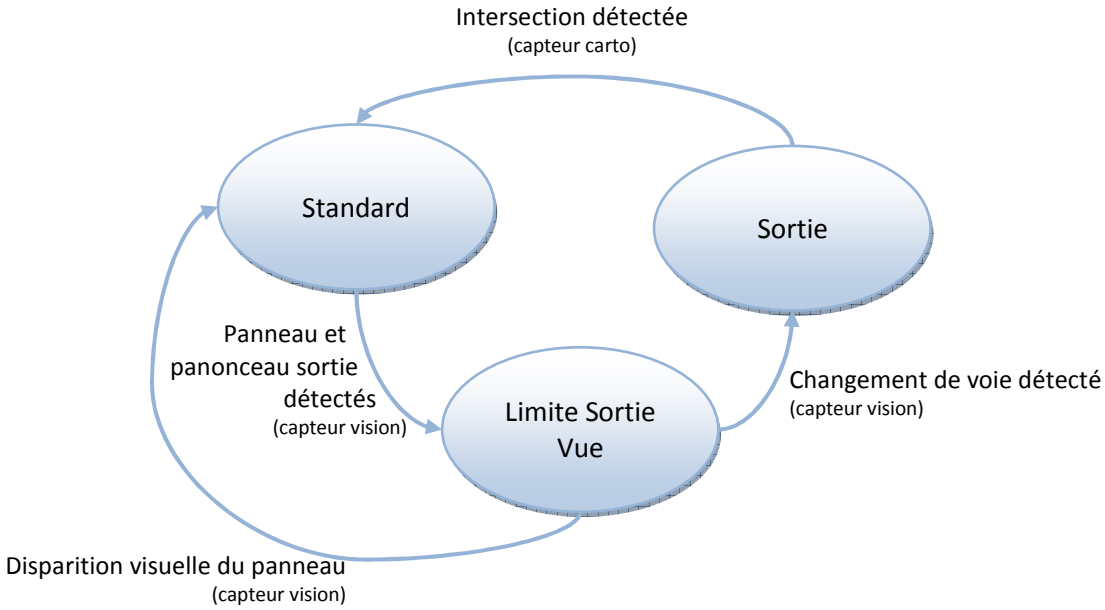
## 4.6. Généralisation

Nous avons vu que les règles mises en place jusqu'ici couvrent l'un des deux problèmes d'un système de détection visuelle : le manque d'interprétation des panneaux détectés.

Le second problème, celui des panneaux occultés (i.e. quand le système de détection visuel ne voit pas un panneau), ne trouve quant à lui pas sa réponse dans le système actuellement décrit. Il faut dans ce dernier cas donner la priorité à l'information de limitation de vitesse issue de la cartographie.

Afin de savoir quand donner la priorité à la cartographie, une règle simple peut être mise en place consistant à évaluer la distance parcourue par le véhicule depuis la dernière détection visuelle. Si

cette distance est supérieure à un seuil donné, alors il peut être envisagé que le système ait potentiellement raté un panneau de limitation de vitesse et retrouver cette information à partir de la base cartographique.




**Règles par état :**

| | |
|---|---|
| Règle « Standard » | Valider les limitations visuelles sans panonceau |
| Règle « Limite Sortie Vue » | Valider les limitations visuelles sans panonceau |
| Règle « Sortie » | Valider les limitations visuelles de valeur inférieure |

**Règle générale :**
SI la limitation validée est trop ancienne ET que la limitation cartographique n'est pas ambiguë ALORS valider la limitation cartographique.

**Figure 173 – Schéma d'états des règles de fusion mises en place**

Voici pour résumer le schéma fonctionnel du processus de fusion à base de règles mis en place et décrit précédemment (Figure 173). Le système de fusion comporte trois états régissant les règles de validation des limitations de vitesses détectées par les différents capteurs. Une règle générale ne dépendant pas de l'état dans lequel se trouve le système de fusion est aussi présente répondant aux problèmes des panneaux de limitation non vus par le capteur visuel. Le système passe d'un état d'exécution à un autre selon les diverses détections des différents capteurs et ne peut en aucun cas se retrouver dans deux états différents au même moment.

## 4.7. Discussions et perspectives

Le processus de fusion mis en place ici va à contre-pied des systèmes proposés dans la littérature pour la détection des limitations de vitesse se basant sur les mêmes capteurs que nous utilisons (caméra + GPS).

Ces derniers se basent tous sur des modèles probabilistes mathématiques (théorie de Dempster-Shafer) afin de déterminer la valeur de la limitation de vitesse à partir des 2 valeurs (ou plus) fournies par ces 2 capteurs. Ce principe de fusion est plus communément appelé dans la littérature « Feature fusion » soit de la fusion d'attributs, une fusion de « niveau 2 » utilisant la redondance de l'information issue des différents capteurs dans le principe que fusionner une information redondante permet d'améliorer sa croyance en celle-ci. L'avantage de cette technique est de donner des indicateurs (indice) de confiance dans la limitation validée (i.e. fusionnée).

Ici est proposé et validé un modèle de fusion de plus haut niveau, basé sur un système à base de règles. L'enjeu majeur de ce processus, n'est pas d'utiliser la redondance des capteurs, mais leur complémentarité afin d'interpréter les informations issues de ces 2 capteurs et de déterminer si une détection est valide ou pas.

L'exemple mis en exergue à la Figure 172 est criant. Les deux capteurs détectent la même mauvaise information (limitation de vitesse). Le processus de plus haut niveau arrive sans difficulté à ne pas se tromper, alors que le processus classique utilisé dans la littérature ne pourrait que tomber en erreur (fusionner 2 fois la même information renforce la croyance en cette information).

Ainsi, comme le disait (Adjaoute, 1988) à propos des systèmes à base de règles (i.e. les systèmes experts) comparés aux approches probabilistes mathématiques :

> « Les méthodes probabilistes classiques ne peuvent disposer ni de stratégies, ni de raisonnements dynamiques, ni de la capacité de justifier leurs résultats, ni même de connaissances proprement dites. L'approche des systèmes experts est totalement différente de l'approche probabiliste. L'essence même des systèmes experts consiste à faire une séparation absolue entre le raisonnement et la connaissance. Le type de raisonnement utilisé dans les systèmes expert est un raisonnement qui cherche seulement les informations permettant des conclusions formelles. »

Cependant, bien que ce processus de décision / fusion mis en place ici soit assez efficace, il manque encore de robustesse sur le fait de savoir si le véhicule a vraiment emprunté la voie de sortie ou a simplement changé de voie se rapprochant du bord de la route.

Pour pallier ce problème, il faudrait détecter la présence d'une autre voie adjacente. Dans ce cas, la voie de sortie étant toujours la dernière, le véhicule ne pourrait l'avoir emprunté. Cette détection peut se faire en comptant le nombre de lignes détectées à gauche ou à droite de l'image (Figure 171, 2 lignes sont détectés à gauche, le véhicule n'est pas sur la voie rapide de gauche). Cependant, les lignes des autres voies sont souvent occultées par les autres véhicules (Figure 167), il est alors envisageable de détecter les autres véhicules afin de déterminer la présence ou non d'autres voies

sur les cotés. Enfin un processus de mémoire pourrait améliorer cette connaissance (de la position du véhicule sur les voies) en tenant compte des anciennes détections et des changements de voie.

Par ailleurs, il faudrait s'assurer que le jeu de règles mis en place sur le cas prototypique présenté est suffisant pour couvrir tous les cas de figure pouvant se présenter.

# Outils développés

## 5.1. Motivations

Au centre de robotique, nous utilisons pour nos recherches en aide avancée à la conduite automobile (ADAS), le logiciel RTMaps qui nous permet d'enregistrer et de rejouer de façon synchroniser des séquences d'acquisition des données des capteurs de nos véhicules (cf. section 3.6.1 de ce document). Ce logiciel, initialement développé au centre à partir d'un besoin d'un outils d'acquisition des données (data logging) et de traitement dans des conditions proches de l'embarqué, nous facilite grandement la tâche dans le fait de pouvoir prototyper nos algorithmes sur nos ordinateurs de bureau à partir des séquences acquises comme si il s'agissait de tests en temps réel lors d'essai en véhicule, puis d'une simple formalité, de les porter sur nos véhicules en copiant les fichiers générés par ce logiciel sur les ordinateurs embarqués.

Les différents modules mis en œuvre dans les systèmes ADAS, et notamment ceux du laboratoire, se basent généralement sur des algorithmes de détections (détection des lignes de marquage au sol, détection des véhicules, détection des panneaux signalétiques, détection des feux tricolores, détection des piétons,…). Ces détections se font par traitement du signal du capteur soit par des algorithmes très spécifiques au problème (ex : détection des cercles dans les images provenant d'une caméra) soit par l'utilisation d'algorithmes du domaine de l'intelligence artificielle dits d'apprentissage numérique (ex : les réseaux de neurone).

L'utilisation répétitive de ces algorithmes au cours de travaux de recherche, et l'exploitation de nombreuses séquences d'acquisitions avec le logiciel RTMaps, amène à se poser plusieurs questions :

- Comment évaluer efficacement nos divers algorithmes de détection mis en place ?
- Comment améliorer ces algorithmes d'apprentissage numérique par l'apport de nouvelles idées ?
- Comment organiser ces giga octets de données issues des différents scénarios d'acquisitions des séquences de tests en véhicule ?

Ces questions que je me suis posées durant mes travaux de thèse m'ont amené à développer et à proposer plusieurs outils présentés dans les trois prochain paragraphes.

Cette présentation pourrait plus ressembler, pour certains outils, aux yeux du lecteur à une description de type ingénierie (au contraire d'un travail de recherche), mais elle a bien sa place dans ce manuscrit de thèse car ces outils développés constituent un vrai travail de réflexion sur des problèmes de recherche, auxquels tout chercheur a dû être confronté à un moment donné. En effet, toutes les questions évoquées ici, trouvent leurs réponses dans chacun de ces logiciels qui

correspondent à un vrai besoin pour l'évaluation et l'analyse des paramètres des algorithmes développés au cours d'un travail de recherche.

## 5.2. Outils d'édition de vérité terrain

### 5.2.1. Présentation générale

Une autre question soulevée au cours de mes travaux de thèse était de savoir comment évaluer efficacement un algorithme de détection, et notamment celui des panneaux de limitation de vitesse mis en place dans les travaux présentés dans ce mémoire.

Il est certes possible d'appliquer l'algorithme développé sur une longue séquence (scénario) de test et de relever manuellement les résultats (i.e. le nombre de bonnes et de non détections) mais cela s'avère très fastidieux et peu précis. En effet, au cours du travail de recherche et de développement d'une méthode de détection (ou autre), les algorithmes mis en place évoluent forcément soit par le changement des jeux de paramètres utilisés afin d'affiner au mieux les résultats, soit tout simplement parce que de nouvelles méthodes sont implémentées. Idéalement, après chaque tentative d'amélioration d'un processus, il faudrait évaluer l'apport des changements afin d'être sûr que ces derniers améliorent bien les résultats. Or il est clair que pour évaluer systématiquement son algorithme après chaque modification sur un scénario de test, même court d'une dizaine de minutes, il est nécessaire d'automatiser ces tests.

Il existe de nombreux logiciels d'annotation vidéo disponibles. Le plus connu et le plus utilisé d'entre eux est sans doute le logiciel Viper - Video Performance Evaluation Resource (Doermann, et al., 2000). Mais on peut aussi citer les logiciels Anvil - Video Annotation Research Tool (Kipp, 2001), ODViS - Open Development Environment for Evaluation of Video Surveillance Systems (Jaynes, et al., 2002),… Une liste non exhaustive mais assez complète peut être trouvée dans (Travis Rose, 2007).

Dans ces travaux, (Travis Rose, 2007) relève de nombreux points sur l'impossibilité d'utiliser un outil existant pour l'annotation de ces séquences de travail, dont entre autres :

- Les outils existants ne fournissent pas toutes les fonctionnalités voulues.
- Il n'y a pas de support pour des formats vidéos non standards.
- Ils ne supportent pas l'annotation de vidéo de plus de 10 min.
- Ils ne supportent pas l'annotation de vidéo issues de plusieurs caméras
- Ils ne supportent pas des synchronisations précises entre de multiples vidéos

Bien que certains de ces points soient discutables, il est aisé de relever un certains nombre de ces points dans le cas des séquences $^{rt}$Maps : le support des vidéos $^{rt}$Maps ne sera jamais pris en compte par les autres logiciels d'annotation vidéo, ils ne supportent pas les synchronisations précises entre de multiples sources, et aussi, ils ne fourniront pas l'ensemble des fonctionnalités voulues.

De ce constat est né SamGT (GT étant l'acronyme de Ground Truth) : un logiciel capable de lire et d'annoter des séquences $^{rt}$Maps afin de réaliser des fichiers dits de vérité terrain (*ground truth* en

anglais, fichiers contenant la position des objets à détecter pour chacune des images d'une vidéo), couplés à un package rtMaps (un ensemble de modules) pouvant comparer automatiquement les résultats d'un algorithme par rapport au fichier de vérité terrain préalablement édité par un opérateur humain. Tout l'intérêt et l'originalité résident dans l'utilisation du package rtMaps, qui, comme son nom l'indique, fonctionne nativement sous notre plateforme de prototypage, l'algorithme mis en place n'aura donc pas besoin d'être adapté (i.e. aucun besoin de traduire les résultats dans un format donné) afin de pouvoir être évalué avec SamGT.

## 5.2.2. L'édition de fichier de vérité terrain

### *a) L'éditeur*

La conception de l'éditeur de fichier de vérité terrain du logiciel SamGT est, à l'instar de la conception d'un algorithme sous rtMaps, réalisé de façon modulaire. Ainsi comme le résume la Figure 174, cet éditeur est conçu de façon à n'être en fait qu'un « moteur » récupérant les images frame par frame dans un flux vidéo et appliquant une série de plugins (i.e. d'algorithmes) sur une image (qui sera donc modifiée) avant de l'afficher.

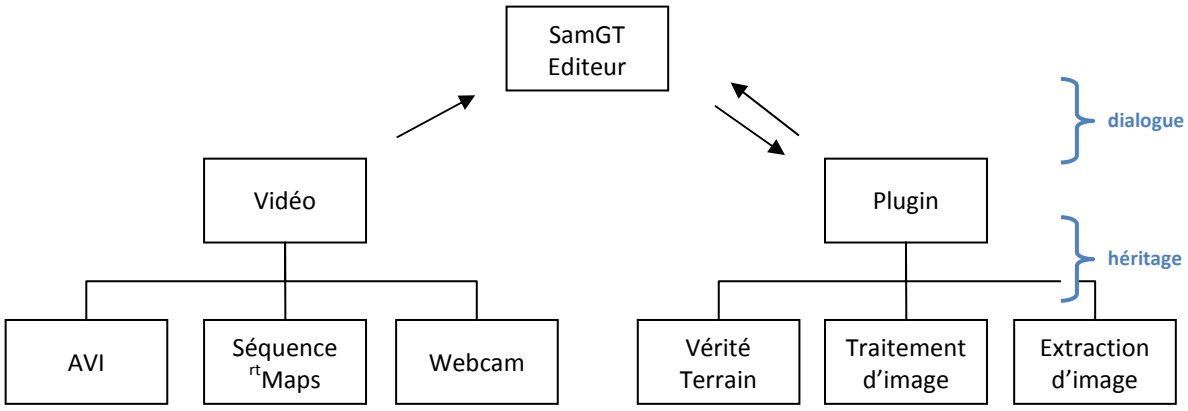



Figure 174 – Schéma de fonctionnement modulaire de l'éditeur SamGT

Les différents types de flux vidéo reconnus par l'éditeur (qui est, pour rappel, un logiciel indépendant de la plateforme rtMaps) sont bien entendu ceux supportés nativement par rtMaps (des formats vidéo propriétaires gérant la datation de chaque image à la microseconde) tel que les formats RAW et JSEQ, mais aussi ceux pouvant être habituellement utilisés sur ordinateur tel que le format AVI et les images vidéo provenant d'une Webcam.

Chaque image extraite, ainsi que les interactions utilisateur (clic sur l'image, déplacement de la souris, texte saisi au clavier,…) sont ensuite envoyées au travers de tous les plugins activés (i.e. choisis et paramétrés par l'utilisateur). Il est de ce fait possible de développer des plugins complexes pour l'éditeur de SamGT, tel qu'un éditeur de vérité terrain, un plugin de traitement d'image ou d'extraction de sous-image,…

Tous ces modules sont bien sûr normalisés et héritent tous d'un même module parent (concept de l'héritage en programmation), permettant de bien définir les fonctionnalités que doit implémenter chaque composant et notamment des fonctionnalités d'intégration graphique dans l'éditeur, un plugin pouvant donc définir ensuite ses propres fenêtres et paramètres visuels.

Il a ainsi été possible par exemple, d'encapsuler facilement les appels des méthodes des fonctions de reconnaissance de la bibliothèque OpenCV (Figure 175), de créer des plugins à affichage complexe, comme l'édition de diagramme de traitement d'image (Figure 176) manipulant ses propres plugins (Figure 177) ; et bien sûr, ce qui nous intéresse ici, la création de fichiers de vérité terrain (Figure 178).

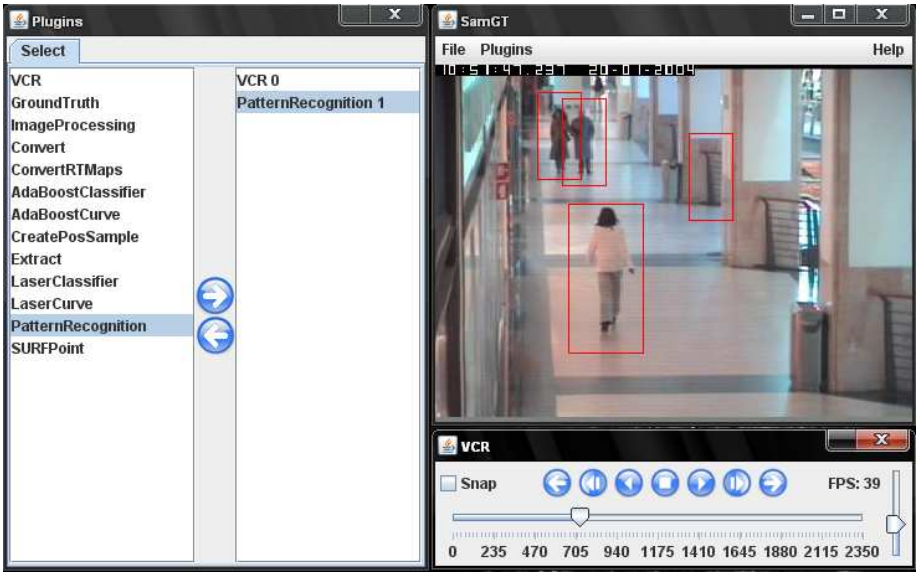



Figure 175 - Plugin de reconnaissance d'image OpenCV, sous SamGT

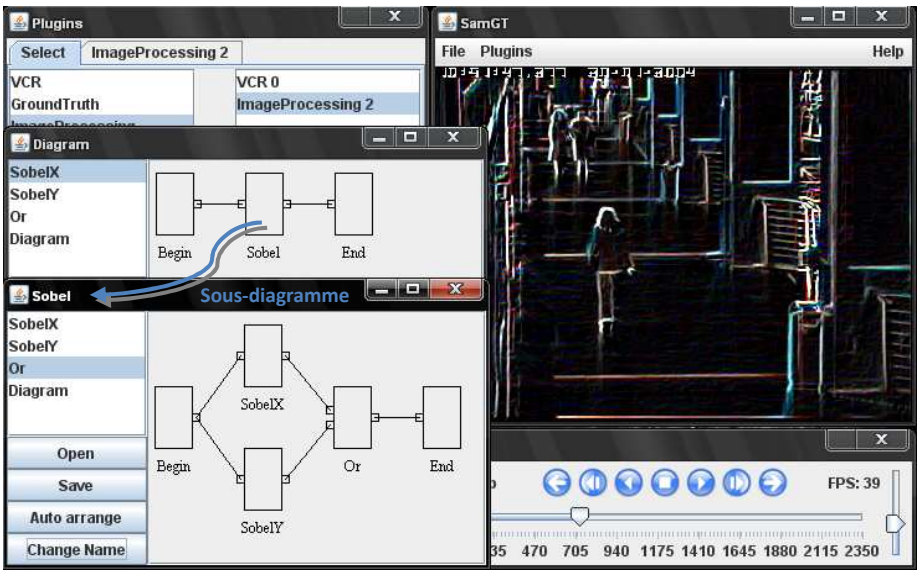



Figure 176 - Plugin de traitement d'image sous forme de diagramme sous SamGT

```
public class Or extends Plugin {
    public Object[] inputs  = {BufferedImage.class, BufferedImage.class};
    public Object[] outputs = {BufferedImage.class};

    public void process() {
        setOutput(0, Functions.or((BufferedImage)inputs[0], (BufferedImage)inputs[1]));
    }
}
```

Typage des entrées / sorties

Figure 177 - Définition d'un plugin de traitement d'image sous SamGT


Figure 178 - Création d'une vérité terrain sous SamGT

### b) *Le format de fichier*

Le format des fichiers de vérité terrain générés (d'extension « .gt ») a été défini afin de pouvoir être facilement lisible par un opérateur humain ou par une machine, et d'être évolutif (i.e. supporter l'ajout de nouveaux types ou sous-types d'objet détectables sans détériorer les anciens « formats »). Ainsi comme le montre l'Exemple 4, un fichier GT est un fichier texte dont chaque ligne correspond à un objet détectable dans une image. Par exemple, si 2 panneaux apparaissent sur une seule image, 2 lignes seront présentes dans le fichier, chacune d'elle correspondant à l'un des panneaux.

