UNIVERSITÉ PARIS-SUD

ÉCOLE DOCTORALE : Sciences et Technologie de l'Information, des Télécommunications et des Systèmes

Institut d'Electronique Fondamentale (IEF)

DISCIPLINE : Physique

RÉSUMÉ DE LA THÈSE DE DOCTORAT EN FRANCAIS

soutenance prévue le 12/06/2015

par

Qiong NIE

Cumulative methods for image based Driver Assistance Systems: applications to egomotion estimation, motion analysis and object detection

Directeur de thèse :	Alain MERIGOT	Professeur (IEF, Université Paris-sud)
Co-Directeur de thèse :	Samia BOUCHAFA	Professeur (IBISC, Université d'Évry Val d'Essonne)
Composition du jury :		
Rapporteurs :	Didier AUBERT	Directeur de Recherche (IFSTTAR, LEPSIS)
	Séverine DUBUISSON	HDR (Université Pierre et Marie Curie)
Examinateurs :	Vincent FREMONT	HDR (Heudiasyc, Universit de Technologie de Compigne)
	Sylvie LELANDAIS	Professeur (IBISC, Université d'Évry Val d'Essonne)

Abstract

La thèse porte sur la détection d'objets à partir d'une caméra embarquée sur un véhicule mobile en exploitant l'approche monoculaire « c-vélocité ». Cette méthode s'inspire de la méthode appelée « v-disparité » utilisée en stéréovision : toutes deux ont pour objectif la détection d'objets en les approximant par des plans d'orientations différentes, ce qui permet d'éviter, en monoculaire, d'estimer la profondeur. Ces deux approches, monoculaires et binoculaires, permettent de transformer le problème complexe de la détection d'objets en un problème plus simple de détection de formes paramétriques simples (droites, paraboles) dans un nouvel espace de représentation où la détection peut être réalisée à l'aide d'une transformée de Hough. La « c-vélocité », pour être efficace, requiert un calcul assez précis du flot optique et une bonne estimation de la position du Foyer d'expansion (FOE). Dans cette thèse, nous avons étudié les approches existantes de calcul de flot optique et sommes arrivés à la conclusion qu'aucune n'est vraiment performante notamment sur les régions homogènes telle que la route dans les scènes qui correspondent à l'application que nous visons à savoir : les véhicules intelligents. Par ailleurs, les méthodes d'estimation du flot optique peinent également à fournir une bonne estimation dans le cas de déplacement importants dans les régions proches de la caméra. Nous proposons dans cette thèse d'exploiter à la fois un modèle 3D de la scène et une estimation approximative de la vitesse du véhicule à partir d'autres capteurs intégrés. L'utilisation de connaissances a priori permet de compenser le flot dominant pour faciliter l'estimation de la partie résiduelle par une approche classique. Par ailleurs, trois approches différentes sont proposées pour détecter le foyer d'expansion. Parmi elles, nous proposons une méthode novatrice permettant d'estimer le FOE en exploitant la norme du flot et la structure de la scène à partir d'un processus « c-vélocité » inversé. En plus d'améliorer ces étapes préliminaires, nous proposons aussi l'optimisation et l'accélération de l'algorithme « c-vélocité » par une implémentation multithread. Enfin, nous proposons une modification de l'approche c-vélocité d'origine afin d'anticiper une éventuelle coopération mouvement/stéréo, proposée en perspective, à travers un jumelage avec la v-disparité.

Table des matières

Abstract i

Introduction 1

1 C-velocité : Principe et Implémentation 5
- 1.1 Introduction base de « c-velocité » . . . 5
- 1.2 Optimisations . . . 6
  - 1.2.1 Transformée directe sans construction des image « c-velocité » . . . 7
  - 1.2.2 Implémentation multi-thread . . . 7
- 1.3 Définition du « v-velocité » . . . 8

2 Estimation du mouvement 2D 12
- 2.1 Méthode d'estimation du flot optique par la compensation . . . 13
- 2.2 Résultats expérimentales . . . 14

3 Estimation du mouvement translational 3D 19
- 3.1 Estimation de FOE basée sur la stratégie de vote . . . 19
- 3.2 Estimation de FOE basée sur les moindres carrés . . . 21
- 3.3 Estimation de FOE basée sur la « c-velocité » inversée . . . 21

4 Détection d'objet par c-vélocité 26

Conclusion 30

Bibliographie 31

Introduction

Comme un des segments qui connaît la croissance la plus rapide dans la recherche automatique, l'aide à la conduite (ADAS en anglais) est pour proposer un conduite sûre et intelligente. Le stucture est souvent composé par un système d'acquisition pour obtenir l'information provenant des différents capteurs, une phase de traitement à transmettre les avertissements pour le conducteur, et parfois une phase à réagir pour contrôler l'accélération, le freinage, etc. Les deux capteurs de type different (proprioceptifs ou exteroceptives) sont largment utilisés dans les systèmes ADAS. Parmi eux, le LIDAR est principalement utilisé pour la plupart des véhicules intelligents (eg. La Google véhicule sans chauffeur) car il peut engendre une carte 3D de l'environnement en haute résolution. Comme les mesures des capteurs sont assez hétérogènes, la fusion de données de caputeurs différents est nécessaire pour réaliser des tâches différentes dans les systèmes ADAS.

Avec l'évolution de traitement d'image, les caméras sont de plus en plus utilisés comme complément par d'autres technologies de détection pour élargir des capacités et aussi pour renforcer la robustesse des résultats. Dans l'exemple de [1], un algorithme précis d'estimation routier est basé sur les mesures de la caméra, RADAR, les capteurs de vitesse de roue et de l'IMU, où la caméra est utilisé pour la détection de marquage de la voie. Comme beacoup d'informations peuvent être exploitées à partir d'images (les attributs visuels, le mouvement, la disparité), et les caméras sont aussi peu cher comparé avec le LIDAR, le système d'aide à la conduite basé sur la vision doit avoir un bon avenir. En fait, ses capacites sont déjà prouvés par de nombreuses recherches. Par exemple, vision stéréo est maintenant utilisé pour décrire l'environnement ou de détecter les obstacles en calculant le quadrillé d'occupation [2–4]. La route, comme une source d'information importante, peut être détectée par l'extraction et classification des primitives [5–7]. L'estimation de mouvement est nécessaire pour de nombreuses tâches

dans les système d'aide à la conduite comme l'estimation d'egomouvement, la detection d'objet en mouvement, et la reconstruction de 3D tandis que le flot optique montre son potentiel.