Cet exemple montre aussi que les données sont formatées de façon très stricte dans le fichier, permettant une lecture logicielle aisée.

```
Timestamp / frameindex x1 y1 x2 y2 id 'type' 'subtype'
00:00.0000 / 0 179 177 197 195 0 'Speed Limit Signs' '110'
00:00.0000 / 0 467 169 486 188 1 'Speed Limit Signs' '50'
00:00.0250 / 1 179 177 197 195 0 'Speed Limit Signs' '110'
00:00.0250 / 1 468 170 487 189 1 'Speed Limit Signs' '50'
00:00.0500 / 2 177 177 195 195 0 'Speed Limit Signs' '110'
00:00.0500 / 2 470 170 489 189 1 'Speed Limit Signs' '50'
00:00.0750 / 3 173 177 191 195 0 'Speed Limit Signs' '110'
```

**Exemple 4 – Exemple de fichier GT**

Chaque ligne de ce fichier comporte les informations nécessaires pour l'évaluation des algorithmes sous rtMaps et la relecture du fichier dans l'éditeur :

- le timestamp de l'image (i.e. sa datation), nécessaire pour la relecture sous rtMaps
- l'index de la frame, pour la relecture dans l'éditeur SamGT
- la position et la taille de l'objet, pour apparier avec les résultats des algorithmes
- l'identifiant unique de l'objet, permettant de le suivre temporellement dans le flux vidéo et de le différencier des autres objets.
- le type et le sous-type de l'objet à détecter, un fichier de vérité terrain pouvant contenir plusieurs types d'objets (panneau 110, panneau 50, véhicule, feu tricolore,…). Ces deux dernières informations sont écrites en toutes lettres, il sera donc possible d'ajouter de nouveaux types sans aucun problème.

L'un des problèmes pouvant découler de la création de fichier GT par différentes personnes pour une même séquence, est de pouvoir les rassembler en un seul fichier et de les partager avec tous. Le cas idéal serait que tous les membres du laboratoire (ceux travaillant sur les problématiques ADAS de détection) travaillent sur une même séquence de test et que, celui ayant créé une vérité terrain pour les panneaux de limitation de vitesse par exemple, puisse la fusionner avec celle de celui qui aurait travaillé sur la détection des feux tricolores ou des piétons ou des véhicules. Ceci est rendu possible via le site Sam où il est facile d'ajouter un fichier GT à une séquence, un algorithme sur le site se chargeant de fusionner cette nouvelle vérité terrain avec celle existante déjà sur le serveur (Figure 179).

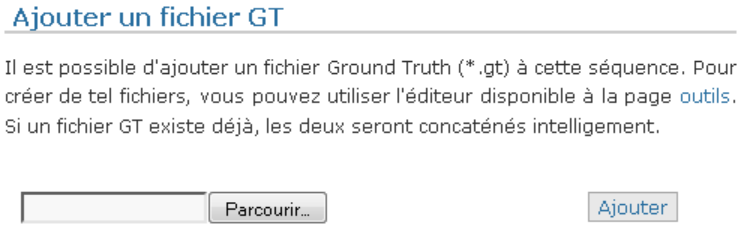



**Figure 179 - Partage des fichiers de vérité terrain sur le serveur SAM**

## 5.2.3. Evaluation des algorithmes

### a) Fonctionnement

L'évaluation des algorithmes avec SamGT est très simple : il suffit de charger sous la même plateforme de prototypage ([rt]Maps) avec laquelle les algorithmes sont développés, le paquet de modules qui vont servir à comparer en temps réel les résultats de l'algorithme avec le fichier de vérité terrain.

Ainsi, il suffit, comme le montre la Figure 180, de :

- placer le module « Decoder » paramétré avec le fichier de vérité terrain et des types et sous-type des cibles à détecter, permettant par exemple de n'extraire que les panneaux de limitation de vitesse du fichier de vérité terrain ;
- de relier la sortie de ce module (i.e. les cibles *ground truth*) ainsi que la sortie de l'algorithme à évaluer (composé d'un ou plusieurs modules) à un module d'appariement (« *Matching* »), qui peut être, soit temporel, soit frame par frame fournissant le nombre de bonnes détections (*vrais positifs*), de non détections (*faux négatifs*) et de mauvaises détections (*faux positifs*) ;

- de relier ces statistiques à un module de comptage pour calculer les statistiques globales des performances de l'algorithme sur l'ensemble de la séquence de test ; et de les enregistrer dynamiquement dans un fichier Excel (version 2003) ou de les visualiser directement sous [rt]Maps.

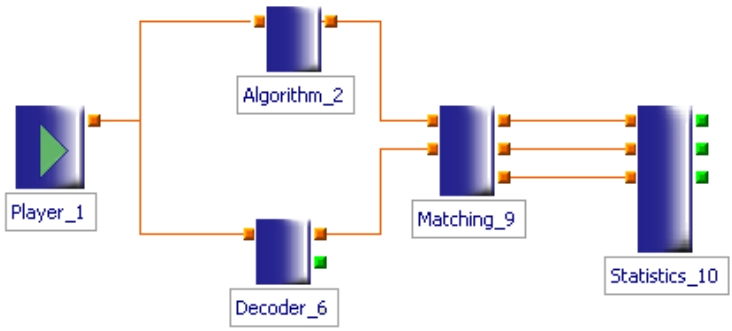



Figure 180 - Diagramme d'utilisation de SamGT

### b) Appariement des cibles

L'appariement des cibles issues de l'algorithme de détection et de celle de la vérité terrain est l'étape la plus importante dans SamGT. En effet, un mauvais appariement fausserait ou embellirait les résultats de l'algorithme développé. Ainsi ce module, après de nombreuses réflexions, propose plusieurs paramètres afin de répondre aux divers cas de figure pouvant entrer en jeu dans l'évaluation d'une détection.

Tout d'abord, la question fondamentale à se poser est de savoir comment considérer que deux cibles (celle de l'algorithme et celle du fichier de vérité terrain) correspondent à la même détection ? En effet, du fait de la discrétisation d'une image et des conditions environnementales (phénomène d'éblouissement de la caméra, arrière plan de l'objet dans l'image,…), il est impossible, même pour un opérateur humain d'encadrer correctement un objet dans une image. Dans l'exemple de la Figure 181, il est très difficile de savoir avec précision quelle est la limite physique du bord droit du panneau de limitation de vitesse. Certes il est possible d'appliquer des contraintes géométriques sur la taille des objets à détecter, afin d'aider le processus d'encadrement (contraintes incluses dans l'éditeur de SamGT), mais celui-ci restera, quoi qu'il arrive, imparfait.

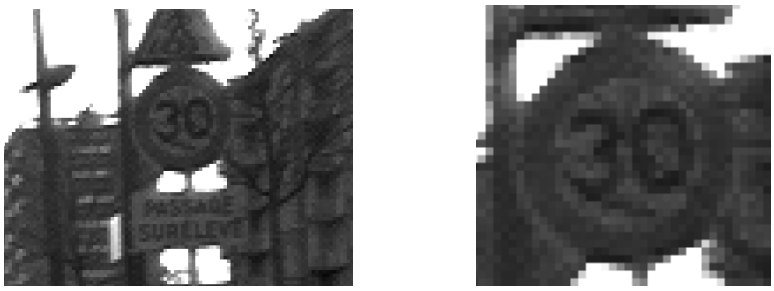


Figure 181 - Un panneau de limitation de vitesse de 30km/h (à gauche), la même image zoomée (à droite)

Ceci pour expliquer qu'une simple comparaison géométrique (taille et position dans l'image) n'est pas suffisante. Les deux cibles ne pouvant quasiment jamais être identiques de ce point de vue. De plus pour diverses autres raisons, un algorithme de détection peut fournir une cible plus petite (par exemple, dans le cas des détections des panneaux de limitation de vitesse, si il détecte le cercle intérieur du panneau et non l'extérieur) et / ou, plus ou moins décalée par rapport à la réalité (i.e. par rapport au fichier de vérité terrain), ainsi que le montre la Figure 182.

Figure 182 - Différentes détections possibles pour le panneau de limitation de vitesse par un algorithme (en vert) comparé à la vérité terrain (en rouge) bien que conservant un ratio de 1.

Ce problème d'appariement de cibles détectées n'est pas nouveau. Ainsi dans ses travaux sur la détection visuelle de véhicule, (Khammari, 2006) définit deux critères de mesure de similarité (Figure

183) : $S_1$ correspondant à l'aire de l'intersection divisée par l'aire de l'union des deux cibles (Figure 184); et $S_2$ qui est l'aire de l'intersection divisée par l'aire de la plus petite des deux cibles.

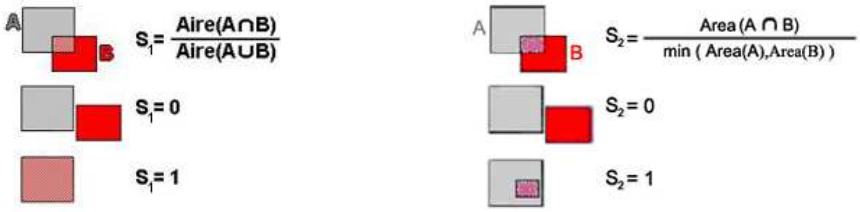



Figure 183 - Différents critères de similarité entre 2 cibles dans une image

La principale différence entre ces deux critères réside dans le fait que $S_2$ est plus tolérant vis à vis de l'écart entre tailles des cibles à comparer et est donc plus adapté à l'appariement de cibles à des instants différents quand l'objet à détecter, ou la caméra, est en mouvement.

La problématique ici est l'appariement de deux cibles (détections) correspondant au même instant. Ainsi le premier critère ($S_1$) répond au mieux à notre problème. Ce critère a donc été retenu, et un premier paramètre de seuil est à définir par l'utilisateur, correspondant donc au pourcentage de chevauchement toléré des deux cibles. Plus ce paramètre sera faible (proche de 0), plus l'appariement sera tolérant, et à l'inverse, plus ce paramètre sera important (proche de 1), plus l'appariement sera strict. Une valeur de 1 correspondant à un appariement parfait. La valeur de 0,75 est généralement employée pour tester nos algorithmes.

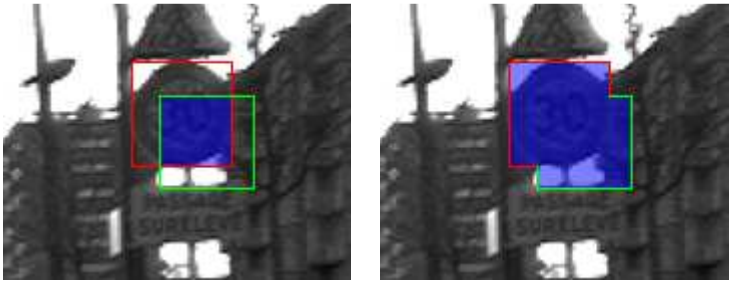


Figure 184 - Intersection des deux cibles (à gauche) et leur union (à droite)

Une autre question qui se pose est de savoir comment interpréter les résultats d'un algorithme détectant plusieurs fois le même objet dans une image. En effet, si un algorithme de détection peut se tromper sur la position et la taille de l'objet détecté, il peut aussi cumuler ses erreurs et détecter plusieurs fois cet objet sur une même image, comme le montre par exemple la Figure 185. Faut-il considérer toutes ces cibles comme une seule et bonne détection (si elles répondent toutes au critère de chevauchement), ou n'en considérer qu'une de correcte et les autres en tant que mauvaises détections ? La réponse à cette question est laissée au choix de l'utilisateur au travers d'un paramètre du module d'appariement, le choix changeant grandement les « résultats » de l'algorithme (i.e. le nombre de vrais positifs et de faux positifs,…).

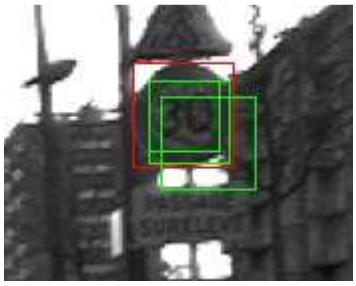


Figure 185 - Plusieurs détections possibles au même instant par un algorithme de détection (en vert), comparé à la vérité terrain (en rouge)

### c) *Statistiques générées*

Le module de statistique (celui qui effectue le comptage du nombre de vrais positifs, faux positifs,… issus du module d'appariement) génère dynamiquement deux fichiers de statistique :

- L'un au format Excel 2007 (*xml*) présentant des statistiques dans une dizaine de feuilles Excel.
- L'autre au format texte contient les données tabulées (qui sont les données brutes, aussi présentes dans le fichier Excel) facilement exploitable par un logiciel tiers ou si l'utilisateur désire développer son propre programme de traitement.

La première feuille du fichier Excel généré contient les statistiques brutes : un tableau représentant les données ligne par ligne pour chacune des détections, ou non détections, en précisant les informations nécessaires à leurs exploitation et interprétation :

- Le type de détection (vrai positif, faux positif ou faux négatif)
- Le type et sous-type de l'objet
- Diverses propriétés (position, taille, identifiant, timestamp,…)

Les autres pages du fichier fournissent des statistiques issues du croisement des données brutes, pour faciliter l'interprétation des résultats, ce qui est nécessaire pour l'étude des améliorations à apporter à ses algorithmes (i.e. savoir sur quel point doivent porter les efforts futurs). Par exemple sont calculés les statistiques sur les détections par rapport à :

- la taille des objets : il est ainsi possible de savoir par exemple à partir de quelle taille dans l'image l'algorithme arrive à détecter systématiquement les objets ;
- leur type et sous-type : savoir si certains objets sont mieux reconnus que d'autre (par exemple les panneaux à 3 chiffres par rapport aux panneaux à 2 chiffres)
- leur identifiant (savoir que le $n^{ième}$ objet n'est que très mal reconnu par rapport aux autres et de pouvoir directement visualiser dans la vidéo comment celui-ci se présente, si il est à moitié caché par exemple)
- …

Cependant, même si ces statistiques générées semblent idéales pour l'étude des algorithmes de détection, elles amènent quand même à se poser quelques questions et notamment sur leur interprétation en cas de détection temporelle et non image par image. En effet, que veulent dire par exemple, les statistiques sur la détection par rapport à la taille des objets d'un point de vue temporel ? En effet, dans un processus de détection temporelle, la seule chose qui peu importer est d'avoir validé la présence de l'objet quand il est présent, que ce soit 50 mètres avant d'être passé à côté du véhicule, ou simplement 10 mètres ; et non pas à réussir à détecter cet objet successivement sur chacune des images. Là aussi, il faudra donc faire attention aux paramètres à étudier, mais SamGT est développé dans un but générique pour l'utilisation de tous pour tout type d'algorithme de détection.

### *d) Evaluation de jeux de paramètres*

Mais l'utilisation de SamGT ne s'arrête pas là. Pour diverses raisons, il peut être nécessaire de tester un algorithme avec plusieurs paramètres possibles. Par exemple, l'utilisation d'un algorithme déjà existant oblige à trouver le bon jeu de paramètres à son exploitation pour répondre au mieux à un problème donné. Trouver ce bon jeu de paramètres peut être réellement fastidieux, même guidé par son expertise passée. Un autre cas de figure intéressant est de pouvoir évaluer son algorithme sous diverses conditions paramétrables en même temps ou, les unes à la suite des autres.

Dans ces deux cas, évaluer systématiquement son algorithme en changeant un à un les paramètres à tester et relever les résultats (i.e. le taux de bonnes détections, fausses détections,…) peut devenir très vite laborieux. En effet, dans le cas où le nombre de paramètres à tester n'est par exemple seulement que de 2, dont chacun peut avoir 10 valeurs différentes possibles, il faudrait évaluer l'algorithme 100 fois sur une séquence de test. Même si cette séquence est très courte, il faudrait un temps de travail assez conséquent pour relever tous les résultats (une séquence de 5min par exemple, demanderait environ 8h de test).

Pour palier à cette contrainte, SamGT propose un module $^{rt}$Maps afin d'évaluer automatiquement des jeux de paramètres. Celui-ci, se charge d'exécuter autant de fois que nécessaire l'algorithme, et d'enregistrer les statistiques pour chaque jeu de paramètres testé. Il est ainsi possible de faire tourner ces nombreux tests en tâche de fond, ou de les exécuter la nuit, et de récupérer l'ensemble des résultats une l'évaluation terminée.

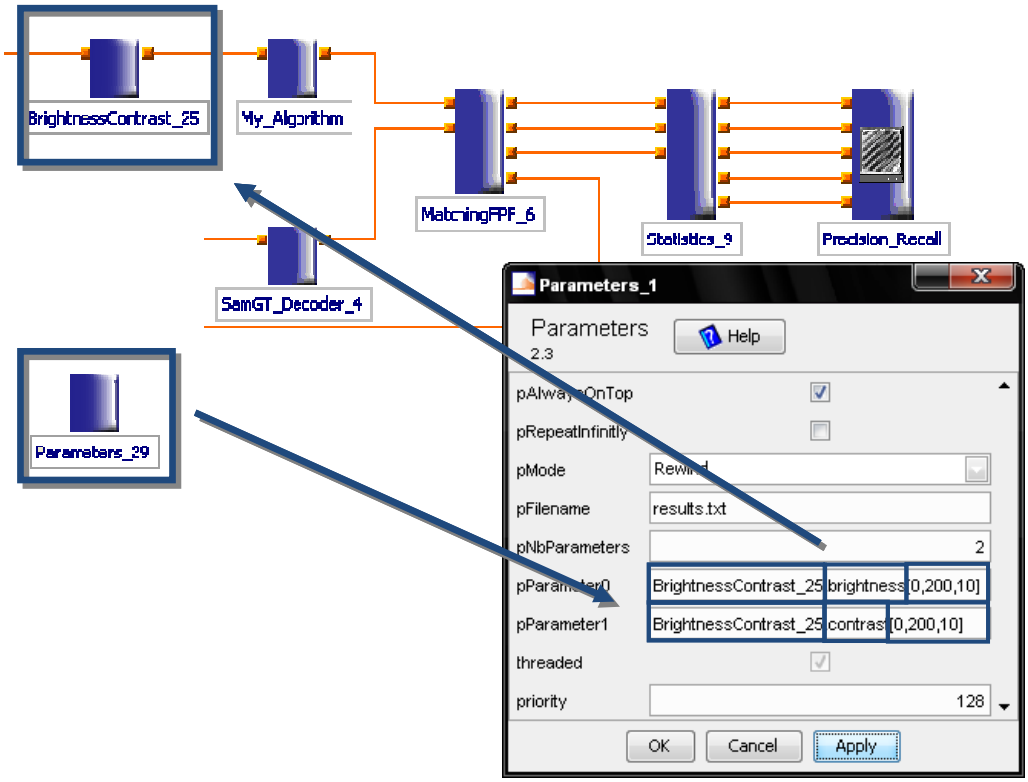



**Figure 186 - Schéma de l'évaluation d'un jeu de paramètres sous rtMaps avec SamGT : ici les valeurs des propriétés *brightness* et *contrast* du module *BrigthnessContrast_25* en amont de l'algorithme à évaluer, sont automatiquement changées entre 0 et 200 avec un pas de 10.**

Par exemple, dans le cas de l'évaluation de l'algorithme de reconnaissance des panneaux de limitation de vitesse, présenté au début de ce manuscrit, cet algorithme a été testé sous diverses conditions de luminosité et de contraste avec 20 valeurs possibles pour chacun de ces 2 paramètres. Soit 400 tests à effectuer sur la séquence de test. Bien entendu cette évaluation n'a pas été réalisée manuellement. Les résultats de ces 400 tests présentés Figure 87 ont été générés, à l'aide du diagramme schématisé par la Figure 186, où le module « *Parameters* » du package rtMaps de SamGT a permis de faire varier un à un ces deux paramètres.

L'avancement de la série de test ainsi qu'une heure estimée de la fin de l'évaluation de tous les jeux de paramètres est affichée dynamiquement dans une fenêtre de progression (Figure 89). Ceci permet à l'évaluateur de savoir quand récupérer ses résultats sans avoir à attendre (ce qui était l'un des buts de ce module).

A titre purement informatif, la Figure 187, présente le schéma rtMaps utilisé pour l'évaluation de l'algorithme de détection des panneaux de limitation de vitesse. Cette figure montre qu'il est tout aussi simple d'évaluer des paramètres sur un diagramme important (comportant plusieurs dizaines de modules).

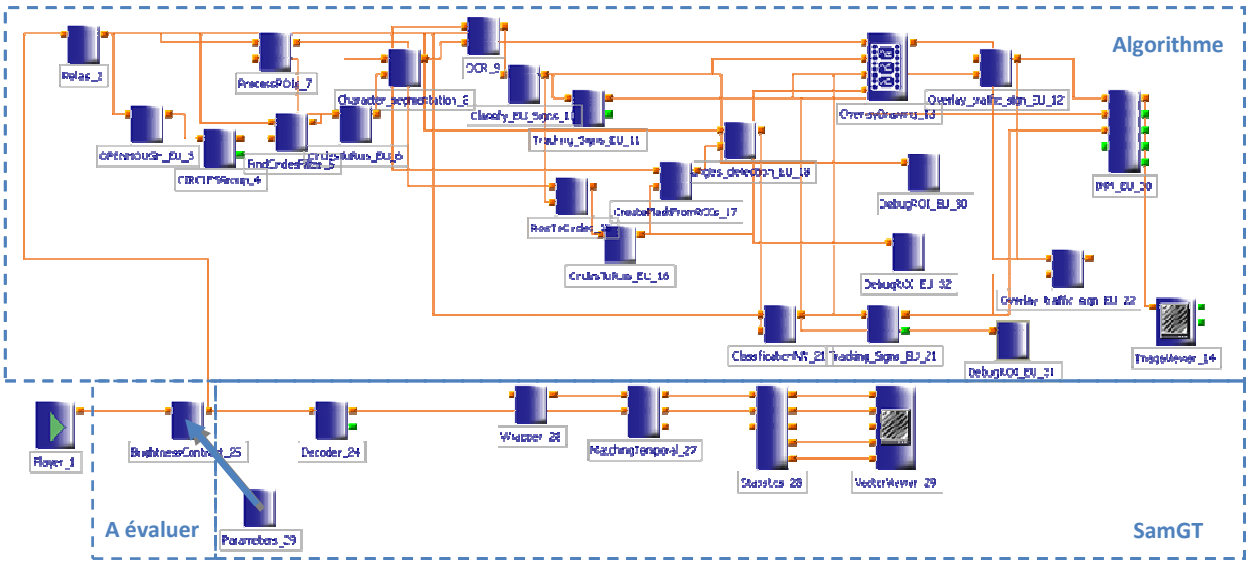



Figure 187 - Diagramme d'évaluation du jeu de paramètres luminosité / contraste pour de l'algorithme de détection des panneaux de limitation de vitesse présenté dans ce mémoire

## 5.3. Apprentissage numérique et classification d'image

### 5.3.1. Motivations

Enfin la dernière question soulevée dans ce chapitre est celle de l'amélioration des algorithmes d'apprentissage numérique par l'apport de nouvelles idées au sein d'un laboratoire de recherche. En effet, lors de la mise en place d'algorithmes de détection visuelle, il n'est pas rare d'utiliser de tels algorithmes d'apprentissage numérique ou tout simplement de porter ses travaux de recherche directement sur l'amélioration d'une méthode d'apprentissage.

Au sein même du centre de robotique, nous sommes nombreux à utiliser ces algorithmes (réseau de neurones, Adaboost,…) pour nos systèmes de détection visuelle (de panneaux, de voitures, de piétons,…), mais nous utilisions divers outils indépendamment des uns et des autres (Winseville pour Adaboost, Janet pour les réseaux de neurones,…). Même si certains de ces outils ont été développés puis améliorés en interne au laboratoire, notamment Winseville par (Abramson, 2005) qui vise à utiliser des algorithmes génétiques pour la génération de *classificateurs faibles* pour l'algorithme Adaboost ; amélioré par les travaux postérieur de (Ghorayeb, 2007) qui porta notamment ces algorithmes lourds en calcul sur carte graphique (bien plus puissante pour les calculs parallèles) ; il n'existait pas d'outil unifié et simple d'emploi pour l'utilisation de ces algorithmes.

Il existe certes quelques bibliothèques déjà existantes (OpenCV, Torch, LTI-Lib, Camellia, AForge,…) proposant un set d'algorithmes de classification / reconnaissance d'image, mais celles-ci ne sont pas vraiment adaptées à notre problématique. En effet, il est très difficile de rentrer dans le code source de ces bibliothèques. Celles-ci sont écrites en C/C++ afin de minimiser le temps d'exécution des algorithmes et elles cherchent souvent à pouvoir traiter n'importe quel type d'image (8bits, 12bits, 16bits, couleur, niveau de gris, avec des données alignées ou non,…) ce qui est, dans un contexte d'étudier et de développer des algorithmes de reconnaissance, pas vraiment intéressant. Le code source résultant est donc, en contre partie, plus ou moins complexe à appréhender et demande de nombreuses heures afin de pouvoir être complètement opérationnel dans l'ajout de fonctionnalités de ces bibliothèques.

L'idée du logiciel LeViS (LEarning algorithms in VIsion System) présenté dans cette section, vient donc du besoin de posséder un outil (une plateforme) simple et unifié au sein du laboratoire regroupant tous ces algorithmes couramment utilisés que chacun pourrait employer et améliorer. L'originalité de ce logiciel est donc inscrite dans ces buts fondamentaux qui sont de pouvoir :

- comparer aisément les performances de chacune des méthodes de reconnaissance sur diverses bases d'apprentissage ;
- utiliser facilement et rapidement au sein de nos applications un algorithme de détection ;
- améliorer facilement ces algorithmes via de nouvelles idées (en pouvant accéder et rééditer le code source simplement et aisément) ;
- créer de nouveaux algorithmes rapidement sans se soucier des problèmes de gestion de bases de données d'images pour l'apprentissage / les tests de ce nouvel algorithme et bien entendu le comparer aux autres déjà existants.

## 5.3.2. Présentation technique

### *a) Présentation générale*

L'outil LeViS est écrit en Java pour la grande lisibilité du code de ce langage et les grandes facilités qu'il apporte : reconnaissance native des formats d'images bmp, png, gif,… couramment utilisés dans les bases d'apprentissage d'image ; sa syntaxe proche du C++ utilisé dans nos développement sous [rt]Maps ; son ramasse miette (*garbage collector*) qui permet d'éviter les fuites mémoire en permettant au développeur de gagner du temps et de se focaliser sur son application ; la possibilité de création d'interface graphique native de ce langage (les composant graphique Swing) ; ainsi que sa compatibilité multiplateforme (Windows, Linux, 32bits, 64bits,…).

Comme le montre le schéma d'implémentation du logiciel Figure 188, un effort a été mené pour bien dissocier d'un point de vue conceptuel, ainsi que dans le code source du logiciel, les codes des algorithmes d'apprentissage, des algorithmes de classification, ainsi que de ceux de l'interface graphique.

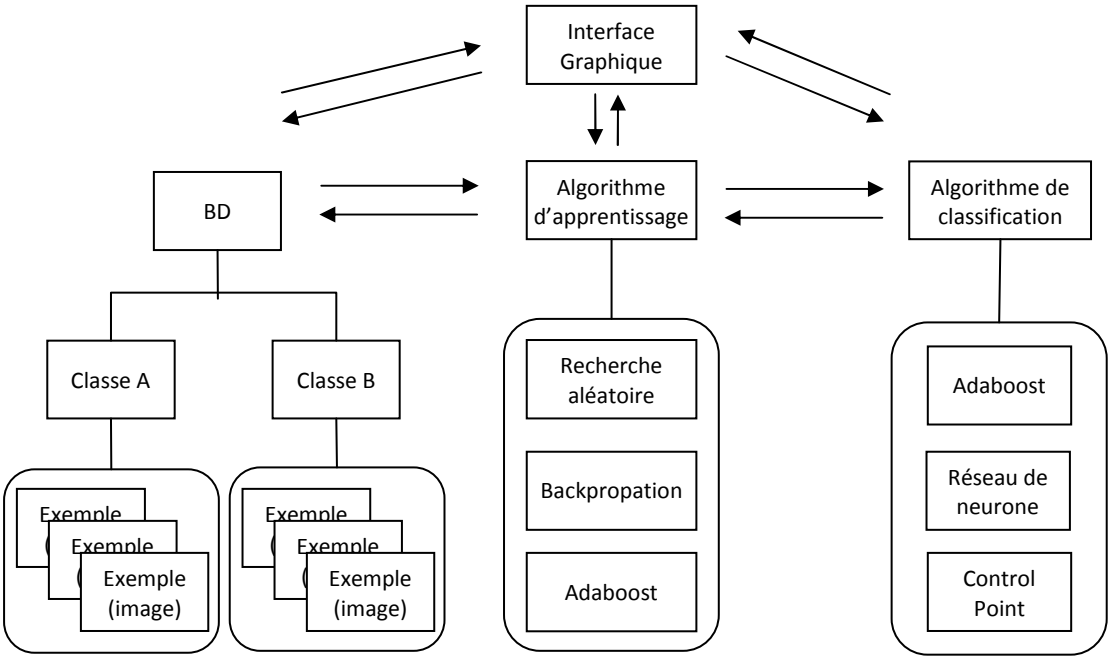



Figure 188 - Schéma conceptuel de LeViS

LeViS se ramène donc dans sa conception à la définition même d'un algorithme de classification et d'un algorithme d'apprentissage (Figure 189) :

- Un algorithme de classification doit être vu comme une boite noire paramétrable qui, lorsqu'une image lui est présentée, fournit la classe de cette image (i.e. piéton, voiture, moto, panneau de limitation de vitesse,…);

- Un algorithme d'apprentissage doit, par présentation successive d'images à un algorithme de classification, trouver le bon jeu de paramètres de cette boite noire afin que celle-ci se trompe le moins possible dans la classification des images présentées.

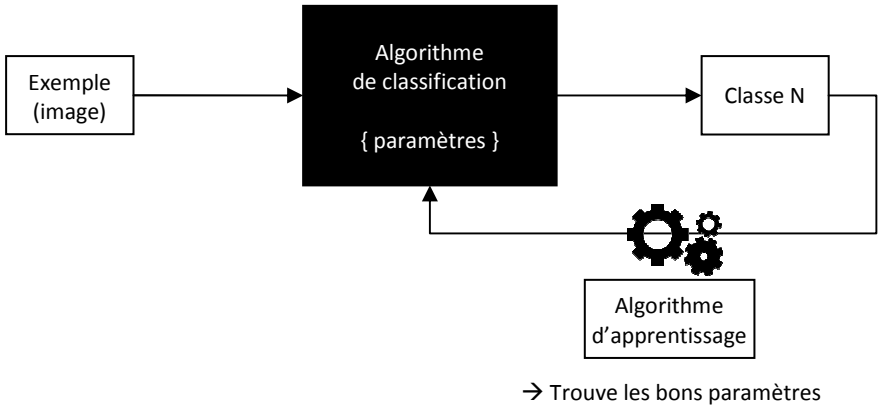



Figure 189 - Définition d'un algorithme d'apprentissage et d'un algorithme de classification

Certains algorithmes d'apprentissage sont propres à un algorithme de classification (par exemple la Backpropagation pour les Réseaux de neurones,…), mais d'autre sont génériques et peuvent être utilisés pour n'importe quel algorithme de classification. Ainsi, si un algorithme de classification implémente certaines fonctionnalités prédéfinies par l'un des algorithmes d'apprentissage (le concept d'*interface* du langage Java), il pourra automatiquement être utilisé avec celui-ci. Par exemple, l'implémentation d'une fonction « *randomize* » au sein d'un tel algorithme (fonction qui doit affecter des valeurs aléatoires aux paramètres), permet d'utiliser l'algorithme d'apprentissage de recherche aléatoire avec celui-ci.

Ce même concept permet aux algorithmes d'apprentissage et de classification, d'implémenter des fonctionnalités prédéfinies pour l'affichage graphique. Ainsi leurs divers paramètres sont automatiquement affichés et gérés par l'interface graphique, l'utilisateur pourra alors aisément choisir et paramétrer les algorithmes qu'il veut utiliser.

Avec une telle normalisation, il est possible d'implémenter chacun des algorithmes au sein de LeViS de façon indépendante (ce qui facilite leur programmation et la lisibilité du code); ainsi que de réutiliser directement les codes des algorithmes de classification (munis des fichiers de paramètres appris) dans d'autres logiciels.

Enfin l'utilisation du langage Java permet d'ajouter une « surcouche » pour utiliser les algorithmes d'apprentissage de façon distribuée sur plusieurs ordinateurs en réseau. En effet, le langage Java intègre depuis l'une de ses premières versions, la 1.1 en 1997, un moteur RMI – Remote Method Invocation. Sans entrer dans les détails, RMI est un ensemble de classes du langage Java permettant de manipuler des objets sur des machines distantes de manière similaire aux objets sur la machine locale sans se soucier des communications réseaux. Ainsi, un algorithme gourmand en temps de calcul peut être exporté facilement sur chacun des ordinateurs du réseau et être « multi-threadé » sur chacun des cœurs du processeur de chaque machine.

Tous les algorithmes de classification ont été de façon similaire (i.e. du code source indépendant) réécrits en langage C++ de façon à pouvoir être compilés au sein d'un package $^{rt}$Maps pour l'utilisation dans nos algorithmes de détection, et au sein d'un logiciel de test. Ce dernier permet de

comparer les résultats obtenus avec l'implémentation des algorithmes de classification en Java et celle en C++ sur une même base de test afin de vérifier que les deux implémentations fournissent les mêmes résultats. Une section dans le logiciel LeViS, où il suffit d'indiquer l'emplacement de ce logiciel (fichier exécutable) ainsi que l'emplacement de la base de test, permet d'automatiser ce processus. Il est alors possible de réécrire les algorithmes de classification dans un autre langage de programmation (Python, Caml,…), de les compiler dans un programme de test, et d'utiliser l'outil dédié dans LeViS pour valider leur implémentation.

### b) *Exemple d'implémentation d'algorithmes d'apprentissage et de classification*

Voici deux exemples montrant la simplicité d'écriture d'algorithmes d'apprentissage et de classification au sein de LeViS.

L'algorithme de classification Exemple 5 est très simple : il compare (fonction *classify*) la valeur de deux pixels dans l'image. Si la différence est supérieure à un seuil alors l'exemple est classé positif, sinon négatif (c'est donc un algorithme de classification binaire). Les coordonnées des deux pixels à comparer ainsi que le seuil sont les paramètres de cet algorithme. Ces paramètres peuvent être tirés aléatoirement via la fonction *randomize* que l'algorithme est obligé de définir vu qu'il implémente l'interface *IRandom* (définie plus loin).

```
public class DemoClassifier extends Classifier implements IRandom {

	private Point p1 = new Point();
	private Point p2 = new Point();
	private int threshold;
	private int maxThreshold = 255;

	public int classify(Sample sample) {
		int pix1 = sample.getPixel(p1);
		int pix2 = sample.getPixel(p2);
		return ((pix1 - pix2) > threshold) ? Classifier.POSITIVE : Classifier.NEGATIVE;
	}

	public void randomize() {
		p1.x = SRandom.getInstance().nextInt(_model.width);
		p2.x = SRandom.getInstance().nextInt(_model.width);
		p1.y = SRandom.getInstance().nextInt(_model.height);
		p2.y = SRandom.getInstance().nextInt(_model.height);
		threshold = SRandom.getInstance().nextInt(maxThreshold);
	}
}
```

**Exemple 5 - Exemple d'un algorithme de classification comparant 2 points dans une image**

La création d'un algorithme d'apprentissage est tout aussi simple. Par exemple, un algorithme d'apprentissage stochastique afin de trouver le bon jeu de paramètres d'un algorithme de classification est décrit par le code de l'Exemple 6. Dans cet exemple, la fonction d'apprentissage (la fonction *learn*) itère sans fin en créant à chaque tour un nouvel algorithme d'apprentissage du même type que celui en paramètre. Les paramètres de cet algorithme de classification sont tout d'abord initialisés aléatoirement (fonction *randomize* vue plus haut). Puis est calculée, de façon complètement transparente car utilisant des fonctions implémentées nativement au sein de LeViS, l'erreur de classification de cet algorithme sur la base d'apprentissage (fonction *computeLearningError*). Si cette erreur est plus faible que l'erreur du meilleur algorithme de classification déjà trouvé jusqu'ici, alors celui-ci est sauvegardé. Enfin, l'appel à la fonction *learnStep* de LeViS, permet de mettre automatiquement à jour le graphe d'apprentissage et de calculer la condition d'arrêt (typiquement si le nouveau taux d'erreur d'apprentissage est inférieur à un seuil).

```
public class DemoLearner extends Learner {

	public interface IRandom {
		public void randomize();
	}

	private Classifier classifierModel = new DemoClassifier();

	protected Classifier learn(DataBase learnData) throws Exception {
		Model sampleModel = learnData.getSampleModel();
		Classifier bestClassifier = null;
		double error = 1;
		while (_learnLoop) {
			Classifier classifier = classifierModel.getNewInstance(sampleModel);
			((IRandom)classifier).randomize();
			classifier.computeLearningError(learnData);
			if (classifier.learningError < error) {
				bestClassifier = classifier;
				error = classifier.learningError;
			}
			learnStep(bestClassifier);
		}
		addfoundClassifier(bestClassifier);
		return bestClassifier;
	}
}
```

**Exemple 6 - Exemple d'un algorithme d'apprentissage aléatoire**

Ces deux codes intégrés dans LeViS seront directement affichés dans l'interface graphique et utilisables. Il est donc clair que créer un nouvel algorithme d'apprentissage ou de classification, ajouter de nouvelles idées au sein du code d'un algorithme,… est une chose réellement aisée ce qui était l'une des motivations de la création du logiciel LeViS.