Dans le cadre des véhicules intelligents, une question importante est de détecter et d'identifier les objets pertinents à partir de l'image dans l'environnement de conduite (obstacles sur la route par exemple) pour un navigation de sécurité. Différente systèmes sont proposés pour atteindre cet objectif. Ces approches sont tellement divers et variés que nous pouvons généralement les classifier en deux catégories : celles qui visent à détecter les obstacles dans les images statiques et ceux qui exploitent stéréo et/ou motion. Dans la première classe, les études étaient fondées sur la définition des attributs visuels à fin de distinguer les obstacles d'autres objets [8–10]. Mais ces methodes sont plutôt adaptée pour la détection d'obstacles rigides comme vehicule, la détection de piétons n'est pas efficace. Dans un tel cas, les approches de classification et de reconnaissance sont plus robustes. Les objets sont caractérisés par un ensemble de fonctionnalités et de la classification repose sur eux. Les classificateurs populaires sont : la machine à vecteurs de support (SVM en anglais) [11], les réseux de neurones [12], Adaboost [13] et les classificateurs en cascade [14]. Concernant les caracteristiques et les descripteurs, il y a l'Histogramme de gradient orienté (HOG en anglais) [15], JRoG (Joint Ranking of Granules en anglais) [16], l'Ondelette de Haar [17] et l'Analyse en composantes principales (PCA en anglais) [18] - dont HOG est largement utilisé pour la détection de piétons. Même si toutes ces méthodes sont très populaires et efficaces, ils ont toutefois besoin d'un processus d'apprentissage hors ligne. Du coup la deuxième catégorie d'approaches exploite la stéréovision ou mouvement afin de détecter les plans frontaux qui sont assimilées à des obstacles. Surtout dans [19], les auteurs ont défini un nouvel espace dans lequel la représentation des plans frontaux est dévenu en lignes, ce qui est facilement extraits par Transforme de Hough. Tel concept de détection d'objets à partir d'un nouvel espace ("V-disparité" image dans [19]) nous a inspirés pour nos études. Même si la stéréovision apparaît largement préféré dans ce contexte, la calibration de caméra et la rectification sont toujours requis. Cela complique le processus de système stéréovision. Donc, notre travail se concentre sur la vision monoculaire pour ses nombreux avantages, notamment son coût économique et énergétique, et des informations richesse extraites de séquences d'images monoculaire (eg. mouvement d'obstacle).

Les "obstacles" dans le cas de conduite autonome se reporter aux structures qui bloquent le chemin par un grippage de la route. Par conséquent, la plupart des méthodes de détection d'obstacle démarrent avec l'extraction de route. Cela permet de définir un corridor de conduite libre et également pour affiner une région d'intérêt (ROI) pour la détection d'obstacle. Certaines hypothèses sont habituellement définis afin de faciliter la localisation de route. Par exemple, [20] a estimé la homographie de la route par la

correspondance des caractéristiques, cette homographie a ensuite utilisé pour compenser le mouvement de la route et à détecter les obstacles comme les zones d'obstacles dans l'image apparaissent non stationnaires après la compensation de mouvement. [21] fait usage des orientations apparente de surface dans l'image afin de détecter la présence d'obstacles. Toutes les deux méthodes sont basées sur l'hypothèse que la route est une surface plane. Certaines techniques ont également supposé que le plan est constant [22]. Un modèle de route plus complexe comme l'estimation de l'inclinaison du plan de route est présentée dans [23]. Récemment, une modélisation B-spline de la surface de route s'est révélé éfficace pour la détection routière [24]. Toutefois, [19] a proposé une méthode de détection routière en stéréovision par la construction et l'enquête de l'image "v-disparité ".

Selon la présentation de [19], nous savons que les pixels d'une surface de la toute ont les mêmes disparités si elles sont situées sur la même ligne de l'image. Et les disparités dans differentes lignes de l'image peuvent être calculé par

$$d(v) = a \times v + b \tag{1}$$

Où v est la ligne d'image, a et b sont des paramètres en fonction de la hauteur de caméra et de l'angle d'inclinaison. Dans un nouvel espace de représentation appelé l'image "v-disparité ", la représentation de route planaire devient une ligne droite qui est facilement extraite. En outre, les obstacles, qui sont considérés comme plan fronto-vertical, projectent sur l'image "v-disparité " comme les lignes droites quasi-verticales qui croisent avec la ligne de route. Cette méthode nous a inspiré sur deux points : 1) la première concerne la simplification de la modélisation de plan. Afin de faciliter la détection, les objets peuvent être modélisés comme des plans avec différentes orientations. Par exemple, la route est généralement assumé comme plan horizontal. Les véhicules au-dessus de la route est considérée comme plan fronto-vertical. 2) l'autre concerne la transformation vers un autre espace qui facilite la détection. En fait, les formes d'obstacles sont très variantes. Au lieu de détecter les objets directement à partir d'images, nous pouvons aussi les transformer tout d'abord en d'autre espace où les objets deviennent facilement détectable formes (par exemple, lignes, paraboles, etc.). Dans [19], la construction d'espace transformé est basée sur la disparité. Lorsque l'on envisage la motion au lieu d'un stéréo, disparité pourrait être comparée avec la vélocité dans certaines conditions de mouvement et de configuration de caméra. Puis une méthode " c-vélocité" est proposée dans [25]. Nos recherches bénéficient de cette méthode pour la détection d'objet dans le contexte de systèmes d'aide à la conduite. Cette méthode utilise des informations de mouvement 2D (le flot optique et le point d'expansion) à définir et à exploiter les espaces de vote spécifiques afin de détecter des objets. L'estimation du flot optique et le

point d'expansion (FOE en anglais) sont considérés comme deux étapes préliminaires. Le succès de ces deux étapes conditionne la qualité de détection. Par conséquent, nous devons faire face à deux difficultés qui sont spécifiques à l'application : les régions homogènes et les grands déplacements. Ces difficultés sont à la fois problématique pour l'estimation du flot optique. Dans cette thèse, nous proposons un modèle basé sur la méthode d'estimation de mouvement qui exploits les connaissances a priori provenant d'autres capteurs pour compenser le flot dominant afin de faciliter l'estimation de la partie restante du mouvement à l'aide d'une méthode du flot optique classique. Nous nous concentrons également sur la deuxième étape préliminaire de notre approche de détection : l'estimation de FOE parce que la méthode "c-velocité" a besoin de son emplacement comme entrée. Nous avons étudié et comparé différentes méthodes d'estimation de FOE. Ensuite nous proposons trois différentes méthodes pour s'adapter à notre algorithme de détection d'objet. Enfin nous proposons aussi diverse stratégies de décision pour précisément détecter les obstacles.

1

C-velocité : Principe et Implémentation

Comme notre méthode de détection d'objet est basé sur l'approache ≪ c-velocité ≫. Nous décrivons dans cette chapitre le principe de la ≪ c-velocité ≫. Ensuite les optimisations seront proposées pour diminuer les temps de calcul. A la fin, une passage de la ≪ c-velocité ≫ à la ≪ v-velocité ≫ sera aussi proposée pour faciliter la coopération entre la stéréovision et le mouvement de la caméra.

1.1 Introduction base de ≪ c-velocité ≫

Dans cette méthode, des objets dans une situation routière urbaine comme la route, les obstacles (véhicules ou piétons) et les bâtiments à côté de la route sont supposé être plans avec differentes orientations :

(1) Route → Plan horizontal

(2) Bâtiments à côté de la route → Plans latéraux

(3) Obstacles → Planes frontaux

  (a) Véhicules avant le caméra → Fuyant / approchant plans frontaux

  (b) Piétons qui traversent la route → Plans frontal qui traversent la route

Le vector de mouvement 2D d'un plan général peut être calculer à partir du mouvement 3D translational $\mathbf{T}$ et rotational $\mathbf{\Omega}$, la distance focale f et les paramètres du plan $\mathbf{n} \times \mathbf{P} = d$:

$$\begin{cases} u = \frac{1}{fd}(a_1x^2 + a_2xy + a_3fx + a_4fy + a_5f^2) \\ v = \frac{1}{fd}(a_1xy + a_2y^2 + a_6fy + a_7fx + a_8f^2) \end{cases} \tag{1.1}$$

TABLE 1.1: Paramètres de quatre plans

Type de plan	**n**	mouvment 3D	Dist. à l'origine
(1)	(0,1,0)	$\mathbf{T} = (T_X, T_Y, T_Z)$	d_r
(2)	(1,0,0)	$\mathbf{T} = (T_X, T_Y, T_Z)$	d_b
(3.a)	(0,0,1)	$\mathbf{T} = (T_X, T_Y, T_Z + T_{Z1})$	d_o
(3.b)	(0,0,1)	$\mathbf{T} = (T_X + T_{X1}, T_Y, T_Z)$	d_o

$$
\begin{aligned}
a_1 &= -d\Omega_Y + T_Z n_X \quad & a_2 &= d\Omega_X + T_Z n_Y \\
a_3 &= T_Z n_Z - T_X n_X \quad & a_4 &= d\Omega_Z + T_X n_Y \\
a_5 &= -d\Omega_Y - T_X n_Z \quad & a_6 &= T_Z n_Z - T_Y n_Y \\
a_7 &= -d\Omega_Z - T_Y n_X \quad & a_8 &= d\Omega_X - T_Y n_Z
\end{aligned}
$$

Comme les objets à détecter sont modélisés par quatre plans d'orientations spéciaux, leurs vecteurs de mouvement 2D sont calculés et affichés dans Table 1.1. Pour les objets statiques (bâtiments et route), le mouvement 3D est le mouvement du camera (T_X, T_Y, T_Z). Mais les obstacles en mouvement (véhicules et piétons) ont leur propres mouvements (T_{X1} pour les piétons qui traversent la route et T_{Z1} pour les véhicules fuyant/approchant).

En ajoutant la définition de Focus d'Expansion (FOE), les equations du vecteur de mouvement sont réecrit dans Table 1.2. Quand nous calculons la norme du mouvement $\|w\|$, nous pouvons remarquer qu'il existe une relation linéaire entre la norme $\|w\|$ et la c-valeur (c_r, c_b et c_o), qui ne dépend que des coordonée de pixel (Voir Equation 1.2) et FOE. Apparament, la projection des pixels d'un plan sur l'image « c-velocité » est une ligne droite de pente K. Et l'idée de l'approche « c-velocité » consiste à transformer les pixels dans les images « c-velocité » dans lesquels la détection de l'objet devient l'extraction de ligne, qui est facilement réalisé par la Transformée de Hough.

$$
\|w\| = \begin{cases} K \cdot c_r & c_r = |y|\sqrt{(x - x_{FOE})^2 + (y - y_{FOE})^2} \\ K \cdot c_b & c_b = |x|\sqrt{(x - x_{FOE})^2 + (y - y_{FOE})^2} \\ K \cdot c_o & c_o = \sqrt{(x - x_{FOE_o})^2 + (y - y_{FOE_o})^2} \end{cases} \tag{1.2}
$$

1.2 Optimisations

Des optimisations sur l'algorithme « c-velocité » seront proposé pour la détection d'objet dans l'application ADAS : la simplification algorithmique et l'implémentation avec

TABLE 1.2: Les equations de mouvement 2D pour les quatre plans

Type de plan	Vecteur de mouvement 2D	Module de mouvement 2D
(1)	$u = \frac{T_Z}{fd_r} y(x - x_{FOE})$ $v = \frac{T_Z}{fd_r} y(y - y_{FOE})$	$\|w\| = \|\frac{T_Z}{fd_r}\| * \|y\| \sqrt{(x - x_{FOE})^2 + (y - y_{FOE})^2}$
(2)	$u = \frac{T_Z}{fd_b} x(x - x_{FOE})$ $v = \frac{T_Z}{fd_b} x(y - y_{FOE})$	$\|w\| = \|\frac{T_Z}{fd_b}\| * \|x\| \sqrt{(x - x_{FOE})^2 + (y - y_{FOE})^2}$
(3.a)	$u = \frac{T_Z + T_{Z1}}{d_o}(x - x_{FOE1})$ $v = \frac{T_Z + T_{Z1}}{d_o}(y - y_{FOE1})$	$\|w\| = \|\frac{T_Z + T_{Z1}}{d_o}\| * \sqrt{(x - x_{FOE1})^2 + (y - y_{FOE1})^2}$
(3.b)	$u = \frac{T_Z}{d_o}(x - x_{FOE2})$ $v = \frac{T_Z}{d_o}(y - x_{FOE2})$	$\|w\| = \|\frac{T_Z}{d_o}\| * \sqrt{(x - x_{FOE2})^2 + (y - y_{FOE2})^2}$

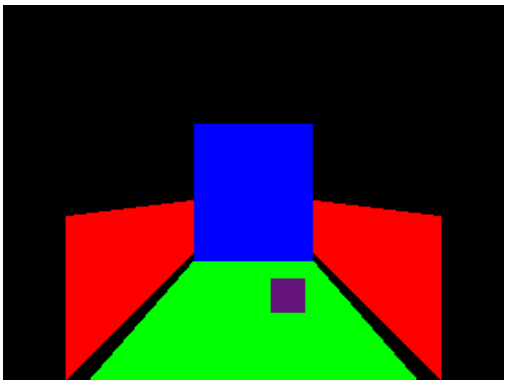
multithread.

1.2.1 Transformée directe sans construction des image « c-velocité »

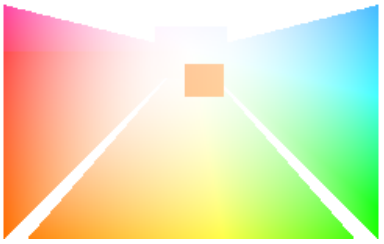
L'approche « c-velocité » peut être divisé en deux étapes : la transformée de « c-velocité » qui construit les images « c-velocité » et la transformée de Hough pour extraire les droites. Pour simplifier le processus de détection, nous proposons de construire l'histogramme de Hough directement après la computation des flots optiques et les c-valeurs. Cela permet de sauter la construction des images « c-velocité ». D'une part, le temps de calcul est largement réduit, et d'autre part, cette modification peut éviter une double quantification. La réduction de temps est montré dans Table 1.3 : deux algorithmes sont executés sur un ordinateur de Mac os avec un Intel i7 Core 2 et leur temps d'execution sont comparés. Les images sont en taille 640×480 et le temps est mesuré en deux facons. *T(s)* mesure le temps en seconde par l'utilisation de fonction *mach_absolute_time*. Et le nombre de circles est obtenu par la lecture de la valeur du registre TSC. Ces deux mesures sont cohérent avec la fréquence du processeur. D'après les résultats, l'algorithme modifié est au moins deux fois plus rapide que celui d'origine.