## 5.3.3. Algorithmes implémentés

Sont présentés ici de façon succincte les algorithmes d'apprentissage et de classification déjà implémentés au sein de LeViS qui étaient couramment utilisés au laboratoire. Ce paragraphe n'a pas pour but de constituer un état de l'art des systèmes d'apprentissage numérique pour la reconnaissance d'image, ni même d'être une étude comparative de ces différentes techniques. Le lecteur pourra se reporter aux références citées dans chacun des sous-paragraphes suivants afin de compléter sa lecture.

### *a) Réseau de neurones*

Un réseau de neurones artificiel est un modèle de calcul dont la conception est très schématiquement inspirée du fonctionnement des vrais neurones. Ce réseau calculatoire est en général composé d'une succession de couches dont chacune prend ses entrées sur les sorties de la précédente (Figure 190).

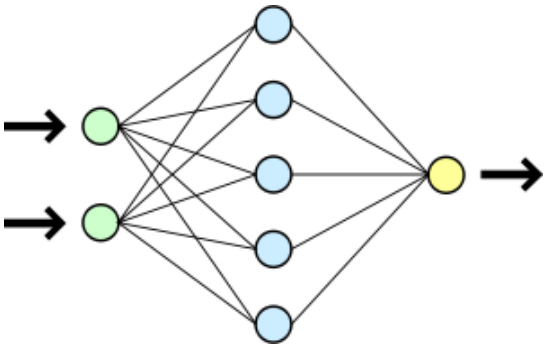


Figure 190 - Vue simplifiée d'un réseau artificiel de neurones

Chaque couche (i) est composée de $N_i$ neurones, prenant leurs entrées sur les $N_{i-1}$ neurones de la couche précédente. À chaque synapse est associé un poids synaptique, de sorte que les $N_{i-1}$ valeurs sont multipliées par ce poids, puis additionnées par les neurones de niveau i. Les valeurs calculées sont ensuite utilisées dans une fonction d'activation qui établira la force du signal (la valeur) envoyée par le neurone courant de niveau i (Figure 191). La fonction d'activation sert à introduire une non-linéarité dans le fonctionnement du neurone. Des exemples classiques de fonctions d'activation sont les fonctions : sigmoïde et tangente hyperbolique.

Au-delà de cette structure simple, le réseau de neurones peut également contenir des boucles qui en changent radicalement les possibilités mais aussi la complexité. De la même façon que des boucles peuvent transformer une logique combinatoire en logique séquentielle, les boucles dans un réseau de neurones transforment un simple dispositif de reconnaissance en une machine complexe capable de toutes sortes de comportements (Wikipedia, 2009).

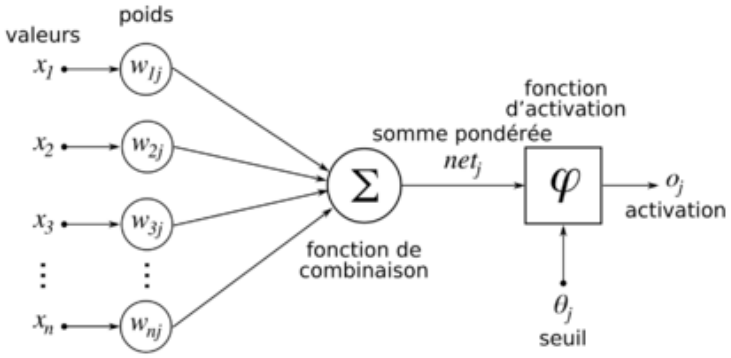



**Figure 191 - Structure d'un neurone artificiel. Le neurone calcule la somme de ses entrées puis cette valeur passe à travers la fonction d'activation pour produire sa sortie.**

La technique la plus populaire pour l'apprentissage (i.e. trouver les poids optimum des connexions des neurones) de tels réseaux est la rétropropagation du gradient (plus communément appelée par son terme anglais "*backpropagation*") : les valeurs calculées en sortie sont comparées avec les valeurs attendues pour calculer une fonction d'erreur prédéfinie. Par différentes techniques, cette fonction d'erreur est utilisée afin d'ajuster les poids des connexions des neurones dans le but de réduire cette erreur. Après avoir répété cette opération sur un nombre de cycles d'apprentissage assez important, le réseau converge habituellement vers un état où l'erreur calculée est faible.

(Cybenko, 1989) a émis et prouvé le théorème qu'un perceptron multicouche (un réseau de neurones sans boucle dont les connexions ne se font qu'avec les neurones de la couche précédentes, du type de la Figure 190) d'une seule et unique couche cachée (i.e. qui comporte une couche d'entrée, une couche intermédiaire dite cachée et une couche de sortie) est capable d'approximer n'importe quelle fonction continue à multiple variables et ce à n'importe quel degré de précision si les variables du réseau (poids, fonction de transfert, nombre de neurones de la couche cachée) sont bien choisies.

Le problème étant alors de choisir le bon nombre de neurones de la couche cachée (l'algorithme d'apprentissage de Backpropagation trouvant les poids des connexions).

Le lecteur désireux d'approfondir le sujet peut se reporter à (Kröse, et al., 1996) qui introduit d'une façon très claire toute la théorie mathématique de l'apprentissage pour les différentes topologies des réseaux de neurones : des simples perceptrons (Adaline, MLP,…), aux réseaux récurrents (Elman, Jordan, Hopfield,…), en passant par les réseaux auto associatifs (Kohonen, Art1, Art2,…) et par les réseaux stochastiques (Boltzmann).

LeViS intègre une forme simple des réseaux des neurones en tant qu'algorithme de classification: ceux de type Perceptron multicouche (chaque neurone d'une couche est connecté à chaque neurone de la couche précédente, sans boucle) dont les valeurs des neurones d'entrée représentent les valeurs des pixels d'une image à classifier; muni de l'algorithme d'apprentissage classique de rétropropagation du gradient.

### b) Adaboost

L'idée du boosting réside dans la question suivante : est-il possible de créer un algorithme d'apprentissage efficace (*strong learner*) à partir d'un ensemble d'algorithmes peu efficaces (*weak learner*) ?

Un algorithme peu efficace étant un algorithme un peu meilleur que le hasard (i.e. dont le taux d'erreur est un peu moins que 0.5). Au contraire, un algorithme efficace est un algorithme dont le taux d'erreur est proche de 0.

Il existe de nombreux algorithmes de boosting : Adaboost, GentleBoost, RankBoost, LPBoost,… Adaboost fut l'une des premières méthodes pleinement fonctionnelles permettant de mettre en œuvre le principe de boosting et a été appliqué avec succès à la reconnaissance de visage dans des vidéo en temps réel (Viola, et al., 2001).

Le principe de base de l'algorithme Adaboost (Adaptive Boosting) est de trouver un à un, d'une manière itérative, un ensemble de classificateurs faibles pour créer un classificateur fort. Chaque classificateur faible est appris sur la même base d'apprentissage exception faite que les poids des exemples sont pondérés en fonction de leur difficulté à être correctement classés par le classificateur fort courant. Ainsi, initialement tous les exemples ont un poids identique, puis à chaque étape, les poids des exemples mal classés par le classificateur fort sont augmentés. Ceci force le classificateur faible courant (en train d'être appris) à se concentrer sur les exemples difficiles.

Adaboost est adaptatif, dans le sens où les classificateurs faibles sont affublés d'un poids correspondant à leur efficacité (i.e. à leur taux d'erreur de classification de la base en cours, qui contient donc des exemples à probabilités différentes). Ces poids permettent d'effectuer un vote pondéré afin de décider de la classe finale de l'objet à reconnaitre. Vers la fin de l'apprentissage, le poids des exemples difficiles à apprendre devient largement dominant. Si on peut trouver un classificateur faible performant sur ces exemples, il sera alors doté d'un coefficient considérable. L'une des conséquences possibles est que les exemples bruités, sur lesquels finit par se concentrer l'algorithme, perturbent l'apprentissage.

(Freund, et al., 1999) qui sont à l'origine de l'algorithme Adaboost, ont prouvé que si chaque classificateur élémentaire assemblé a une erreur inférieure à 0.5, alors l'erreur empirique du classificateur fort décroit exponentiellement avec le nombre d'étapes.

Malheureusement l'algorithme d'Adaboost n'est applicable que pour la création de classificateur binaire, ce qui le rend inutilisable pour le problème de classification des panneaux de limitation de vitesse. Mais des travaux de recherche en cours au sein du laboratoire visent à rendre cet algorithme multi-classes.

La Figure 192 montre un exemple simple d'application d'Adaboost pour classifier des données. La taille de chaque exemple (+ ou -) est proportionnelle à son poids.

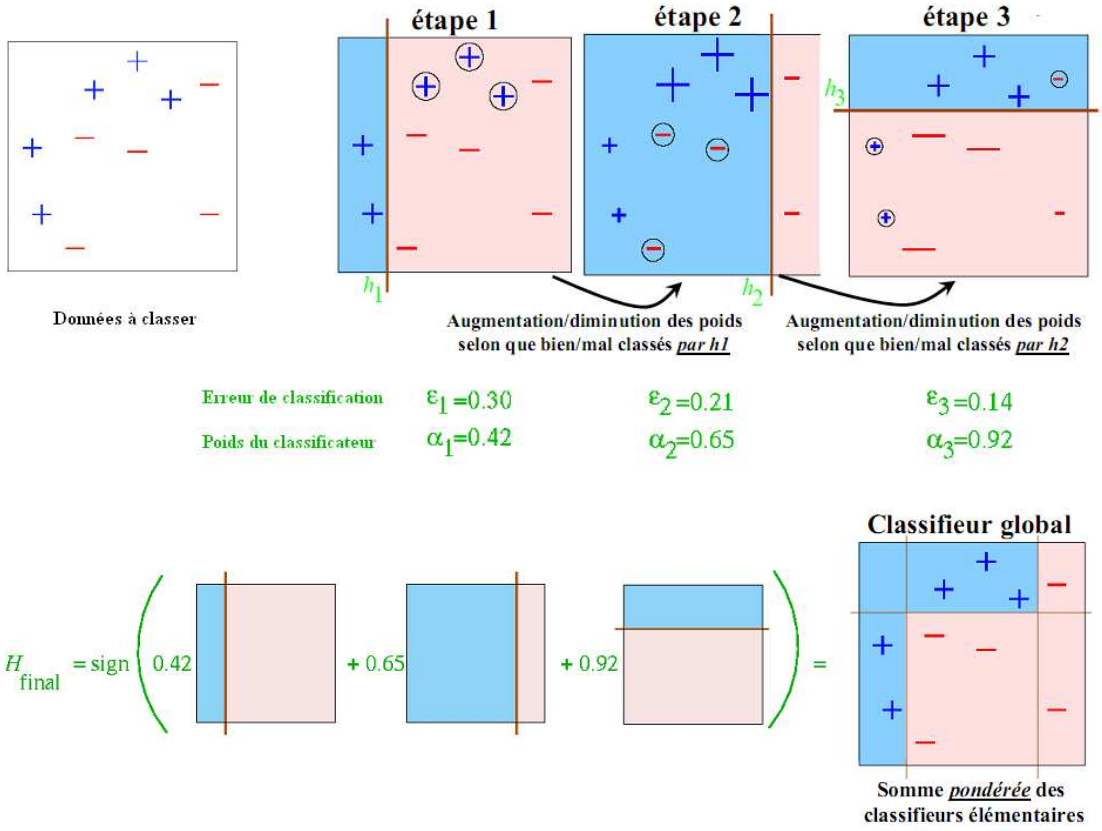



Figure 192 - Exemple d'application de l'algorithme Adaboost

Dans le domaine de la reconnaissance visuelle d'objet, les classificateurs faibles couramment utilisés sont ceux introduits par (Viola, et al., 2001) : les classificateurs de Haar. Ces classificateurs calculent la différence absolue entre la somme des valeurs des pixels en bleu et en blanc. Si cette différence est supérieure à un seuil, alors l'exemple est classé positivement (Figure 193).

Figure 193 - Exemple de classificateurs de Haar

D'autres classificateurs faibles ont été proposés pour Adaboost comme les Control-Points (Abramson, et al., 2005) qui sont plus rapides à calculer et plus robustes aux variations d'illumination (Figure 195). Ils sont constitués de deux groupes de points (positifs et négatifs). Un exemple est classé positif, si la différence entre la valeur minimale représentée par un pixel positif et la valeur maximale d'un pixel négatif (et inversement) est supérieur à un seuil (Figure 194).

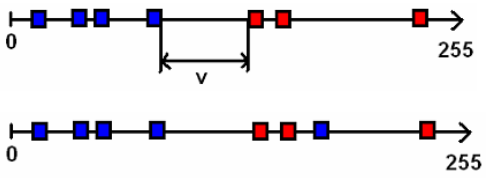



Figure 194 - Représentation linéaire d'un classificateur de type Control-Point : en haut exemple positif respectant le seuil V - en bas exemple négatif

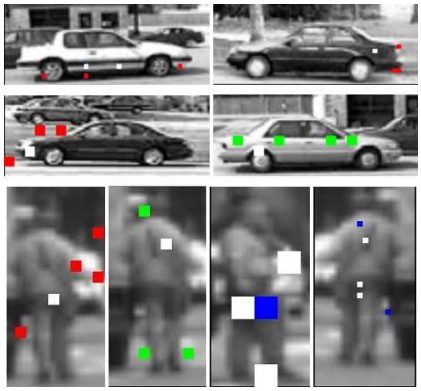


Figure 195 - Exemple de classificateur Control-Points

Une amélioration des Control-Points, les Connected-Control-Points, a été apporté par (Stanciulescu, et al., 2007) en ajoutant une contrainte de connexité sur un voisinage de 8, ce qui implique que chaque point doit toucher au moins un autre point par un coin (Figure 196). Ils ont prouvé que l'ajout de cette contrainte permet d'améliorer significativement l'algorithme.

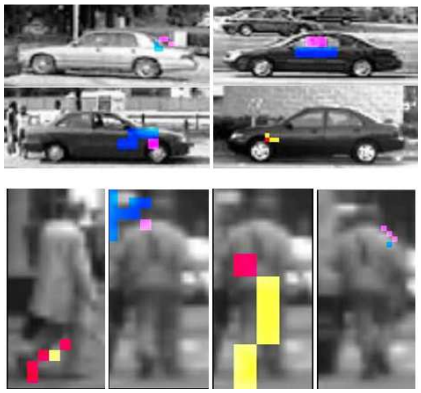


Figure 196 - Exemple de classificateur Connected-Control-Points

Dans LeViS sont actuellement implémentés ces 3 types ce classificateurs faibles, ainsi que l'algorithme Adaboost et sa variante cascadée.

Le lecteur désireux d'approfondir le sujet pourra se référer à (Meir, et al., 2003) et en ce qui concerne son application à la reconnaissance d'image à (Viola, et al., 2001) et (Abramson, 2005) ou encore (Schapire, 2009) qui contient une collection d'articles de référence mise à jour régulièrement.

#### *c) Meta learner*

Il ne s'agit pas vraiment d'un algorithme d'apprentissage en tant que tel, mais d'une fonctionnalité que LeViS implémente sous la forme d'un tel algorithme grâce à la grande généricité du code source de cet outil.

Il est parfois difficile de trouver les bons paramètres d'un algorithme (nombre de neurones de la couche cachée d'un réseau de neurones, nombre de points de contrôle à utiliser dans Adaboost,…) et il serait bon d'avoir un « outil » afin d'aider à les choisir.

Ainsi, au sein de LeViS a été implémenté un algorithme d'apprentissage qui, à l'instar de SamGT, teste successivement toutes les valeurs définies en paramètre d'un autre algorithme, et effectue un apprentissage pour chaque jeu de paramètres sur la même base de données. Les valeurs des paramètres testés sont sauvegardées dans un fichier, ainsi que les graphes d'apprentissage correspondants. Il suffit alors de consulter ces graphes en fin d'apprentissage pour choisir le meilleur jeu de paramètres à utiliser.

Le choix automatique du meilleur jeu de paramètres n'est pas aisé et soulève de nombreuses questions. En effet, il est possible, dans le choix de la topologie d'un perceptron multicouches par exemple, d'atteindre un meilleur taux d'erreur avec un grand nombre de neurones sur la couche cachée qu'avec un nombre moindre. Cependant si en utilisant moins de neurones, le taux d'erreur reste acceptable (ou est très peu supérieur au meilleur taux), la question de savoir laquelle des deux topologies est la meilleure pour l'application visée ne peut avoir de réponse automatique : meilleur taux d'erreur avec une forte charge calculatoire, ou taux d'erreur moins bon, mais algorithme plus rapide. Ce type de question peut se poser pour tous les algorithmes / jeux de paramètres à évaluer. C'est pourquoi, aucun processus de choix automatique n'a été implémenté au sein de ce Meta-Learner, surtout qu'il est rapide d'analyser visuellement un a un les graphes d'apprentissage.

### 5.3.4. Expérimentations

Dans un premier onglet (Figure 197), LeViS propose le chargement d'une base de données avec certaines options dont :

- La taille de ré-échantillonnage des exemples
- Les prétraitements à effectuer sur chacun des exemples (égalisation d'histogramme, conversion en image de gradient,…)

- Le canal sur lequel effectuer l'apprentissage (couleur, rouge, vert, bleu ou en niveau de gris)
- Le facteur de séparation pour créer les bases d'apprentissage et de test (typiquement 66%). La base de test servant à vérifier que l'algorithme arrive à correctement classer des exemples qui n'ont pas servit à l'apprentissage (i.e. des exemples « inconnus »).

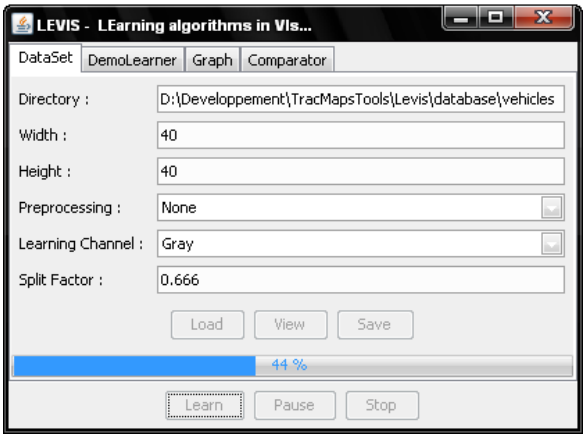



Figure 197 - Choix d'une base de données sous LeViS

Une contrainte majeure de nombreux algorithmes de classification d'images réside dans le fait qu'ils doivent traiter des données qui ont toujours les mêmes dimensions. Or, choisir une taille de ré-échantillonnage n'est pas aisé. Pour aider à ce choix, LeViS propose un outil visuel qui permet de visualiser dynamiquement l'image ré-échantillonnée par le dépacement de *sliders* (Figure 198).