1.2.2 Implémentation multi-thread

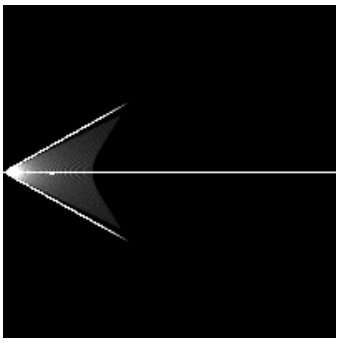
Parce que la construction des espaces de vote (les images « c-velocité » et les histogrammes 1D) est séparée pour les différents types de plan, une accélération de temps est réalisé par une implémentation de multithread. Ici nous créons un thread pour chaque espace de vote. Et l'algorithme est testé sur un ordinateur avec Intel Core i7 of 2.7 GHz. Tableau 1.4 nous montre les résultats en mesurant le temps de calcul. Ces sont comparé



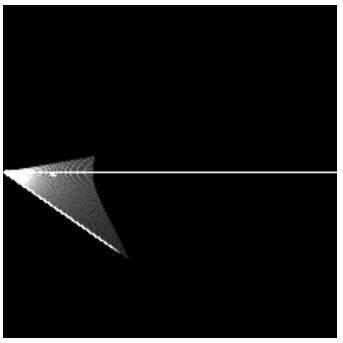
(a) Scene d'image synthetique (rouge : bâtiments ; vert : route ; bleu : l'obstacle qui approche du véhicule ; .violet : l'obstacle qui traverse la route)



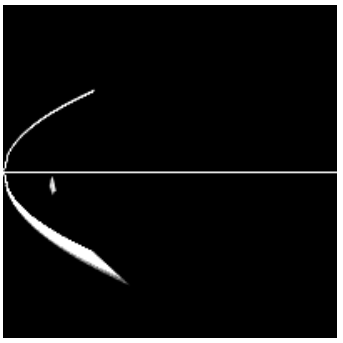
(b) Vecteurs de mouvement 2D de l'image



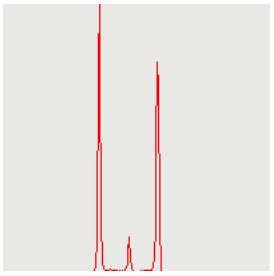
(c) L'image ≪ c-velocité ≫ pour l'espace de bâtiment



(d) L'image ≪ c-velocité ≫ pour l'espace de route

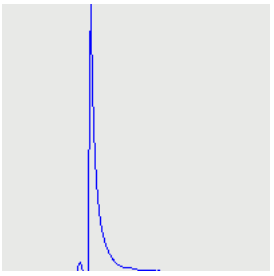


(e) L'image ≪ c-velocité ≫ pour l'espace d'obstacle



(f) Histogramme de Hough 1D pour l'image ≪ c-velocité ≫ de bâtiment

(g) Histogramme de Hough 1D pour l'image ≪ c-velocité ≫ de route



(h) Histogramme de Hough 1D pour l'image ≪ c-velocité ≫ d'obstacle

FIGURE 1.1: Un example synthétique pour la construction des image ≪ c-velocité ≫ et les histogrammes de Hough

avec les résultats dans Tableau 1.3 comme ils sont testé sur le même ordinateur. Bien que le processus de vote pour les trois différentes orientations du plan est conçu par trois threads séparés, le partage de données rend le temps de calcul pas trop accéléré.

1.3 Définition du ≪ v-velocité ≫

Dans l'approache ≪ c-vélocité ≫, nous considérons que la caméra est équipée horizontalement dans la voiture, et le plan de route est aussi horizontale. Cela signifie que la surface

TABLE 1.3: Comparaison de temps de calcul pour deux algorithmes « c-velocité »

Algorithmes / Temps	Ancien algorithme		Algorithme modifié	
	T(s)	nombre de circles	T(s)	nombre de circles
1	0.290146	781607562	0.118234	318497650
2	0.277248	746862387	0.111056	299164911
3	0.313165	843619656	0.112126	302044550
4	0.287641	774859146	0.112892	304109682
5	0.282248	760330440	0.112202	302252325
6	0.277852	748491420	0.110819	298525383
7	0.289525	779933592	0.110547	297791556
8	0.277402	747277443	0.111624	300693888
9	0.288778	777923826	0.110522	297726720
10	0.305127	821964654	0.112925	304197298
moyenne	0.2889132	778285013	0.1122947	302500396

TABLE 1.4: Temps de calcul pour l'algorithm multi-thread

	T(s)	nombre of circles
1	0.0753531	202985970
2	0.0777929	209558034
3	0.0758184	204239355
4	0.0747896	201467637
5	0.0757412	204031359
6	0.0740406	199450257
7	0.0738521	198938229
8	0.072257	194647038
9	0.0751435	202421949
10	0.0768012	206888535
moyenne	0.07515896	202462800

de la route est parallèle à l'orientation de la caméra. Toutefois, cette hypothèse est trop stricte. Mais quand on definit un petit angle θ entre le plan de route et la caméra, le module de mouvement est ensuite exprimée comme

$$\|w\| = |\frac{T_Z}{fd}||y\cos\theta + f\sin\theta|\sqrt{(x - x_{FOE})^2 + (y - y_{FOE})^2} \tag{1.3}$$

$$v_r = \frac{T_Z}{fd}(y - y_{FOE})(y\cos\theta + f\sin\theta) \tag{1.4}$$

Apparemment la c-valeur devient plus complexe comme θ est inconnu. Du coup la définition des courbes iso-velocité et la construction des images « c-velocité » demande de savoir θ. Mais si nous considérons les composants de mouvement (u_r, v_r) séparément ? A vue de l'Eaquation 1.4, c'est une equation de ordre deux de y (ligne d'image) et qu'il

peut être exprimé comme le produit de deux termes qui dépendent uniquement sur la première ordre de y :

$$v_r = \mathbf{a_1}(y - \mathbf{a_2})(y - \mathbf{a_3}) \tag{1.5}$$

avec

$$\begin{cases} \mathbf{a_1} = \frac{T_Z}{fd} \cos\theta \\ \mathbf{a_2} = y_{FOE} \\ \mathbf{a_3} = -f \tan\theta \end{cases}$$

À partir de l'équation ci-dessus, si nous construsons l'image « v-velocité », le plan de route est représenté par une courbe de l'ordre deux. Une telle courbe peut être extraite à l'aide de Transforme de Hough 3D en estimant les paramètres a_1, a_2, a_3. Cette « v-velocité » approche est plus robuste que son 'original "c-vélocité" approche comme l'angle entre la surface de la route et l'orientation de la caméra est pris en compte. En outre, pour des considérations pratiques, les « v-velocité » et « v-disparité » images pourraient être combinées pour détection routière car ils ont le même axe (ligne d'image).