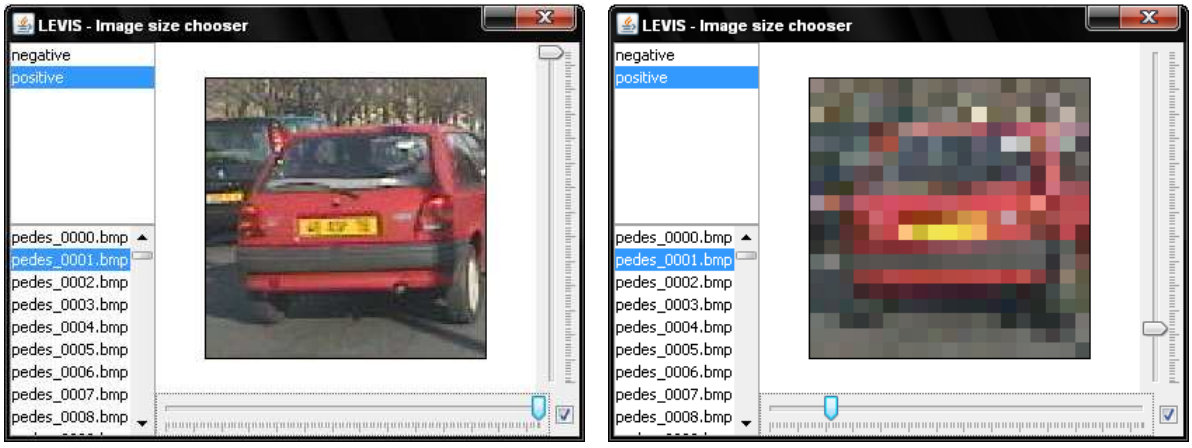



Figure 198 - Choix d'une taille de ré-échantillonnage sous LeViS : à gauche taille réelle de l'image, à droite image sous-échantillonnée

Voici à titre d'exemple les résultats obtenus avec les algorithmes d'apprentissage et de classification présentés plus haut dans les Exemple 5 et Exemple 6. Pour rappel ce classificateur calcule simplement la différence des valeurs de 2 pixels, et, si cette différence est supérieure à un seuil, classe l'exemple en tant que positif sinon en tant que négatif (ici en tant que voiture ou non voiture).

L'algorithme d'apprentissage trouve itérativement le meilleur classificateur en affectant des valeurs aléatoires aux positions des points et du seuil.

LeViS permet de visualiser en temps réel l'évolution du graphe d'apprentissage, i.e. les taux d'erreur de classification sur la base de test et d'apprentissage, courbes rouge et bleue, en fonction de l'itération (Figure 199). Une fois un classificateur appris, il est possible de visualiser, dans une nouvelle fenêtre, les exemples mal classés de la base de test ou d'apprentissage ainsi que, si il implémente des fonctions d'affichage, de visualiser le classificateur sur une image de la base (Figure 200 – Les deux points du classificateur sont affichés : en bleu, sous la voiture, et rouge au coin inférieur gauche de l'image).

Il est intéressant de voir que ces algorithmes simples arrivent à classifier une base de voiture avec un taux d'erreur de seulement 17%. Un tel taux d'erreur n'est pas admissible pour utiliser cet algorithme de classification directement dans un système ADAS, mais laisse envisager de pouvoir l'utiliser conjointement avec d'autres algorithmes (boosting,…).

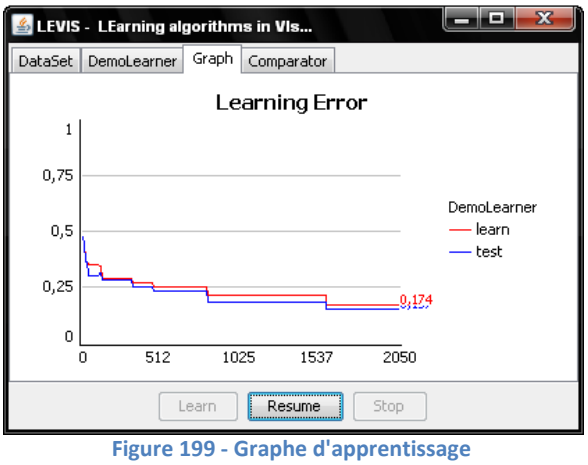



Figure 199 - Graphe d'apprentissage

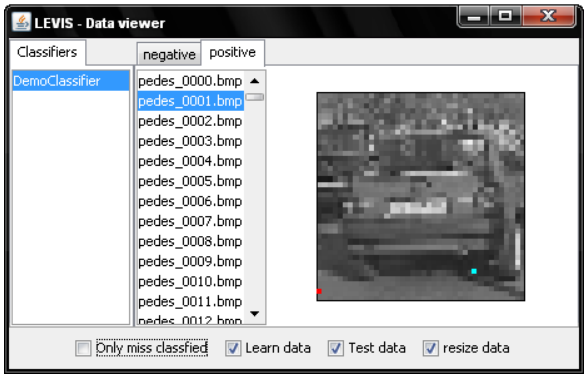



Figure 200 - Fenêtre d'affichage de la base de données et des classificateurs

En effet, l'utilisation de ce classificateur avec l'algorithme Adaboost permet, comme le montre la Figure 201, d'obtenir un taux d'erreur de classification de 0% sur la base d'apprentissage et de 3% sur la base de test. Le classificateur implémentant l'interface *Randomize* peut être utilisé avec l'algorithme d'apprentissage aléatoire multithreadé proposé dans LeViS. Le graphe d'apprentissage montre l'évolution de l'apprentissage de chaque thread de chacun des algorithmes d'apprentissage en cours d'exécution (i.e. ceux de l'algorithme d'apprentissage fort et ceux du faible). La Figure 202 montre l'ensemble des classificateurs faibles appris par Adaboost appliqués sur une image de la base.

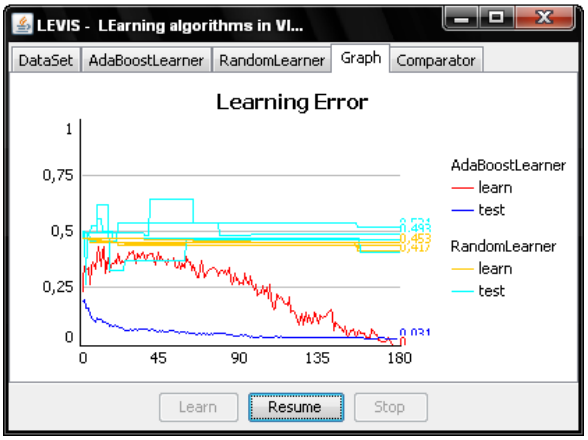



Figure 201 - Graphe d'apprentissage

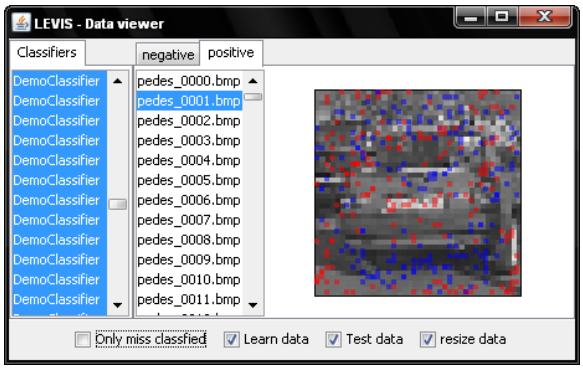



Figure 202 - Fenêtre d'affichage de la base de données et des classificateurs

Enfin, l'utilisation l'algorithme *MetaLearner*, permet par exemple d'aider au choix du nombre de neurones de la couche cachée d'un Perceptron Multicouches (Figure 203), qui est pour rappel, la principale variable « difficile » à trouver, l'algorithme d'apprentissage (la *Backpropagation*) se chargeant de trouver (apprendre) les poids des connexions entre les neurones. L'exemple est ici appliquée sur la base de chiffres ayant servi à l'apprentissage pour l'algorithme de détection visuelle des panneaux de limitation de vitesse présenté dans le premier chapitre de ce document. Le nombre d'itérations par apprentissage du réseau de neurones est limité à 750 pour cette étude.

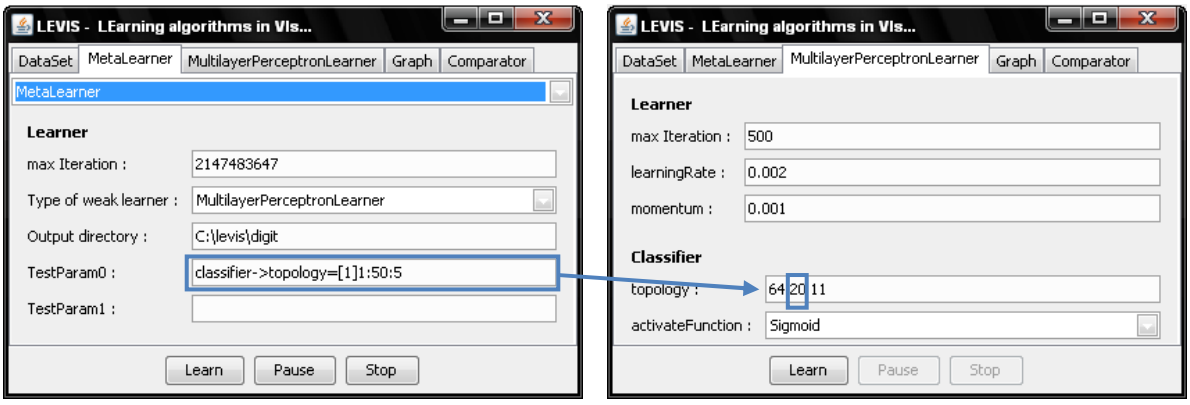



Figure 203 - Le *MetaLearner* sous LeViS, permet d'effectuer automatiquement des apprentissages avec des paramètres différents

Les résultats obtenus sont représentés par les Figure 204 à Figure 210. A partir de 20 neurones sur la couche cachée, la baisse du taux d'erreur sur les bases d'apprentissage et de test n'est plus significative (36% pour 20 neurones, 35% pour 30). C'est ce nombre de 20 neurones qui a été choisi pour le perceptron multicouche de reconnaissance des chiffres utilisé dans l'algorithme de détection des panneaux de limitation de vitesse.

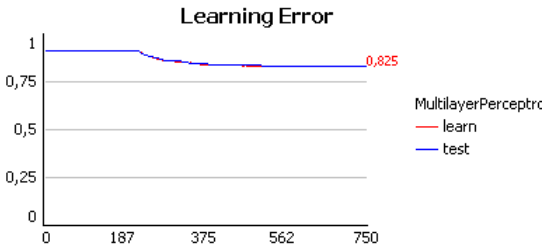



Figure 204 - Graphe d'apprentissage d'un Perceptron Multicouches à 1 neurone sur la couche cachée sur la base de chiffres

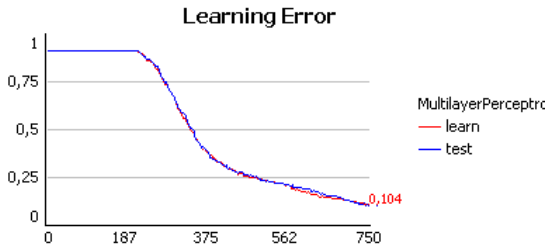



Figure 205 - Graphe d'apprentissage d'un Perceptron Multicouches à 11 neurones sur la couche cachée sur la base de chiffres

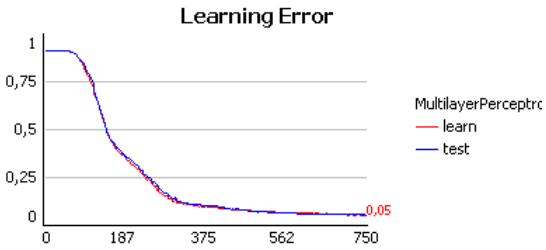



Figure 206 - Graphe d'apprentissage d'un Perceptron Multicouches à 16 neurones sur la couche cachée sur la base de chiffres

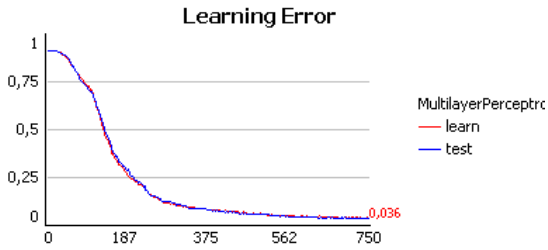



Figure 207 - Graphe d'apprentissage d'un Perceptron Multicouches à 21 neurones sur la couche cachée sur la base de chiffres

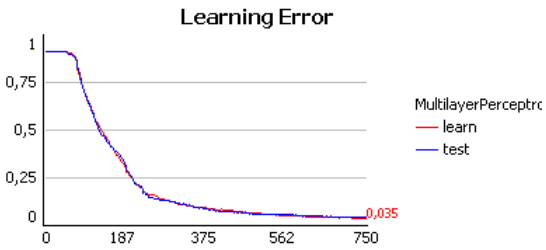



Figure 208 - Graphe d'apprentissage d'un Perceptron Multicouches à 31 neurones sur la couche cachée sur la base de chiffres

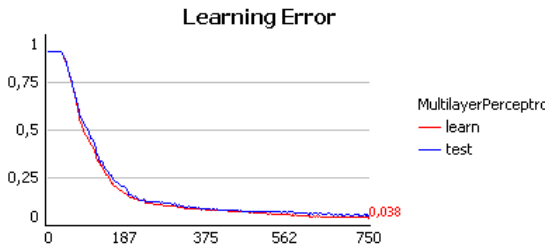



Figure 209 - Graphe d'apprentissage d'un Perceptron Multicouches à 41 neurones sur la couche cachée sur la base de chiffres

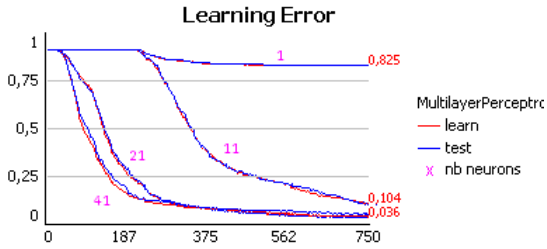



Figure 210 - Superposition des graphes d'apprentissage pour 1 à 41 neurones sur la couche cachée d'un Perceptron Multicouches sur la base de chiffres

# 5.4. Serveur d'acquisitions rtMaps

## 5.4.1. Présentation générale

Le logiciel rtMaps utilisé au laboratoire et pour mes travaux de recherche, est un logiciel permettant entre autres d'acquérir, d'enregistrer et de rejouer des bases de données multisensorielles. Ces bases de données peuvent donc contenir des séquences d'images, des fichiers son, des fichiers contenant des données issues de capteurs divers choisis par l'utilisateur et en fonction de l'application visée. Les séquences acquises au laboratoire sont essentiellement formées de données issues de capteurs couramment utilisés dans l'automobile, soit pour la détection de l'environnement soit pour relever les paramètres d'état du véhicule porteur. Il s'agit de : caméras, télémètres laser, radars, ultrasons, GPS, gyromètres, bus CAN, centrales inertielles, etc.

Tout cela explique que les bases de données peuvent présenter des contenus de tailles différentes et de « syntaxes » différentes et variées. Stocker et archiver tous ces enregistrements nécessite donc un espace disque important mais aussi un effort d'indexation et de référencement efficace et dédié.

Les utilisateurs de rtMaps, tout au moins au sein du laboratoire, n'ayant pas de lieu de partage structuré, se contentent généralement de sauvegarder leurs séquences personnelles (i.e. celles qu'ils ont acquises eux-mêmes pour leurs travaux de recherche) sur leurs propres ordinateurs. Ainsi, quand une autre personne a besoin d'une séquence particulière, il est amené, soit à demander à chacun si il n'aurait pas ce type de séquence, soit à aller ré-effectuer cet enregistrement, ce qui peut être, dans les deux cas, long et fastidieux.

Le Serveur d'Acquisition rtMaps (SAM) a ainsi été créé afin de répondre à ces besoins. Telle une bibliothèque ou plutôt une médiathèque, le serveur SAM présente un espace énorme dans lequel sont archivées et stockées des centaines de bases de données rtMaps. Celles-ci sont classées selon l'auteur de l'acquisition, la date, le contenu (c'est-à-dire les capteurs qui nourrissent chaque base) mais aussi le thème (c'est-à-dire le type d'environnement acquis, de quel temps, ...). Comme pour tout serveur d'archivage, l'utilisateur, après enregistrement, peut y rechercher des séquences selon divers critères et en ajouter de nouvelles, l'accès y étant facilité via l'utilisation d'un site intranet dédié et développé spécifiquement.

## 5.4.2. Rechercher une séquence

### *a) Par critères*

Afin de faciliter la recherche d'une séquence parmi les centaines de gigaoctets de données déjà présents sur le serveur SAM, le site intranet propose une page de recherche permettant de retrouver une séquence de trois façons différentes (Figure 211).

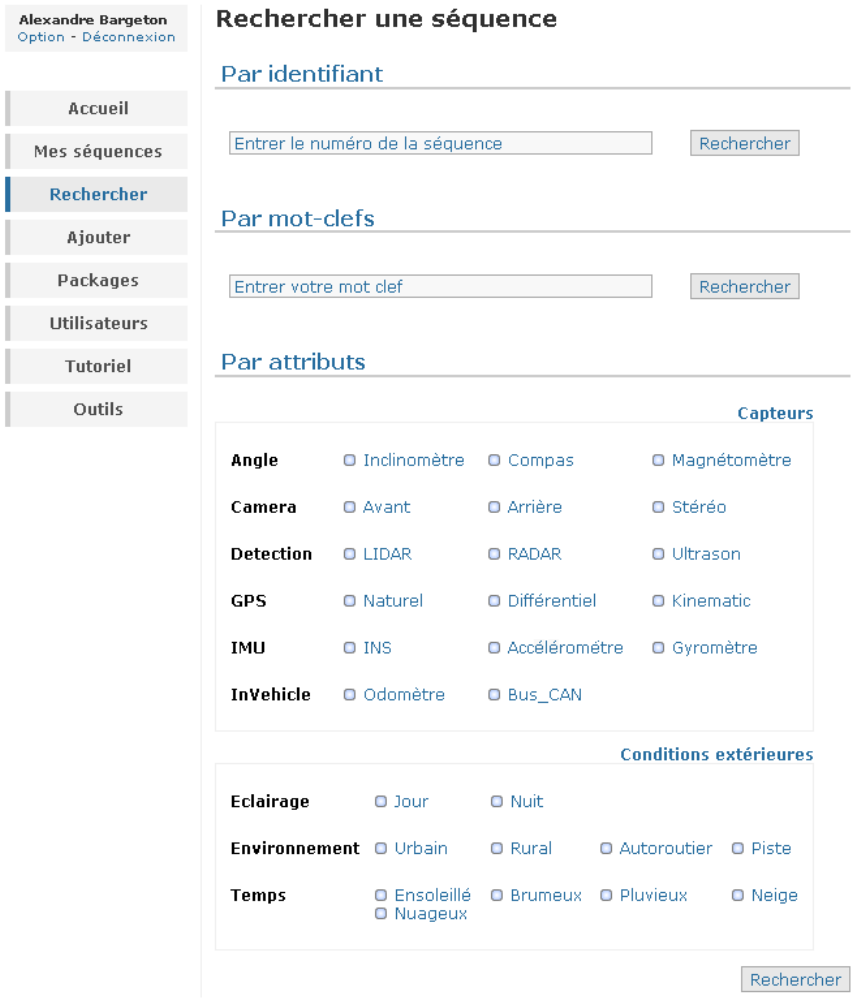



Figure 211 - Page de recherche de séquences du site intranet du server SAM

1. Soit l'utilisateur connaît l'identifiant de la séquence à rechercher, il pourra alors dans ce cas accéder directement à la séquence. Cette méthode simple, rapide et efficace, permet de partager des séquences facilement avec d'autres utilisateurs en diffusant directement leurs identifiants

2. Soit, si il a des critères précis sur les types de capteurs utilisés (caméra avant, laser,…) et de conditions extérieures lors de l'acquisition (de jour, avec de la pluie,…), l'utilisateur pourra alors effectuer sa recherche par attributs.

3. Soit il peut effectuer une recherche par mots clefs, qui seront retrouvés parmi les descriptions des séquences (par exemple : le nom d'un lieu, si on sait que l'acquisition y a été effectué) au cas où la liste des attributs disponibles ne suffirait pas à satisfaire sa requête.