Un exemple est montré dans Figure 1.2, donc la synthetic scene 3D (l'image à gauche dans Figure 1.2) est générée par la simulateur SiVIC [26]. Comme toutes les paramètres sont connues a priori, la disparité et le mouvement 2D peuvent être calculé à partir de ces paramètres. L'image « v-velocité » (l'image au milieu de Figure 1.2) et l'image « v-disparité » (l'image à droit de Figure 1.2) sont ensuite facilement construites. La représentation de la route dans l'image « v-velocité » est une courbe, qui est détectable par Transforme de Hough 3D. Nous montrons un exemple de détection dans la figure 1.3. La courbe violete est détecté par Transforme de Hough. Les pixels correspondants sont indiqués dans la figure 1.3(a) comme regions violetes. Alors que sa représentation dans l'image « v-disparité » est une ligne, ce qui est plus facile à détecter par Transforme de Hough 2D. Ses résultats de détection est affiché dans la Figure 1.3(c)(d).

À partir de la présentation ci-dessus, un plan horizontal (ou quasi-horizontal) tels que la surface de la route peut être extrait par l'approche « v-velocité » en utilisant les informations de vélocité dans la direction y. Nous pouvons aussi construre une image « u-velocité » pour détecter les plans verticaux tels que batîments et obstacles. Une telle image peut être intégrée avec l'approach « u-disparité » pour faciliter la détection des plans verticaux.

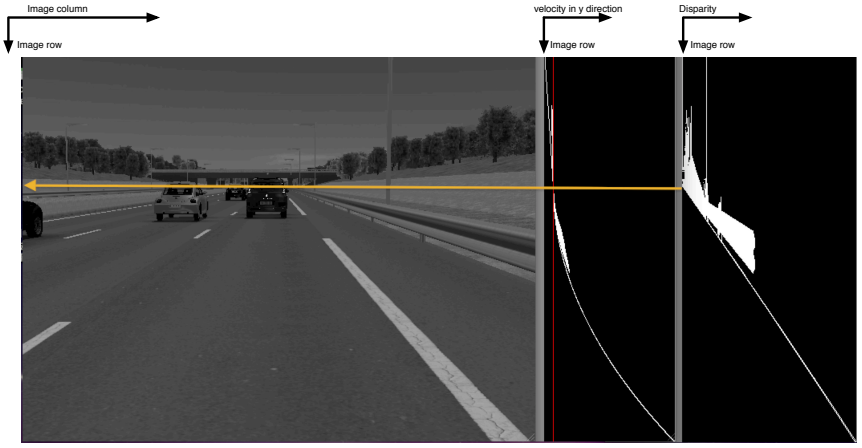
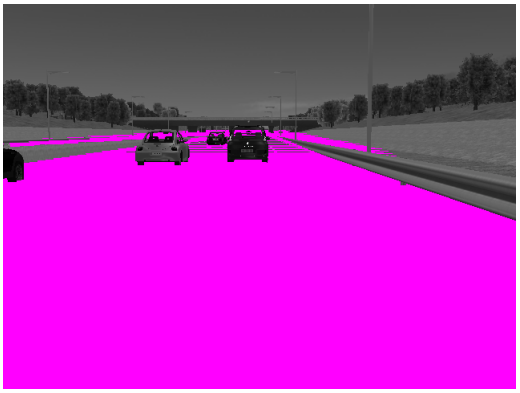
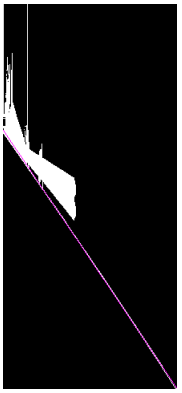


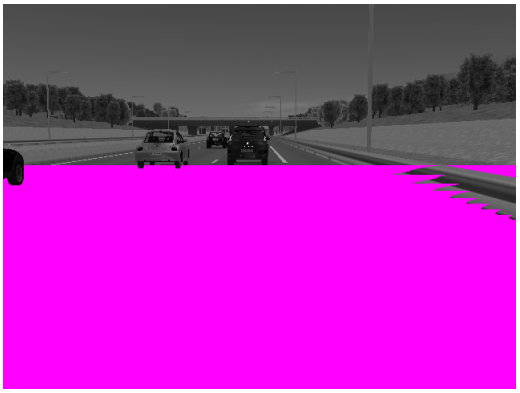
FIGURE 1.2: La comparaison entre l'image « v-velocité » et l'image « v-disparité »



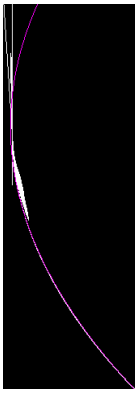
(a) Les pixels de route détectés par l'image « v-disparité »



(b) Ligne détecté par Transforme de Hough 2D dans l'image « v-disparité »



(c) Les pixels de route détectés par l'image « v-velocité »



(d) Ligne détecté par Transforme de Hough 3D dans l'image « v-velocité »

FIGURE 1.3: Détection de la route à partir de l'image « v-disparité » et de l'image « v-velocité »

2

Estimation du mouvement 2D

Comme l'approche « c-velocité » exploite flot optique (module de mouvement) pour construire les image « c-velocité » afin de simplifier la représentation et la détection d'objet. Une estimation de mouvement précis est important pour construire une exacte « c-velocité » image. Il y a beaucoup de recherches sur l'estimation du flot optique, comme la stratégie de block-matching, l'estimation du mouvement paramétrique, et l'approche différentielle. Surtout pour des stratégies différentielles, après les publications de Horn&Schunck méthode [27] et de Lucas Kanade méthode [28], beaucoup d'extensions sont proposées pour aborder les différents problèmes d'estimation et donner de bons résultats. Une présentation exhausive sur toutes ces extensions sont donnés dans [29]. Avec tels approches nombreuses et différentes, une évaluation est donc important pour nous aider à sélectionner une méthode propore pour l'approche « c-velocité ».

Pour créer une évaluation solide sur estimation du flot optique, on devrait d'abord obtenir la vérité de terrain, c'est une tâche difficile, en particulier pour des images réelles puisque aucun capteur peut mesurer le flot optique. Lorsque nous comparons les résultats avec la vérité de terrain, nous devrions aussi définir plusieurs mesures d'erreur pour évaluer les résultats. Des mesures générales comme l'erreur angulaire moyenne (AE en anglais), l'erreur de module moyenne (EE en anglais), l'erreur d'interpolation (IE en anglais) et de l'erreur d'interpolation normalisée (NE en anglais) sont habituellement appliquées pour l'évaluation. Comme la vérité de terrain pour les image réelles ne sont pas de vrai flot. Elle est calculé soit par un matching de primitives [29] ou la fusion de capteurs [30], nous pouvons proposer d'autre mesures plus significatives pour meilleur évaluer le flot optique. Nous savons que le flot optique est généralement utilisé pour récupérer l'ego-mouvement et segmenter les objets. Si un flot optique permet de récupérer un egomotion plus précis ou mieux segmenter les objets, ce flot peut être considéré comme un bon résultat. Basé sur ce point de vue, nous proposons trois mesures d'erreur en fonction de l'application. Ceux qui estime l'erreur de FOE et l'erreur de vitesse

d'avancement est utilisés pour la récupération d'ego-mouvment, parce que la position de FOE raconte les observateurs dans quel sens ils conduisent , et la vitesse d'avancement raconte les observateurs à quelle vitesse ils se déplacent. D'ailleurs, l'approche « c-velocité » utilise le flot optique pour la segmentation des plans. Du coup les résultats de segmentation dépend de la précision de flot. Selon cette propriété, nous pouvons aussi évaluer le flot optique en analysant les résultats de segmentation de la méthode « c-velocité ».