### b) Par liste

Une autre méthode pour retrouver facilement des séquences sur le serveur est l'utilisation des listes de séquences. Ainsi, sur le site intranet de SAM, il est possible de définir des listes de séquences à la manière des « favoris / bookmarks » des navigateurs web. Cela peut permettre, par exemple, de créer une liste de séquences par projet ou par type de travaux de recherche afin de les retrouver / partager au plus vite. Il est ainsi possible de regrouper toutes les séquences utilisées par exemple pour ses travaux de détections des panneaux de limitation de vitesse,…

Une liste peut bien entendu être créée, avec un nom spécifié par l'utilisateur, enrichie jour après jour de nouvelles séquences, et enfin une liste peut être supprimée (Figure 212).


**Figure 212 - Gestion des listes de séquences et liste des séquences ajoutées sur le site intranet du server SAM**

### c) Par séquence ajoutée

Enfin, l'utilisateur a la possibilité de connaître toutes les séquences qu'il a lui même ajoutées. Généralement ce sont ces séquences auxquelles il accédera souvent ou auxquelles il fera référence afin de les partager avec ses collaborateurs, en leur fournissant directement leurs identifiants par exemple (Figure 212).

## 5.4.3. Télécharger une séquence

### a) *Les droits d'accès*

Les séquences dont nous disposons au sein du laboratoire peuvent provenir de sources différentes :

- de nos propres enregistrements ;
- de nos partenaires (ex : Valeo, PSA LiViC, INRIA, …)

Il est toujours intéressant de pouvoir stocker le maximum de séquences provenant de toutes ces sources et de les proposer en une utilisation interne à tous les chercheurs du laboratoire. En effet, nos partenaires peuvent avoir des capteurs particuliers ou avoir enregistré des séquences sous des conditions spécifiques qu'il serait difficile de reproduire. Ainsi pouvoir proposer ces séquences aux autres membres du laboratoire peut permettre un gain de temps considérable pour les travaux de recherche du laboratoire.

Cependant, en gage de respect de la propriété intellectuelle, il faut pouvoir garantir que les données provenant d'un de nos partenaires industriels ne soient pas fournies à un autre.

Ainsi certaines séquences sont soumises à des droits d'accès où il sera demandé à l'utilisateur d'accepter les « accords de confidentialité » avant de pouvoir les télécharger et d'en respecter leurs conditions d'utilisation. Ces accords, où la provenance des séquences y est stipulée, sont bien entendu présentés à titre informatif, leur acceptation (via une case à cocher) permet de s'assurer au minimum que l'utilisateur en a bien eu connaissance (Figure 213).


**Figure 213 - Droit d'accès et accord de confidentialité d'une séquence sur le site intranet du server SAM**

### b) *Les informations disponibles sur la séquence*

Une fois qu'une séquence a été retrouvée, et après acceptation des accords de confidentialité si il y a lieu, sont présentées sur une page distincte toutes les informations disponibles et utiles sur cette séquence (Figure 214) :

- Son identifiant (afin de pouvoir y accéder plus rapidement une prochaine fois, ou de l'ajouter dans une liste) ;
- Sa date d'ajout sur le serveur ;
- La personne qui a ajouté cette séquence, qui est potentiellement celle qui a effectué l'acquisition et qui pourra répondre aux éventuelles questions sur cette séquence (par exemple si elle dispose de fichiers de calibration non présents sur le serveur,…) ;
- Sa date et son heure d'acquisition (ce qui peut être utile pour en déduire des informations climatiques, par exemple un soleil rasant de crépuscule) ;
- La taille physique de la séquence (en octet), qui peut être utile avant le téléchargement, les séquences pouvant atteindre quelques giga-octets ;
- sa durée (utile pour pouvoir discriminer, si l'on recherche une séquence de plusieurs dizaines de minutes et non de quelques secondes par exemple)
- Sa description, renseignée par l'auteur de cette séquence, et proposant des informations complémentaires aux attributs.

De plus, si une séquence possède une ou plusieurs acquisitions vidéo, celles-ci pourront être rejouées en ligne (directement depuis le site), en une taille d'image réduite, permettant de vérifier facilement si cette séquence est bien celle recherchée (Figure 214).


Figure 214 - Les informations relatives à une séquence sur sa page dédiée du site intranet du server SAM

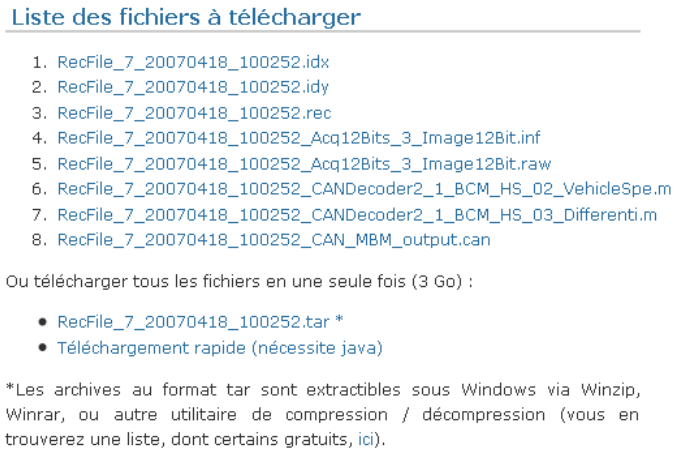
### c) *Le téléchargement*

Afin de récupérer les fichiers d'une séquence, trois modes de téléchargement sont proposés (Figure 215) :

1. Fichier par fichier (transfert HTTP): les fichiers constituant la séquence sont listés et il est possible de ne télécharger seulement que celui voulu.

2. Sous forme d'archive au format tar (transfert HTTP) : pour télécharger tous les fichiers en une seule fois. Un fichier *tar* n'est pas compressé, c'est simplement une concaténation de tous les fichiers en un seul. La taille de cette archive pouvant donc être assez conséquente, celle-ci est rappelée afin de vérifier l'espace disque disponible avant le téléchargement.

3. Via une page de téléchargement dédiée (transfert FTP) : les deux précédents modes de téléchargement utilisent le transfert via le protocole HTTP. Celui-ci, pour des raisons techniques pour le transfert de volumes importants, est plus lent que le protocole FTP. En effet, en HTTP, les données sont encapsulées et la réception des paquets est vérifiées, alors qu'en FTP, seul l'envoi des données est effectué.


Liste des fichiers à télécharger

1. RecFile_7_20070418_100252.idx
2. RecFile_7_20070418_100252.idy
3. RecFile_7_20070418_100252.rec
4. RecFile_7_20070418_100252_Acq12Bits_3_Image12Bit.inf
5. RecFile_7_20070418_100252_Acq12Bits_3_Image12Bit.raw
6. RecFile_7_20070418_100252_CANDecoder2_1_BCM_HS_02_VehicleSpe.m
7. RecFile_7_20070418_100252_CANDecoder2_1_BCM_HS_03_Differenti.m
8. RecFile_7_20070418_100252_CAN_MBM_output.can

Ou télécharger tous les fichiers en une seule fois (3 Go) :

- RecFile_7_20070418_100252.tar *
- Téléchargement rapide (nécessite java)

*Les archives au format tar sont extractibles sous Windows via Winzip, Winrar, ou autre utilitaire de compression / décompression (vous en trouverez une liste, dont certains gratuits, ici).

**Figure 215 - Téléchargement des fichiers d'une séquence sur le site intranet du server SAM**

## 5.4.4. Ajouter une séquence

### *a) Etiqueter et décrire une séquence*

Beaucoup d'informations peuvent être automatiquement déduites des fichiers qui seront ajoutés sur le serveur (taille, durée,…), mais de nombreuses autres doivent être correctement renseignées par l'utilisateur. Ainsi, cette étape est la plus critique pour le bon fonctionnent du serveur SAM (i.e. pouvoir rechercher une séquence efficacement sur le serveur).

A l'instar du formulaire de recherche, le formulaire de description d'une séquence propose exactement les mêmes zones à renseigner : attributs, conditions climatiques, et description.

### *b) Spécifier le niveau de confidentialité*

Pour garantir un accès plus ou moins restreint à une séquence, il est nécessaire de lui définir son niveau de confidentialité lors de son ajout sur le serveur, qui pourra être :

- Aucune
  La séquence pourra être partagée avec tout le monde (y compris avec nos divers partenaires industriels)

- Utilisation interne seulement
  La séquence ne pourra être utilisée que pour une recherche interne au laboratoire et avec le partenaire industriel spécifié.

### c) *Ajout des fichiers*

L'ajout des fichiers sur le serveur ne se fait ici que via le protocole FTP (le protocole HTTP ne supportant pas l'envoi de fichier volumineux car n'étant pas conçu pour). Une page dédiée contenant son logiciel de transfert (un applet Java), s'occupe de façon transparente pour l'utilisateur de la connexion et du transfert des fichiers vers le serveur SAM. Là aussi l'utilisation du protocole FTP permet d'atteindre des vitesses de transfert assez rapide (Figure 216 – transfert à 9Mo/s).

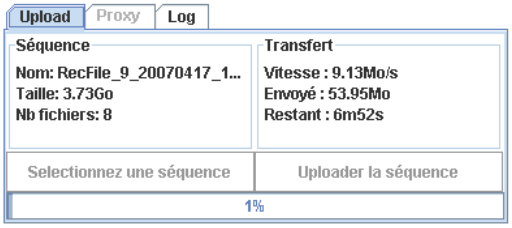



Figure 216 - Transfert d'une séquence sur le serveur SAM

Une fois le transfert terminé, le serveur enregistre toutes les informations fournies par l'utilisateur (description, attributs,…) et affiche sur une nouvelle page l'identifiant unique de cette séquence.

## 5.4.5. D'un point de vue plus technique

Le serveur (i.e. la plateforme logicielle) et le site intranet ont été conçus sur mesure pour les besoins de SAM et pour l'intégration dans le réseau informatique existant du laboratoire.

Il existait déjà un serveur web (i.e. une machine avec un logiciel serveur HTTP) pour l'hébergement du site internet du laboratoire. Cependant ce serveur n'a pas été conçu pour être un serveur de stockage massif de données. Ainsi une nouvelle machine a été acquise, comportant 4 disques durs de 1To chacun, montés en RAID (technologie qui permet entre autre une plus grande tolérance aux pannes via le stockage redondant des données sur plusieurs disques).

Comme le résume la Figure 217, le serveur SAM est donc en fait constitué de deux ordinateurs distincts : l'ancien serveur web où est hébergé le site intranet et le nouveau serveur de stockage où sont enregistrées les séquences $^{rt}$Maps. Le dialogue (i.e. le transfert des données) entre ces deux machines se faisant via des communications dédiées décrites dans les paragraphes suivants et n'est diffusé qu'à un utilisateur authentifié sur le site intranet (login + mot de passe) que lors de l'ajout ou du téléchargement d'une séquence par celui-ci. Un quelconque vol du mot de passe FTP n'est alors que peu utile car celui-ci ne sera plus valide quelques secondes après.

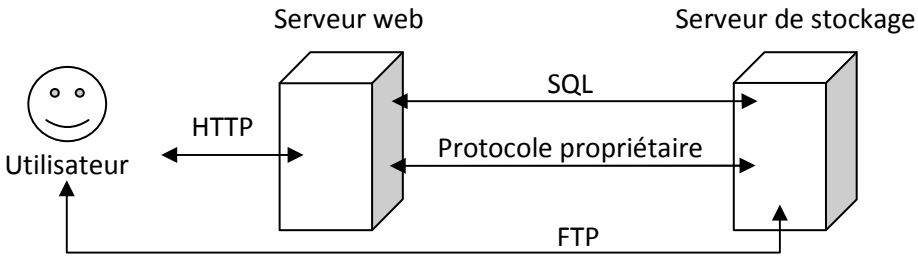



Figure 217 - Fonctionnant du serveur SAM

Tout d'abord, sur le serveur de stockage est mis en place un serveur de transfert de fichier FTP (logiciel externe développé par des tiers) pour assurer le transfert des séquences (ajout et téléchargement) depuis ou vers le serveur. Afin de garantir un niveau de sécurité optimal, le mot de passe du compte FTP est changé automatiquement et aléatoirement toutes les minutes.

Une base de données MySQL avec serveur d'accès pour y effectuer des requêtes est aussi mise en place sur le serveur de stockage. Cette base de données sert à stocker les attributs des séquences renseignées par les utilisateurs, ainsi que les informations des utilisateurs (nom, prénom, login, mot de passe crypté, adresse mail,…).

Ensuite la communication entre le serveur web (plus spécifiquement avec le site web) et le serveur de stockage est établie via un protocole propriétaire spécifiquement mis en place. Un serveur (logiciel) dédié a donc été développé (en C). Ce protocole / serveur permet d'effectuer des recherches physiques sur le serveur de stockage comme des recherches sur les noms de fichiers, sur les tailles,… mais permet aussi d'extraire et de transférer les images (après réduction de taille) des acquisitions vidéo des séquences qui pourront ensuite être visualisées sur le site web sous forme de vidéo ; et de créer à la volée, les archives au format *tar*. En effet, ces archives ne sont jamais stockées physiquement sur le serveur (cela doublerait l'espace disque nécessaire pour l'archivage des séquences) mais ne sont générées et transférées sur le réseau qu'à la demande d'un utilisateur.

Le site intranet a lui aussi été développé sur mesure pour répondre au plus près aux besoins requis par SAM. Ainsi, une première couche logicielle en PHP permet de communiquer avec le serveur de stockage soit via des requêtes SQL, soit via le protocole spécifiquement mis en place et de générer dynamiquement les pages web du site intranet. Cela permet par exemple de gérer une connexion sécurisée par mot de passe des utilisateurs avant qu'il puisse avoir accès aux données et avant de pouvoir transférer des séquences.

## 5.5. Perspectives

L'ensemble des outils présentés ici est maintenant couramment utilisé au sein du laboratoire, gage d'une réelle utilité pour les travaux de recherche de chacun. Certains sont aussi utilisés par nos partenaires industriels (Valeo,…) et au sein de projets collaboratifs nationaux (LoVE,…) regroupant d'autres laboratoire de recherche ainsi que d'autres industriels du milieu automobile.

Ces outils sont proposés, en interne au laboratoire, sur un serveur de gestion de version des codes source (serveur Subversion - SVN). Ceci permet aux autres doctorants ou chercheurs du laboratoire, de pouvoir les récupérer et les utiliser directement ; mais aussi de pouvoir les modifier et les améliorer ; puis de mettre à jour l'ensemble du code source sur le serveur. Les modifications seront alors « automatiquement » répercutées auprès des autres membres du laboratoire qui bénéficieront des améliorations apportés par chacun.

Plusieurs doctorants du laboratoire utilisent activement ces logiciels, et notamment ceux du groupe ADAS, qui y ont déjà apporté quelques améliorations (code Adaboost pour LeViS, génération d'un fichier de statistiques plus élaborés dans SamGT,…), gage d'un réel engouement pour ces outils et de leur utilité.

Enfin, la mise en commun des codes sources Java des logiciels LeViS et SamGT permet d'utiliser les même codes élémentaires (lectures des vidéo des séquences $^{rt}$Maps par exemple) pour chacun des logiciels, sans les dupliquer ; mais aussi a permis de réaliser des fonctionnalités plus avancées, comme par exemple l'extraction automatique des parties d'image représentatives de classes négatives (par comparaison avec une fichier de vérité terrain de SamGT) pour la création de bases d'apprentissage dynamiques dans LeViS utile pour certains algorithmes (Adaboost Cascadé,…).

# Conclusion et perspectives

L'un des enjeux majeurs actuels de sécurité routière, en vue notamment de réduire les accidents et le nombre de tués sur les routes, est le respect des limitations de vitesse imposées. Pour cela, les pouvoirs publics mettent en place depuis quelques années un système de contrôle automatisé des vitesses des véhicules (radar) incitant les automobilistes à respecter les limitations de vitesse. Cependant, il peut arriver que, pour diverses raisons, le conducteur n'ait pas vu la limitation de vitesse actuelle ou qu'il ne se rende pas compte du fait qu'il roule trop vite ou trop peu vite.

Ainsi, les professionnels de l'industrie de l'automobile tentent d'intégrer dans leurs véhicules des systèmes détectant et reconnaissant ces limitations de vitesse, pouvant ainsi proposer une aide à la conduite et prévenir le conducteur en cas de faute afin qu'il régularise sa vitesse.

Les travaux qui ont été présentés dans ce document visent justement à l'élaboration d'un tel système d'aide à la conduite automobile.

Les sources d'informations disponibles pour ce faire sont uniquement celles provenant de l'utilisation d'un capteur visuel et / ou d'un capteur cartographique. Nous avons vu qu'un capteur cartographique dispose uniquement d'une connaissance statique relative aux informations des routes (dont les limitations de vitesse de celles-ci) et ne peut donc notamment pas rendre compte des informations dites dynamiques, telles que les zones temporaires de travaux qui sont généralement soumises à une nouvelle limitation de vitesse provisoire, ou encore les limites variables sur autoroute. Quant à lui, un capteur visuel peut souffrir de non détection des panneaux de limitation de vitesse en cas d'occlusion, ou d'un manque d'interprétation des détections (cas par exemple des détections des panneaux des voies de sortie).

C'est donc tout naturellement que le système présenté dans ces travaux se base sur une détection visuelle des limitations de vitesse et ce pour pouvoir connaitre les limitations temporaires qui pourraient ne pas être prises en compte dues aux habitudes des automobilistes par exemple,

Il a été prouvé dans le chapitre 1, que ce système de détection visuel fonctionne correctement (94% de taux de bonne détection avec une seule fausse détection sur notre jeu de test), mais qu'à lui seul il ne peut prendre de décision quant à savoir si un panneau détecté est réellement applicable au véhicule embarquant le système. C'est pourquoi, un système de fusion a été proposé reposant sur d'autres sources d'information et notamment sur un capteur cartographique (chapitre 2) et une détection visuelle des lignes de marquage au sol (chapitre 3). Ce système complet a été présenté dans le chapitre 4 de ce mémoire.

L'ensemble des approches jusque là proposées dans la littérature utilisant des techniques de fusion de données afin de pallier aux problèmes de ces deux capteurs (visuel et cartographique) s'est focalisé, comme stipulé dans le chapitre 2, à essayer de fusionner les valeurs probables des limitations de vitesses fournies par ces différents capteurs par l'utilisation des théories de fusion

probabiliste, tel que la théorie des croyances. Or cette fusion ne peut être la réponse à un système ADAS évolué, car il existe des situations où les deux capteurs risquent de fournir la même information erronée, et d'autres où la fusion amène dans un état de doute où il ne serait possible de prendre une décision (notamment quand les deux capteurs ne sont pas en adéquation). Ce constat de ne pas entrer dans une situation d'incertitude avait déjà été émis en 1833 par Honoré de Balzac qui, dans son livre *L'illustre Gaudissart*, avait notamment écrit :

*« Choisir! c'est l'éclair de l'intelligence. Hésitez-vous?.. tout est dit, vous vous trompez. »*

Ainsi, comme il a été démontré dans le chapitre 4, la fusion doit être utilisée pour améliorer l'**interprétation** de l'environnement en considérant que les sources disponibles sont complémentaires et non concurrentes. C'est cette interprétation qui permet de clarifier certains cas et de ne pas *hésiter* lors d'une prise de décision par le système.

Ce processus de fusion par complémentarité est ici appliqué à la détection des limitations de vitesse via une fusion à base de règles, mais il est clair que n'importe quelle détection ne peut pas se suffire à elle-même. Il est donc impératif d'interpréter l'environnement routier pour savoir si cette détection est finalement cohérente et doit être prise en compte pour le système ADAS développé.

Ainsi, sans cette interprétation, qui ne peut se faire sans fusion multisources, comme le ferait un être humain avec ses sens (sa vue, son ouïe,…) mais ici avec les sens du véhicule (télémètre laser, caméra, capteur proprioceptifs,…), il ne peut être envisageable de réaliser de systèmes ADAS réellement « avancés ». Or les systèmes ADAS ne sont ils pas le fondement des futurs véhicules intelligents ?