2.1 Méthode d'estimation du flot optique par la compensation

Avec tant de différentes méthodes, nous constatons qu'ils ne sont pas convenables pour notre approche « c-velocité ». Après avoir analysé les résultats de l'estimation sur différents objets, nous obtenons les conclusions suivantes : tout d'abord, les objets possibles découlant d'une situation de route urbaine sont le ciel, la route, les bâtiments, des obstacles (piétons ou véhicules). L'estimation de mouvement de ciel est inutile parce que le ciel est à infini et la mouvement devrait être zéro. Par conséquent, nous devrions tout d'abord filtrer la région de ciel avant l'estimation de mouvement. Deuxièmement, les résultats de l'estimation des bâtiments et des obstacles par différentes méthodes sont acceptables pour l'approche « c-velocité », mais les résultats sur la route sont très mauvaises. Les échecs de ces méthodes sur l'estimation de route sont causées par la spécificité de la route : une mauvaise texture et des grands déplacements dus à ego-mouvment du véhicule. En particulier, pour des points de route à proximité de la caméra, le problème de grand mouvement est habituellement résolu par la stratégie pyramidale mais cette stratégie peut conduire à une mauvaise texture dans le niveau supérieur, pour ne pas mentionner la scène avec la texture initialement mauvaise. Par conséquent, nous proposons une méthode pour compenser les flots dominants en utilisant les connaissances a priori puis de faciliter l'estimation de la partie restante par une méthode du flot optique classique. Cette méthode a été proposée d'abord par nous dans [31]. L'idée principale est comme ceci : le mouvment 2D peut être divisé en deux parties $U_0 = (u_0, v_0)$ et $\Delta U = (\Delta u, \Delta v)$ (Voir l'équation 2.1). L'ancienne partie U_0 peut être calculé directement par l'Equation 2.2 parce que la distance focale f est connue, l'hauteur de caméra est mesurable et le mouvment de caméra peut être fourni par d'autres capteur embarqués comme IMU ou d'estimation de mouvment dans les image précédentes. Puisque c'est la partie dominante, la partie restante ΔU peut alors être estimées par toutes les méthodes de l'estimation du flot optique classique.

$$\mathbf{U} = \mathbf{U_0} + \mathbf{\Delta U} \tag{2.1}$$

$$\begin{cases} u_0 = \frac{xyT_{Z_0}}{fd} \\ v_0 = \frac{y^2 T_{Z_0}}{fd} \end{cases} \tag{2.2}$$

$$\begin{cases} \Delta u = \frac{xy}{f}\Omega_X - (\frac{x^2}{f} + f)\Omega_Y + y\Omega_Z + \frac{xy\Delta T_Z - fyT_X}{fd} \\ \Delta v = (\frac{y^2}{f} + f)\Omega_X - \frac{xy}{f}\Omega_Y - x\Omega_Z + \frac{y^2\Delta T_Z - fyT_Y}{fd} \end{cases} \tag{2.3}$$

2.2 Résultats expérimentales

Cette approche est d'abord testé sur des séquences d'images synthétiques qui viennent de la simulateur SiVIC. La vérité de terrain du flot optique est connue, de sorte que nous évaluons notre méthode (appelé MMC) et les comparons avec d'autre méthodes classiques comme Horn&Schunck et Lucas&Kanade, deux variants récents MDP [32] et FOLKI [33], et un block-matching méthode. Les critères de comparaison comprennent les indicateurs classiques comme l'erreur angulair, l'erreur de module, et l'erreur relative des vecteurs de vitesse. Nous mesurons également l'erreur de FOE et l'erreur de vitesse d'avancement, que nous avons présentées ci-dessus. Les résultats de comparaison sont indiquées dans le tableau 2.1. Les résultats nous montrent que la combinaison de l'un des trois méthodes sélectionnées (FOLKI, MDP ou le bloc correspondant) avec notre méthode (MMC) donne de meilleurs résultats que n'importe quel d'entre eux seulement.

TABLE 2.1: Comaraison des techniques différentes

Technique	ARE Vx	ARE Vy	AAE	AEE	Erreur de vitesse	Error de FOE
Horn & Schunck	0.796245	0.867634	0.575541	21.7678	0.9139	9.85
Lucas & Kanade	1.06093	0.912752	1.33899	24.8179	0.8795	2.4638
MDP	0.940293	1.00278	1.10666	24.4643	0.9276	2.7192
FOLKI	0.559967	0.480318	0.202962	14.6121	0.3535	2.4958
Block Matching	1.15774	0.937277	1.49806	25.0439	0.8061	0.8764
MMC-MDP	0.0970013	0.0870454	0.0334058	1.71356	0.01994	2.92667
MMC-FOLKI	0.249	0.153	0.070	3.36146	0.0607	1.7368
MMC-BM	0.242834	0.159128	0.0838975	3.28049	0.0313	0.6992

Afin de tester la tolérance de notre approche à la variation de vitesse initiale (IS), nous évaluons notre approche avec différentes IS. Les erreurs relatives et les erreurs egomotion sont illustrés dans la Figure 2.1. Apparemment, la combinaison avec le block-matching ou MDP est très tolérant à variation.

En plus de toutes ces mesures d'erreur, nous proposons également l'approche « c-velocité » pour évaluer le flot optique comme la projection du plan sur l'image « c-velocité

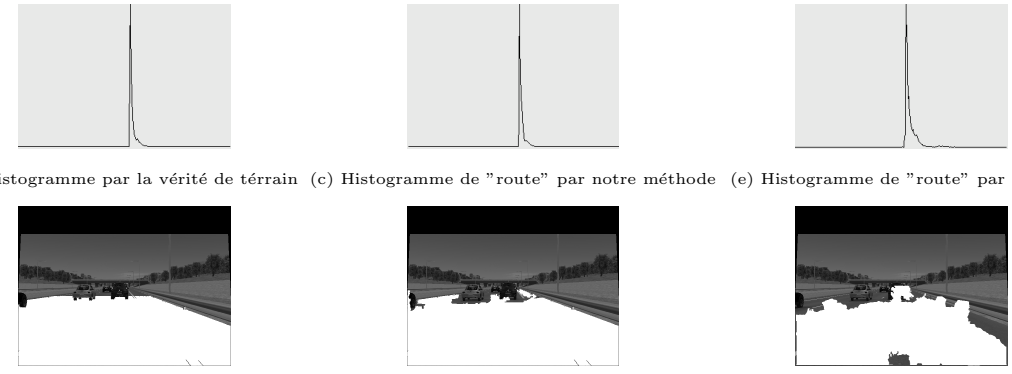
(a) Erreurs relatives (b) Erreur d'ego-mouvement

FIGURE 2.1: Comparaison de tolérance entre les méthodes différentes

» dépend de la précision du flot optique. En conséquence, nous proposons d'utiliser la mesure de ligne dans l'espace « c-velocité » comme indicateur de flot optique. La figure 2.2 et la Figure 2.3 montrent l'histogramme de la ligne de route après Transforme de Hough. Et le pic finesse reflète la précision de flot utilisé. Ici, seule la méthode FOLKI et notre approche fournissent un pic fine et perment de détecter le plan de route avec succès. Leur taux de détection de pixels routières sont indiqués dans le tableau 2.2. En outre, nous pouvons voir que nos résultats sont plus proches de la vérité de terrain (Figure 2.3).

(a) Résultat de Horn&Schunck (b) Résultat de Lucas Kanade (c) Résultat de MDP (d) Résultat de Block-Matching

FIGURE 2.2: Les histogrammes « c-velocité » de route par des flots differents



(a) Histogramme par la vérité de térrain (c) Histogramme de "route" par notre méthode (e) Histogramme de "route" par FOLKI

(b) Route segmentée par la vérité de térrain (d) Route segmentée par notre méthode (f) Route segmentée par FOLKI

FIGURE 2.3: Route segmentée par deux méthodes et leurs comparaisons avec la vérité de térrain

TABLE 2.2: Le taux de détection de route par le flot MMC-MDP flow et le flot FOLKI

	taux de pixel détecté (%)	taux de pixel mal détecté (%)
FOLKI	85.99	2.81
MMC-MDP	97.23	0.89

Nos images réelles sont tirées de la base de données KITTI. Toutes les séquences sont capturées par les caméras montées sur une voiture se déplaçant dans des situations urbaines. Le système IMU est également équipée pour donner les information de position et de vitesse. Comme nous n'avons aucune vérité de térrain sur le mouvement du véhicule, nous calculons une erreur d'interpolation exprimée par l'Equation 2.4.

$$Err = \frac{\sum |I_2(x,y) - I_1(x+u_x, y+u_y)|}{\text{pixel numbers belong to road}} \tag{2.4}$$

À l'aide d'analyse synthétique, notre méthode lorsqu'il est combiné avec MDP donne la meilleur résultat. En conséquence, nous avons choisi cette combinaison et de l'évaluer avec des images réelles. Des erreurs d'interpolation en utilisant différentes estimations de vitesse initiale sont indiqués par la courbe rouge dans la Figure 2.4. Les meilleurs résultats (la plus petite erreur) sont obtenues lorsque la vitesse initiale est d'environ 13,5 m/s. À partir de la Figure 2.4, nous pouvons voir évidemment que nos résultats dépendent de la vitesse initiale. Nous montrons qu'une autre itération en utilisant la vitesse en résultat comme une vitesse initiale permet d'améliorer les résultats (voir le clignotement bleu dans la Figure 2.4). Cela signifie que notre méthode entraîne la convergence de l'écoulement même si la supposition initiale est loin d'être le flot correct.

Ici nous utilisons également la « c-velocité » pour la segmentation de route afin d'évaluer la précision du flot optique. Les résultats sont comparés avec qui utilise directement MDP dans deux scènes differentes (voir Figure 2.5). Les résultats de la scène 1 est illustrée à la figure 2.6. Apparemment MMC-MDP a fourni un meilleur route segmentée, et elle donne un pic fine dans la histogramme de route (voir le pic marqué par le cercle violet dans la Figure 2.6(c)). Presque toute la route sont détectée, sauf la petite région près du centre de l'image comme leur modules de flots sont trop faibles (voir la Figure 2.6(e)). Tandis que sur le résultat de flot de MDP directe, une grande région de route proche de la caméra n'a pas été extraite (voir la Figure 2.6(f)). Et leurs votes dans l'histogramme de route sont concentrés dans les zones rouges (voir la Figure 2.6(d)), du coup ils ne sont pas détectés. Les résultats de scène 2 sont présentées à la Figure 2.7. Par comparaison avec la scène 1, le mouvement de la caméra dans cette scène est légèrement plus grand. Mais le flot calculé par la méthode MDP direct est totalement faut, surtout dans la centre de marquage de route (voir la Figure 2.7(b)). Alors que notre méthode fournit toujours un bon flot (voir Figure 2.7(a)), un pic fin (voir la Figure 2.7(c)) et une route bien segmenté (voir la Figure 2.7(e)).

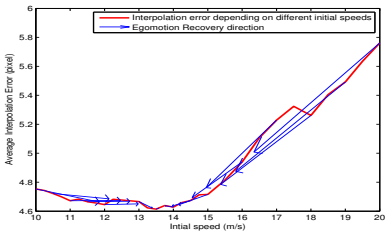


FIGURE 2.4: Erreur d'interpolation à partir de la vitesse intitial differente

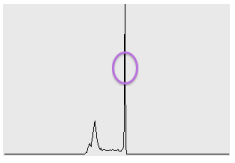
(a) scène 1

(b) scène 2

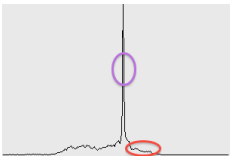
FIGURE 2.5: Deux scènes à partir de base de données KITTI

(a) Resultat de flot par MMC-MDP

(b) Résultat de flot par MDP



(c) Histogramme de route normalisée par le flot MMC-MDP



(d) Histogramme de route normalisée par le flot MDP

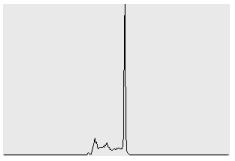
(e) Résultats de segmentation de route (région blanche) par le flot MMC-MDP

(f) Résultats de segmentation de route (région blanche) par le flot MDP

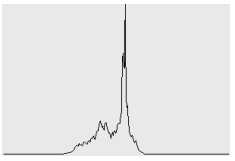
FIGURE 2.6: Résultats de segmentation de route dans scène 1 par MMC-MDP et MDP

(a) Resultat de flot par MMC-MDP

(b) Resultat de flot par MDP



(c) Histogramme de route normalisée par le flot MMC-MDP



(d) Histogramme de route normalisée par le flot MDP

(e) Résultats de segmentation de route (région blanche) par le flot MMC-MDP

(f) Résultats de segmentation de route (région blanche) par le flot MDP

FIGURE 2.7: Résultats de segmentation de route dans scène 2 par MMC-MDP et MDP

3

Estimation du mouvement translational 3D

L'estimation de FOE, comme l'autre étape préliminaire de l'approche « c-velocité » sera présenté dans cette section. Trois differente méthodes seront proposées et testé par les flots synthétiques et les images réelles. Toutes ces méthodes utilisent flot optique pour localiser FOE en sachant que le flot optique à cause du mouvement translational de la caméra converge vers FOE.