Cependant, le système mis en place ici n'est pas exempt de défaut. Par exemple, il faudrait s'assurer que le jeu de règles établi dans le processus de fusion sur le cas prototypique présenté dans ces travaux est suffisant pour couvrir tous les cas de figure pouvant survenir dans la détection des limitations de vitesse.

De plus, afin de robustifier le système dans sa globalité, il faudrait améliorer les résultats des algorithmes de détection issus de chaque source d'information :

- Améliorer la détection visuelle des panneaux de limitation de vitesse en prenant compte tous les types de panneaux (par exemple les zones et les fins de zone 30 situées en ville), parfaire la détection des fins de limitation, et peut être envisager de nouvelles approches pour s'affranchir des rares problèmes des rotations et des multiples résolutions des panneaux (comme utiliser la transformée de Fourrier-Mellin).

- Améliorer la détection des panonceaux par la reconnaissance de multiples panonceaux autres que seulement celui de sortie de voie.

- Améliorer la détection des lignes de marquage au sol, afin d'essayer de compter le nombre de voie sur la route et d'estimer la position du véhicule par rapport à ces voies, ce qui pourrait confirmer le passage du véhicule sur la voie de sortie.

Sur ce dernier point, il pourrait être envisagé d'utiliser les informations d'autres algorithmes, comme ceux des détections des autres véhicules (par lidar, radar ou par caméra) afin d'estimer et de compter le nombre de voies aux alentours de notre véhicule.

Enfin, plus généralement, il pourrait être envisagé de développer un système expert générique à plusieurs niveaux dédié à l'élaboration d'un système ADAS plus poussé afin de faciliter l'établissement et l'intégration de règle de conduite automobile.

Certaines pistes et d'autres encore sont en cours d'étude au Centre de Robotique. C'est ainsi qu'un nouveau projet a été créé afin de continuer ces travaux de recherche dans la détermination de la limitation de vitesse. Les défis majeurs de ce projet, du nom de SPEEDCAM, subventionné par l'ANR (Agence Nationale de la Recherche) et en collaboration avec divers partenaires industriels (Valeo et Daimler AG), portent d'une part sur l'amélioration de la détection des panonceaux et surtout sur le développement d'une technique de fusion robuste entre les capteurs cartographique et visuel. La collaboration réussie avec Valeo dans le cadre des travaux présentés ici, et le renouvellement de cette coopération dans le cadre du projet SPEEDCAM, a amené l'établissement au sein du laboratoire d'un nouveau sujet de thèse portant sur la suite et sur l'amélioration des travaux présentés dans ce mémoire.

Enfin, les derniers problèmes à répondre dans le futur avant de voir l'intégration d'un tel système interagissant directement avec le conducteur sur la vitesse de sa conduite sont ceux d'ordre juridiques, comme la question de la responsabilité en cas de la défaillance du système. Ces problèmes sont des questions récurrentes pour tous les systèmes ADAS présents et futurs.

# Glossaire

| | |
|---|---|
| **ACC** | Advanced Cruise Control |
| **ADAS** | Advanced Driving Assistance System |
| **API** | Application Programming Interface |
| **CNN** | Convolutional Neural Networks |
| **DGPS** | Differential GPS |
| **DSS** | Driver Support System |
| **DT** | Distance Transform |
| **ENSMP** | Ecole Nationale Supérieure des Mines de Paris |
| **ERSO** | European Road Safety Observatory |
| **FOM** | Figure Of Merit |
| **GDF** | Geographic Data Files |
| **GPS** | Global Positioning System |
| **GS-SLS** | Global Segmentation for Speed-Limit Sign |
| **GT** | Ground Truth |
| **HSI** | Hue Saturation Intensity |
| **HSL** | Hue Saturation Lightness |
| **HSV** | Hue Saturation Value |
| **IGN** | Institut Géographique National |
| **INRIA** | Institut National de Recherche en Informatique et Automatique |
| **INSPIRE** | Infrastructure for Spatial Information in the European Community |
| **IPM** | Inverse Perspective Mapping |
| **ISO** | International Organization for Standardization |
| **ITS** | Intelligent Transportation Systems |
| **kNN** | k-Nearest Neighbor |
| **LARA** | LA Route Automatisée |

| | |
|---|---|
| **LDS** | Lane Departure System |
| **LED** | Light-Emitting Diode |
| **MLP** | MultiLayer Perceptrons |
| **MMS** | Mobile Mapping System |
| **MNS** | Modèle Numérique de Surface |
| **NHTSA** | National Highway Traffic Safety Administration |
| **OEM** | Original Equipment Manufacturer |
| **ONISR** | Observatoire National Interministériel de Sécurité Routière |
| **PMC** | Perceptron MultiCouche |
| **POI** | Point Of Interest |
| **RAID** | Redundant Array of Inexpensive Disks |
| **RANSAC** | RANdom SAmple Consensus |
| **RBF** | Radial Basis Function |
| **RFID** | Radio-Frequency Identification |
| **RMI** | Remote Method Invocation |
| **ROI** | Region Of Interest |
| **$^{RT}$MAPS** | Real Time, Multisensor, Advanced Prototyping Software |
| **SAM** | Serveur d'Acquisition Maps |
| **SIG** | Système d'Information Géographique |
| **SDK** | Software Development Kit |
| **SLA** | Speed Limit Assist |
| **SLS** | Speed Limit Support |
| **SQL** | Structured Query Language |
| **SVM** | Support Vector Machine |
| **SVN** | Subversion (version control system) |
| **TIGER** | Topologically Integrated Geographic Encoding and Referencing |
| **TSR** | Traffic Sign Recognition |
| **UTM** | Universal Transverse Mercator |

# Publications

**Bargeton A., Moutarde F., Nashashibi F., Bradai B. et Chanussot L.** Towards a reliable speed-limit assistant combining Traffic Sign Recognition with GPS information [Workshop] // British Machine Vision Association (BMVA) technical meeting on "Vision for Automotive Applications" held in London, UK, 20th May 2009.

**Bargeton A., Moutarde F., Nashashibi F., Bradai B. et Chanussot L.** "Traffic Sign Recognition and fusion with navigation for reliable determination of current speed-limit", , SIA congress on "Vehicle and Infrastructure Safety Improvement in adverse cONditions and night driving" (V.I.S.I.O.N.'2008), held in Satory (France), 7-8 oct 2008.

**Bargeton A., Moutarde F., Nashashibi F. et Bradai B.** Improving pan-European speed-limit signs recognition with a new “global number segmentation” before digit recognition [Article] // Proc. 2008 IEEE Intelligent Vehicles Symposium (IV08), Eindhoven, Netherlands, June 4–6, 2008.

**Hamdoun O., Bargeton A., Moutarde F., Bradai B. et L. Chanussot** Detection and recognition of end-speed-limit and supplementary signs for improved European speed limit support [Article] // Proc. of 15th World congress on Intelligent Transportation Systems (ITSWC), New York, USA, November 16-20, 2008.

**Moutarde F., Bargeton A., Herbin A. et Chanussot L.** Modular Traffic Sign Recognition applied to on-vehicle real-time visual detection of American and European speed limit signs [Article] // Proc. of 14th World congress on Intelligent Transportation Systems (ITSWC), Beijing, China, October 9-13, 2007

**Moutarde F., Bargeton A., Herbin A. et Chanussot L.** Robust on-vehicle real-time visual detection of American and European speed limit signs, with a modular Traffic Signs Recognition system [Article] // Proc. 2007 IEEE Intelligent Vehicles Symposium (IV07), Istanbul, Turkey, June 13-15, 2007.

**Nashashibi F. et Bargeton A.** Laser-based vehicles tracking and classification using occlusion reasoning and confidence estimation [Article] // Proc. 2008 IEEE Intelligent Vehicles Symposium (IV08), Eindhoven, Netherlands, June 4–6, 2008.

# Bibliographie

**Abramson Y.** AdaBoost/GA et filtrage particulaire: La vision par ordinateur au service de la sécurité routière. [Rapport]. - [s.l.] : PhD thesis Informatique-Robotique, CAOR - Centre de robotique, ENSMP, 2005.

**Abramson Y., Steux B. et Ghorayeb H.** YEF (Yet Even Faster) Real-Time Object Detection [Article] // Proceedings of International Workshop on Automatic Learning and Real-Time. - 2005.

**Abuhadrous I.** Système embarqué temps réel de localisation et de modélisation 3D par fusion mutli-capteur [Rapport] : Thèse. - [s.l.] : Ecole des Mines de Paris, 2005.

**Adam S. [et al.]** Utilisation de la Transformée de Fourier-Mellin pour la reconnaissance de formes multi-orientées et multi-échelles : application à l'analyse automatique de documents techniques [Article] // Revue Traitement du Signal. - 2001. - 1 : Vol. 18.

**Adjaoute A.** Rylm : generateur de systemes experts pour les problemes d'aide aux diagnostics. ykra : systeme d'enseignement [Revue]. - [s.l.] : Thèse : Doctorat D'etat : Sciences Et Techniques, Paris 6, 1988.

**Adorni G. [et al.]** Shape searching in real world images: a CNN-based approach [Article] // IEEE Fourth Workshop on Cellular Neural Networks and their Applications. - 1996.

**Ajdari Rad A., Faez K. et Qaragozlou N.** Fast Circle Detection Using Gradient Pair Vectors [Article] // Proceding of the 7th Digital Image Computing: Techniques and Applications. - 2003.

**Aly M.** Real time Detection of Lane Markers in Urban Streets [Revue]. - [s.l.] : IEEE Intelligent Vehicles Symposium, 2008.

**Aoyagi Y et Asakura T** A Study on Traffic Sign Recognition in Scene Image using Genetic Algorithms and Neural Networks [Article] // IEEE IECON International Conference on Industrial Electronics, Control, and Instrumentation. - 1996.

**Arnoul P. [et al.]** Traffic signs localisation for highways inventory from a video camera on board amoving collection van [Article] // Proceedings of the IEEE Intelligent Vehicles Symposium. - 1996.

**Bahlmann C. [et al.]** A system for traffic sign detection, tracking, and recognition using color, shape, and motion information [Article] // in Proceedings. IEEE Intelligent Vehicles Symposium. - 2005.

**Bahlmann C. [et al.]** Mutli-modal speed limit assistants : Combining camera and gps maps [Article] // IEEE Intelligent Vehicles Symposium. - 2008.

**Bargeton A.** Architecture logicielle générique et fusion de données pour le développement d'un système temps réel d'assistance à la conduite automobile [Rapport]. - [s.l.] : Stage d'ingénierie, Centre de Robotique, Ecole des Mines de Paris, 2005.

**Barnes N. et Zelinsky A.** Real-time radial symmetry for speed sign detection [Article] // IEEE Intelligent Vehicles Symposium. - 2004.

**Bastiaensen E.** The Electronic Horizon - In-vehicle digital maps state of art & needs [Rapport]. - [s.l.] : ERTICO, 2004.

**Bertozzi M. [et al.]** An Evolutionary Approach to Lane Markings Detection in Road Environments [Revue] // In Atti del 6 Convegno dell'Associazione Italiana per l'Intelligenza Artificiale. - 2002.

**Bertozzi M. et Broggi A.** GOLD: A Parallel Real-Time Stereo Vision System for Generic Obstacle and Lane Detection [Revue] // Croute. - [s.l.] : IEEE Trans. Image Processing, 1997.

**Betke Margrit et C. Makris Nicholas** Fast Object Recognition in Noisy Images Using Simulated Annealing [Article] // In Proceedings of the Fifth International Conference on Computer Vision. - Cambridge : [s.n.], 1994.

**Bhatia P.** Vehicle Technologies to Improve Performance and Safety [Rapport]. - [s.l.] : University of California Transportation Center (UCTC), 2003.

**Bloch I.** Fusion d'informations en traitement du signal et des images [Livre]. - [s.l.] : Lavoisier, Hermes Science Publication, 2003.

**Borgefors G.** Distance Transformations in Digital Images [Article] // Computer Vision, Graphics and Image Processing. - 1986.

**Bradai B.** Optimisation des Lois de Commande d'Éclairage Automobile par Fusion de Données [Rapport]. - [s.l.] : Thèse, Ecole Doctorale Multidsciplinaire Jean Henri Lambert Sciences Dures , 2007.

**Broggi A. [et al.]** Real Time Road Signs Recognition [Article] // IEEE Intelligent Vehicles Symposium. - 2007.

**Brun X.** Modélisation 3D texturée en temps réel d'environnements urbains et routiers, et application au calcul de distance de visibilité routière. [Rapport]. - [s.l.] : Centre de Robotique, Ecole des Mines de Paris, 2007.

**Brunel H.** Etat de l'art des systèmes d'information géographique [Rapport] : Mémoire d'ingénieur. - [s.l.] : CNAM, 2005.

**Caraffi C. [et al.]** An algorithm for Italian de-restriction signs detection [Article] // IEEE Intelligent Vehicles Symposium. - 2008.

**Casey R.G. et Lecolinet E.** A Survey of Methods and Strategies in Character Segmentation [Article] // IEEE Transactions on Pattern Analysis and Machine Intelligence. - 1996.

**Casey R.G. et Lecolinet E.** Strategies in character segmentation: a survey [Article] // Proceedings of the Third International Conference on Document Analysis and Recognition. - 1995.

**Cecotti H., Choisy C. et Belaid A.** Réseau de neurones à topologie dynamique, comparaison avec des invariants pour la reconnaissance de caractères multi-orientés multi-échelles [Article] // Huitième Colloque International Francophone sur l'Ecrit et le Document. - 2004.

**Chapelon J.** La sécurité routière en France, Bilan de l'année 2008 [Rapport]. - [s.l.] : Observatoire national interministériel de sécurité routière (ONISR), 2009.

**Cheng H.Y. [et al.]** Lane Detection With Moving Vehicles in the Traffic Scenes [Revue] // IEEE Transactions on Intelligent Transportation. - 2006.

**Cheng W. [et al.]** Automatic road vector extraction for mobile mapping systems [Revue] // International Society for Photogrammetry and Remote Sensing. - 2008.

**Cybenko G.V.** Approximation by Superpositions of a Sigmoidal function [Article] // Mathematics of Control, Signals and Systems. - 1989.

**Daimler** Lane Keeping Assist: Always on the right track [En ligne]. - 16 Juillet 2009. - http://www.daimler.com/dccom/0-5-1210220-1-1210351-1-0-0-1210338-0-0-135-7165-0-0-0-0-0-0-0.html.

**Daimler** PreVENT - Final report [Rapport]. - [s.l.] : ERTICO, 2008.

**de la Escalera A. [et al.]** Road Traffic Sign Detection and Classification [Article] // IEEE Transaction on Industrial Electronics. - 1997. - 6 : Vol. 44.

**Dessalle P.** Conception et réalisation d'une plateforme de déploiement de services géolocalisés [Rapport] : Mémoire d'ingénieurie. - Bruxelles : Université Libre de Bruxelles, 2006.

**Dickmanns E.D. et Mysliwetz B.D.** Recursive 3-D road and relative ego-state recognition [Revue] // IEEE Trans. Pattern Anal. Mach. Intell.. - 1992.

**Doermann D. et Mihalcik D.** Tools and techniques for video performance evaluation. [Revue]. - [s.l.] : In International Conference on Pattern Recognition (ICPR), 2000. - Vol. 4.

**ERSO** Annual Statistical Report 2006, based on data from CARE / EC [Rapport]. - [s.l.] : European Road Safety Observatory, 2007.

ERTICO [En ligne] // ERTICO. - 01 10 2009. - http://www.ertico.com.

**ERTICO [et al.]** ActMAP Final Report [Rapport]. - [s.l.] : Ertico, 2005.

**ERTICO** ERTICO [En ligne] // ERTICO. - ERTICO, 01 10 2009. - http://www.ertico.com.

**Escalera A., Armingol J.M. et Mata M.** Traffic sign recognition and analysis for intelligent vehicles [Article] // Image and Vision Computing. - 2003. - Vol. 21.

**Escalera S. et Radeva P.** Fast Greyscale road sign model matching and recognition [Article] // Frontiers in Artificial Intelligence and Applications / Recent Advances in Artificial intelligence Research and Development. - 2004.

**Fallou O. [et al.]** Comité de coordination du projet RODRIGUE - Bilan des actions - Rapport final [Rapport]. - [s.l.] : Erdyn, LCPC, 2008.

**Fang C.Y. [et al.]** A road sign recognition system based on dynamic visual model [Article] // in Proc IEEE Conf. on Computer Vision and Pattern Recognition. - 2003.

**Fang C.Y., Chen S.W. et Fuh C.S.** Road-sign detection and tracking [Article] // IEEE Trans. Vehicular Technology. - 2003.

**Freund Y. et Schapire E.** A short introduction to boosting. [Revue]. - [s.l.] : Journal of Japanese Society for Artificial Intelligence, 1999.

**Garcia M.A., Sotelo M.A. et Martin-Gorostiza E.** Fast Traffic Sign Detection and Recognition Under Changing Lighting Conditions [Article] // Proceedings of the IEEE ITSC 2006, 2006 IEEE Intelligent Transportation Systems Conference, Toronto. - 2006.

**Gavrila D.M.** Traffic sign Recognition Revisited [Article] // Proceding of the 21st DAGM Symposium fur Musterekennung. - 1999.

**Gehrig S., Wagner S. et Franke U.** System architecture for an intersection assistant fusing image, map and GPS information [Article] // IEEE Intelligent Vehicles Symposium. - 2003.

**Ghorayeb H.** Conception et mise en oeuvre d'algorithmes de vision temps-réel pour la vidéo surveillance intelligente [Rapport]. - [s.l.] : Doctorat Informatique temps réel, robotique et automatique, CAOR- Centre de robotique, ENSMP, 2007.

**Gopalan R. [et al.]** Video-based lane detecton using Boosting principles [Revue] // Snowbird Workshop. - 2009.

**Guibert L. [et al.]** On-board optical joint transform correlator for real-time road sign recognition [Revue]. - [s.l.] : Optical-Engineering, 1995. - 1 : Vol. 34.

**Guibert L.** Etude et réalisation d'un corrélateur a cristaux liquides ferroélectriques. application a la détection automatique de panneaux routier [Rapport]. - [s.l.] : Université de Mulhouse, 1995.

**Herbin Anne, Chanussot Lowik et Moutarde Fabien** Procédé de détection d'un objet cible [Brevet] : FR2917869 : Logiciel / éd. INPI. - France, 25 06 2007.

**Hetrick A. et Virost S.** Lane Detection for Autonomous Vehicles [Revue] // Spring. - 2007.

**Hibi T.** Vision based extraction and recognition of road sign region from natural colour image by using HSL and coordinates transformation [Article] // 29th ISATA Int conf . - 1996.

**Horache E.H.** Corrélation optique multiplexe basée sur la modulation spatiale de cohérence d'une source à spectre large: application à la reconnaissance des formes [Rapport]. - [s.l.] : Université de Marnes La Vallée, 2001.

**Hsien J.C. et Chen S.Y.** Road Sign Detection and Recognition Using markov Model [Article] // The 14th Workshop on OOTA. - 2003.

**Hsu S.H. et Huang C.L.** Road sign detection and recognition using matching pursuit method [Article] // Image and Vision Computing. - 2001.

**Hu M.K.** Visual pattern recognition by moment invariants [Article] // IRE Trans. on Information Theory. - 1962.

**Isard M. et Blake A.** CONDENSATION – conditional density propagation for visual tracking [Revue]. - [s.l.] : International Journal of Computer Vision , 1998.

**Jabbour M., Bonnifait P. et Cherfaoui V.** Map-Matching Integrity Using Multihypothesis Road-Tracking [Revue]. - [s.l.] : Journal of Intelligent Transportation Systems, 2008. - 4 : Vol. 12.

**Janssen H. et Niehsen W.** Vehicle surround sensing based on information fusion of monocular video and digital map [Article] // IEEE Intelligent Vehicles Symposium. - 2004.

**Janssen R. [et al.]** Hybrid approach for traffic sign recognition [Article] // Proceding of Intelligent Vehicles Conference. - 1993.

**Jaynes C. [et al.]** An open development environment for evaluation of video surveillance systems [Revue]. - [s.l.] : In IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS), 2002.

**Kang D.S., Griswold N.C. et Kehtarnavaz N.** An invariant traffic sign recognition system based on sequential color processing and geometrical transformation [Article] // Proceedings of the IEEE Southwest Symposium on Image Analysis and Interpretation. - 1994.

**Khammari A.** Système embarqué de détection multi-sensorielle de véhicules : application à la gestion intelligente des interdistances [Rapport]. - [s.l.] : Thèse Centre de Robotique, Ecole des Mines de Paris, 2006.

**Khan S.** Character Segmentation Heuristics for Check Amount Verification [Article] // Thesis MIT. - 1998.

**Kim S.K. et Forsyth D.A.** A new approach for road sign detection and recognition algorithm [Article] // 30th International Symposium on Automotive Technology and Automation. Robotics, Motion and Machine Vision in the Automotive Industries. - 1997.

**Kim Z.W.** Robust Lane Detection and Tracking in Challenging Scenarios [Revue] // IEEE Transactions on Intelligent Transportation Systems. - 2008.

**Kipp M.** Anvil - A Generic Annotation Tool for Multimodal Dialogue [Revue]. - [s.l.] : In Proceedings of the 7th European Conference on Speech Communication and Technology (Eurospeech) , 2001.

**Klausner A., Rinner B. et Tengg A.** I-SENSE: Intelligent Embedded Multi-Sensor Fusion [Revue]. - [s.l.] : In Proceedings of the 4th IEEE International on Intelligent Solutions in Embedded Systems, 2006.

**Kluge K.C.** Extracting road curvature and orientation from image edge [Revue] // IEEE Intelligent Vehicles Symposium. - 1994.

**Kröse B. et van der Smagt P.** An introduction to Neural Networks [Livre]. - 1996. - Eighth edition.

**Lauffenburger J. Ph. [et al.]** Navigation as a virtual sensor for enhanced lighting preview control [Article] // IEEE Intelligent Vehicles Symposium. - 2007.

**Lauffenburger J.P. [et al.]** Navigation and Speed Signs Recognition Fusion for Enhanced Vehicle Location [Revue]. - [s.l.] : Proceedings of the 17th World Congress The International Federation of Automatic Control, 2008.

**Liang Y.M. [et al.]** Video Stabilization for a Camcorder Mounted on a Moving Vehicle [Revue]. - [s.l.] : IEEE Trans. on Vehicular Technology, 2004.

**LIM K.H. [et al.]** Lane-Vehicle Detection and Tracking [Revue]. - [s.l.] : Proceedings of the International MultiConference of Engineers and Computer Scientists, 2009.

**Linn R.J. et Hall D.L.** A Survey of Multi-sensor Data Fusion Systems [Revue]. - [s.l.] : Proceedings of the SPIE - The International Society for Optical Engineering., 1991.

**Liu C.H. et Fujisawa H.** Classification and Learning Methods for Character Recognition: Advances and Remaining Problems [Article] // Machine Learning in Document Analysis and Recognition. - 2008.

**Liu W. [et al.]** Vision based real time lane marking detection and tracking [Revue] // IEEE Intelligent Transportation Systems. - 2008.

**Loy G. et Barnes N.** Fast shape-based road sign detection for a driver assistance system [Article] // in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. - 2004.

**Loy G. et Zelinsky A.** Fast radial symmetry for detecting points of interest [Article] // IEEE Transactions on Pattern Analysis and Machine Intelligence. - 2003.

**Luo R.C., Potlapalli H. et Hislop D.** Neural network based landmark recognition for robot navigation [Article] // IEEE International Conference on Industrial Electronics, Control, Instrumentation, and Automation, Power Electronics and Motion Control. - 1992.

**Macek K. [et al.]** A lane detection vision module for driver assistance [Revue] // IEEE/APS Conference on Mechatronics and Robotics. - IEEE/APS Conference on Mechatronics and Robotics : [s.n.], 2004.

**Maldonado-Bascón S. [et al.]** Road-Sign Detection and Recognition Based on Support Vector Machines [Article] // IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS. - 2007.

**McCall J.C. et Trivedi M.M.** Video-Based lane estimation and tracking for driver assistance : survey, system, and evaluation [Revue] // IEEE transaction on intelligent transporation system. - 2006.

**Meir R. et Rätsch G.** An introduction to boosting and leveraging. [Article] // In Advanced Lectures on Machine Learning. - 2003.

**Meng Y. [et al.]** A simplified Map-Maptching Algorithm for In-Vehicle navigation unit [Article] // The association of chinese professionals in geographic information systems. - 2002.

**Miura J., Kanda T. et Shirai Y.** An Active Vision System for Real Time Traffic Sign Recognition [Article] // Proceeding 2000 IEEE Int. Conf. On Intelligent Transportation Systems. - 2000.

**Moutarde F., Stanciulescu B. et Breheret A.** Real-time visual detection of vehicles and pedestrians with new efficient adaBoost features [Article] // 2nd Workshop on Planning, Perception and Navigation for Intelligent Vehicles (PPNIV), IEEE International Conference on Intelligent RObots Systems. - 2008.

**Nedevschi S. [et al.]** 3D Lane Detection System Based on Stereovision [Revue] // IEEE Intelligent Transportation Systems Conference (ITSC). - 2004.

**Nienhüser D., Gumpp T. et Zöllner J.M.** A situation context aware Dempster-Shafer fusion of digital maps and a road sign recognition system [Article] // IEEE Intelligent Vehicles Symposium. - 2009.

**Pacheco L., Batlle J. et Cufi X.** A New Approach to Real Time Traffic Sign Recognition Based on Colour Information [Revue]. - [s.l.] : Proceedings of the Intelligent Vehicles Symposium, 1994.

**Paclik P. [et al.]** Road sign classification using Laplace kernel classifier [Article] // Pattern Recognition Letter. - 2000.

**Petillot Y.** Vers une implantation de corrélateurs optiques temps réel: applications a divers problèmes de reconnaissance des formes [Rapport]. - [s.l.] : Université de Brest, 1996.

**Piccioli G. [et al.]** Robust method for road sign detection and recognition [Article] // IVC. - 1996. - 3 : Vol. 14.

**Polychronopoulos A. [et al.]** Extended path prediction using camera and map data for lane keeping support [Article] // IEEE Intelligent Trasnportation System ITS. - 2005.

**Pomerleau D.** Ralph: Rapidly adapting lateral position handler [Revue] // In IEEE Symposium on Intelligent Vehicles. - 1995.

**Priese L. [et al.]** Traffic Sign Recognition Based on Color Image Evaluation [Article] // Proceding of Intelligent Vehicles Symposium. - 1993.

**Quddus M.A.** High Integrity Map Matching Algorithms for Advanced Transport Telematics Applications [Rapport] : Thesis. - Imperial College London : [s.n.], 2006.

**Revue A., Wybo S. et Nashashibi F.** Multisensor based collision risk assessment: a case of crossroads approach [Article] // IEEE Intelligent Vehicles Conference. - 2002.

**Robinson A.H. [et al.]** Elements of Cartography [Livre]. - [s.l.] : John Wiley and Sons, 1995.

**Rombaut M.** Fusion : état de l'art et persepectives [Rapport]. - [s.l.] : IUT de Troyes, laboratoire LM2S-UTT, 2001.

**Routière Sécurité** Les grandes données de l accidentologie, Caractéristiques et causes des accidents de la route [Rapport]. - [s.l.] : Direction de la sécurité et de la circulation routière, 2006.

**Routière1 Sécurité** Synthèse générale de l'année 2008 [Rapport]. - [s.l.] : Sécurité Routière, 2009.

**Routière2 Sécurité** Les grandes données de l'accidentologie [Rapport]. - [s.l.] : Sécurité Routière, 2009.

**Schapire E.** Boosting [En ligne]. - 5 Août 2009. - http://www.cs.princeton.edu/~schapire/boost.html.

**Seitz P. [et al.]** The robust recognition of traffic signs from a moving car [Article] // Proceding of th 13th DAGM Symposium on pattern recognition. - 1991.

**Simard P.Y., Steinkraus D. et Platt J.C.** Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis [Article] // Proceding of the 7th International Conference on Document Analysis and Recognition. - 2003.

**Ślot K. [et al.]** Road Lane Detection with Elimination of High-Curvature Edges [Revue] // Lecture Notes in Computer Science, Computer Vision and Graphics. - 2009. - pp. 33-42.

**Smith D. et Singh S.** Approaches to Multisensor Data Fusion in Target Tracking: A Survey [Revue]. - [s.l.] : Knowledge and Data Engineering, IEEE Transactions on In Knowledge and Data Engineering, 2006.

**Soetedjo A. et Yamada K.** Traffic sign classification using ring partitioned method [Article] // IEICE Trans. Fundamentals. - 2005.

**Southall B. et Taylor C.J.** Stochastic road shape estimation [Revue] // International Conference on Computer Vision. - 2001.

**Stanciulescu B., Breheret A. et Moutarde F.** Introducing New AdaBoost Features for Real-Time Vehicle Detection [Revue]. - [s.l.] : Proceedings of COGIS2007 Cognitive Systems with Interactive Sensors, 2007.

**Steux B.** RTMaps un environnement logiciel dédié à la conception d'applications embarquées temps-réel. Utilisation pour la détection automatique de véhicules par fusion radar / vision. [Rapport]. - [s.l.] : PhD Thesis, Centre de Robotique, Ecole des Mines de Paris, 2001.

**Subramanian R.** Motor vehicle traffic crashes as a leading cause of death in the United States, 2003 [Rapport]. - [s.l.] : Traffic Safety Facts Research Notes, Washington, D.C.: Department of Transportation, National Highway Safety Administration, 2006.

**Thomas C.M. et Featherstone W.E.** Validation of vincenty's formulas for the geodesic using a new fourth-order extension of kivioja's formula [Revue] // Journal of Surveying Engineering,. - 2005.

**Thomson M. et Westland S.** Colour-imager characterization by parametric fitting of sensor responses [Article] // Color Reserch and Application. - 2001.

**Tian M. [et al.]** Vision Based Lane detection for Active Security in Intelligent Vehicle [Revue]. - [s.l.] : IEEE Conference on Vehicular Electronics and Safety, 2006.

**Torresen J., Bakke J.W. et Sekanina L.** Efficient recognition of speed limit signs [Article] // in Proceedings The 7th International IEEE Conference on Intelligent Transportation Systems. - 2004.

**Travis Rose R.** MacVisSTA: A System for Multimodal Analysis of Human Communication and Interaction [Rapport]. - [s.l.] : Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University in partial ful Master of Science, 2007.

**Tsai L.W. [et al.]** Lane detection using directionnal random walks [Revue] // IEEE Intelligent Vehicles Symposium. - 2008.

**Vincenty T.** Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations [Revue] // Survey Review. - 1975.

**Viola P. et Jones M.** Robust Real-time Object Detection [Article] // Second International Workshop on Statistical and Computational Theories of Vision – Modeling, Learning, Computing and Sampling. - 2001.

**Wang Y., Teoh E.K. et Shen D.** Lane detection and tracking using B-snake [Revue] // Image Vision Comput. - 2004.

**White C.E., Bernstein D. et Kornhauser A.L.** Some map matching algorithms for personal navigation assistants [Revue] // Transportation Research Part. - 2000.

**Wikipedia** Lane departure warning system [En ligne]. - 16 Juillet 2009. - http://en.wikipedia.org/wiki/Lane_departure_warning_system.

**Wikipedia** Réseau de neurones [En ligne] // Wikipedia. - Wikipedia, 4 août 2009. - http://fr.wikipedia.org/wiki/Réseau_de_neurones.

**Wu X. et Shi P.** Unconstrained handwritten numeral recognition using Hausdorff distance and multi-layer neural network classifier [Article] // Proceedings of the Fifth International Conference on Document Analysis and Recognition. - 1999.

**Yang H.M., Liu C.L. et Huang S.M.** Traffic Sign Recognition in disturbing Environments [Article] // Proceedings of the Fourteenth International Symposium on Methodologies for Intelligent Systems. - 2003.

**Zadeh M.M., Kasvand T. et Suen C.Y.** Localization and recognition of traffic signs for automated vehicle control systems [Article] // Proceedings-of-the-SPIE. - 1998.

**Zhang Y. et Zhang P.** A new algorithm for character segmentation of license plate [Article] // IEEE Intelligent Vehicles Symposium. - 2003.

# Annexes

# Annexe A : Formules de Vincenty

Les formules de Vincenty sont des méthodes itératives utilisées en géodésie, développée par Thaddeus Vincenty en 1975. Elles sont basées sur le fait que la surface de la Terre est ellipsoïdale, et sont donc plus précises que les autres méthodes telle la distance des grands cercles qui supposent que la Terre est sphérique.

La première méthode (la méthode directe) permet de calculer la localisation d'un point à partir d'un point de départ, d'une direction et d'une distance. La seconde méthode (la méthode inverse) est utilisée pour calculer la distance entre 2 points. Ces deux méthodes sont très utilisées en géodésie car elles sont précises à 0.5 mm (0.000015″) sur l'ellipsoïde de la Terre.

La méthode inverse permettant de calculer la distance entre 2 points, utilisée dans les travaux de ce mémoire, est donnée ici ci-dessous (sources : [http://www.movable-type.co.uk/scripts/latlong-vincenty.html](http://www.movable-type.co.uk/scripts/latlong-vincenty.html)).

$a$, $b$ = major & minor semiaxes of the ellipsoid

$f$ = flattening $(a-b)/a$

$\phi_1$, $\phi_2$ = geodetic latitude

$L$ = difference in longitude

$U_1 = \text{atan}((1-f).\tan\phi_1)$ *(U is 'reduced latitude')*

$U_2 = \text{atan}((1-f).\tan\phi_2)$

$\lambda = L$ *(first approximation)*

iterate until change in $\lambda$ is negligible *(e.g. $10^{-12} \approx 0.06$mm)* {

$$\sin\sigma = \sqrt{[\ (\cos U_2.\sin\lambda)^2 + (\cos U_1.\sin U_2 - \sin U_1.\cos U_2.\cos\lambda)^2\ ]} \quad (14)$$

$$\cos\sigma = \sin U_1.\sin U_2 + \cos U_1.\cos U_2.\cos\lambda \quad (15)$$

$$\sigma = \text{atan2}(\sin\sigma, \cos\sigma) \quad (16)$$

$$\sin\alpha = \cos U_1.\cos U_2.\sin\lambda / \sin\sigma \quad (17)$$

$\cos^2\alpha = 1 - \sin^2\alpha$ *(trig identity; §6)*

$$\cos 2\sigma_m = \cos\sigma - 2.\sin U_1.\sin U_2/\cos^2\alpha \quad (18)$$

$$C = f/16.\cos^2\alpha.[4+f.(4-3.\cos^2\alpha)] \quad (10)$$

$$\lambda' = L + (1-C).f.\sin\alpha.\{\sigma+C.\sin\sigma.[\cos 2\sigma_m+C.\cos\sigma.(-1+2.\cos^2 2\sigma_m)]\} \quad (11)$$

}

$u^2 = \cos^2\alpha.(a^2-b^2)/b^2$

$A = 1+u^2/16384.\{4096+u^2.[-768+u^2.(320-175.u^2)]\}$ (3)

$B = u^2/1024.\{256+u^2.[-128+u^2.(74-47.u^2)]\}$ (4)

$\Delta\sigma = B.\sin\sigma.\{\cos2\sigma_m+B/4.[\cos\sigma.(-1+2.\cos^2 2\sigma_m) - B/6.\cos2\sigma_m.(-3+4.\sin^2\sigma).(-3+4.\cos^2 2\sigma_m)]\}$ (6)

$s = b.A.(\sigma-\Delta\sigma)$ (19)

$\alpha_1 = \text{atan2}(\cos U_2.\sin\lambda, \cos U_1.\sin U_2 - \sin U_1.\cos U_2.\cos\lambda)$ (20)

$\alpha_2 = \text{atan2}(\cos U_1.\sin\lambda, -\sin U_1.\cos U_2 + \cos U_1.\sin U_2.\cos\lambda)$ (21)

Où :

- $s$ est la distance (dans la même unité de mesure que a et b)
- $\alpha_1$ est l'angle initial, ou l'azimut avant
- $\alpha_2$ est l'angle final (de la direction $p_1 \rightarrow p_2$)

*Note:* Vincenty a observé que l'équation (18) devient indéterminée près des lignes équatoriales (vue que $\cos^2\alpha \rightarrow 0$) ; dans ce cas, il faut affecter la valeur de 0 à cos2σm afin que le résultat puisse être calculé correctement. Il a aussi montré que la formule peut ne pas avoir de solution entre deux points antipodaux proches ; un seuil sur le nombre d'itération résout ce problème (Vincenty dit que cela apparaît quand λ, tel que calculer à l'équation (11), est plus grand en valeur absolue que π, mais cela n'est pas toujours un test fiable).

## Fusion multi sources pour l'interprétation de l'environnement routier

**Résumé :**

Le dépassement des limitations de vitesse est l'une des causes majeures des accidents de la route, qui pourraient être réduits par l'utilisation de système robuste de détection des limitations de vitesse pouvant continuellement informer le conducteur de la bonne limitation imposée. Les travaux présentés dans ce document portent sur la réalisation d'un tel système basé sur une détection visuelle des panneaux de limitation de vitesse. Afin de rendre le système robuste, il est nécessaire de fusionner les résultats de ces détections avec les informations d'autres capteurs pour interpréter les résultats issus de la détection visuelle. C'est ainsi qu'a été entre autre spécialement développé un capteur cartographique permettant d'avoir une vision plus large sur l'horizon électronique du véhicule, ainsi qu'un système détection des lignes de marquage au sol pour analyser les changements de voie. Le processus de fusion mis en place traitant ces diverses sources d'information est fondé sur des modèles à base de règles permettant de s'affranchir des problèmes inhérents aux processus de fusion probabilistes pouvant parfois mener à des situations de doute mettant le système global en faute. Ces travaux sont le fruit d'une collaboration avec un industriel et le prototype développé a été validé expérimentalement sur route. Un outil de vérité terrain a été spécialement développé pour quantifier les résultats. Le système montre d'excellents résultats en détection et reconnaissance des panneaux de limitation de vitesse ainsi que dans la clarification de situations complexes.

**Mots clés :** aide à la conduite automobile, détection de panneau, reconnaissance de panneau routier, limitation de vitesse, fusion, vision, cartographie routière

## Fusion of multi sensor data for road environment comprehension

**Abstract :**

Exceeding speed limits is a major cause of road accidents, which could be reduced by the use of robust detection of speed limits that may continuously inform the driver of the proper speed limitation. The work presented in this document relate to the achievement of such a system based on a visual detection of speed limit signs. To make the system robust, it is necessary to merge the results of these detections with information from other sensors to interpret the results of the visual detection. For this aim, two algorithms were developed. First, a specific geographic information system was developed in order to expand the electronic horizon of the vehicle. The fusion process in place addressing these various sources of information is based on model-based rules to overcome the problems inherent to the probabilistic fusion process that can sometimes lead to uncertain situations putting the whole system in global fault. These works are the fruit of collaboration with an automotive supplier and the prototype has been validated experimentally on the road and in real conditions. A ground truth tool has been specially developed to quantify the results. The system shows excellent results with high detection and classification rates for speed limit signs recognition and complex situations analysis.

**Keywords :** driving assistance system, traffic signs recognition, detection, classification, speed limit determination, fusion, vision, geographic information system