3.1 Estimation de FOE basée sur la stratégie de vote

Le flot optique pour les objets statiques capturés par un caméra en mouvement translational est exprimé par

$$\begin{cases} u = \frac{xT_Z - fT_X}{Z} = \frac{T_Z}{Z}(x - f\frac{T_X}{T_Z}) = \frac{T_Z}{Z}(x - x_{FOE}) \\ v = \frac{yT_Z - fT_Y}{Z} = \frac{T_Z}{Z}(y - f\frac{T_Y}{T_Z}) = \frac{T_Z}{Z}(y - y_{FOE}) \end{cases} \tag{3.1}$$

Si on divise les deux composantes du flot, la relation entre le flot optique et FOE est présentée comme

$$\frac{u}{v} = \frac{x - x_{FOE}}{y - y_{FOE}} \tag{3.2}$$

À partir de cette équation, tous les flots optique émanent de FOE. En d'autres mots, le FOE est situé sur la ligne d'extension inverse de flot optique. Ainsi, lorsque chaque flot vote pour tous les points situé sur sa ligne d'extension inverse comme candidature de FOE, le point qui est voté par le plus nombre de flot est la vraie localisation du FOE. C'est l'idée principale de notre première méthode.

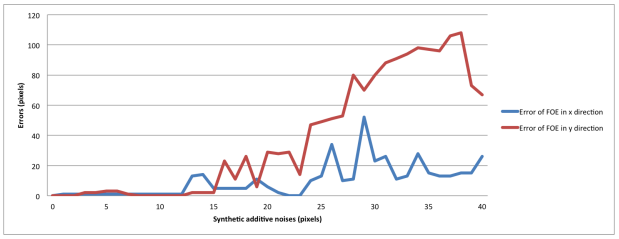


FIGURE 3.1: Impact of optical flow on FOE location

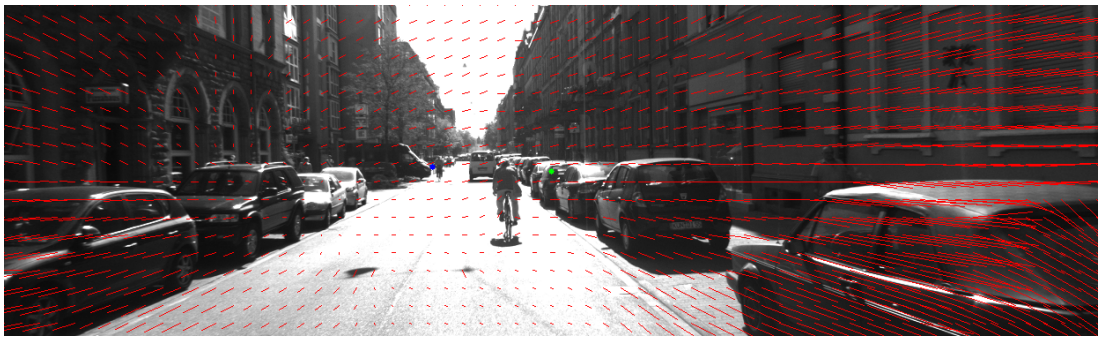


FIGURE 3.2: Un exemple de l'estimation du FOE sur les images réelles

Comme le flot optique est utilisé pour estimer le FOE, cette méthode devrait résistant au bruit du flot optique. En fait, le bruit du flot optique peut provenir d'une mauvaise estimation ou d'un objet en mouvement. Pour le premier cas, nous pouvons le testé par flot synthétique en ajoutant un bruit Gaussien blanc. Leur résultats sont affichées par la figure 3.1. Lorsque la variance de bruit $\delta < 13$, l'erreur du FOE est seulement 1 ou 2 pixels, ce qui nous montre la robusteness de notre méthode sur le bruit de flot optique.

En cas de flot optique sur les objets en mouvement, tous ces flot émanent de leur propre FOE. Les votes de ces FOEs sont normalement moins nombreuse parce que les pixels statiques sont généralement beaucoup plus nombreuse que les pixels en mouvement. Du coup les influences de ces flot optique sont facilement suprimé par la stratégie de vote.

La méthode proposée est également testé sur des images réelles de database KITTI [30]. Le flot optique est estimé par l'approche proposée dans [34]. Comme aucune vérité de terrain est fourni, seulement la évaluation qualitative peut être présentée. Un exemple est illustré à la Figure 3.2. Dont le point vert est le centre de l'image Apparemment, le FOE (le point bleu) est situé à la partie gauche de l'image. C'est cohérent avec la direction du flot optique. Dans l'exemple affiché à la Figure 3.3, le véhicule se déplace en avant, la position de FOE estimée (le point bleu) est du coup très proche au centre de l'image (le point vert).

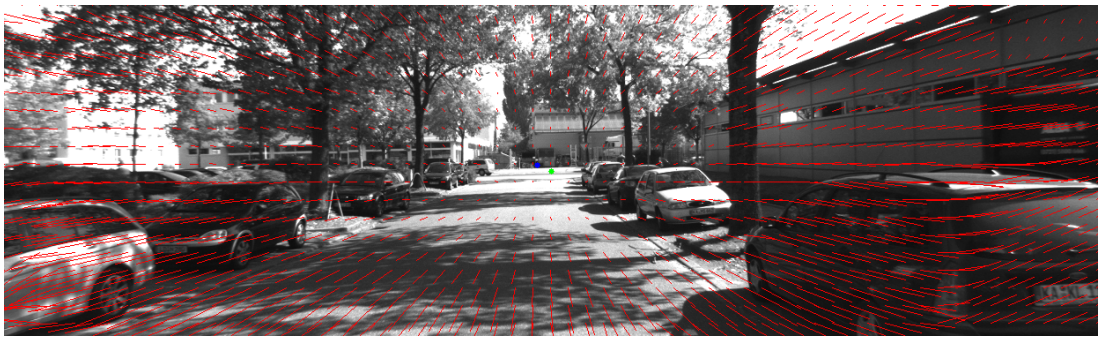


FIGURE 3.3: Un exemple de l'estimation du FOE sur les images réelles

3.2 Estimation de FOE basée sur les moindres carrés

On peut déduire que le FOE peut être calculé par les moindres carrés :

$$\begin{bmatrix} x_{FOE} \\ y_{FOE} \end{bmatrix} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{B} \tag{3.3}$$

Avec

$$\mathbf{A} = \begin{bmatrix} v_1 & v_2 & \dots & v_n \\ -u_1 & -u_2 & \dots & -u_n \end{bmatrix}^T$$
$$\mathbf{B} = \begin{bmatrix} x_1v_1 - y_1u_1 & x_2v_2 - y_2u_2 & \dots & x_nv_n - y_nu_n \end{bmatrix}^T$$

La décomposition en valeur singulière (SVD en anglais) sera du coup utilisé pour calculer la matrice inversée de $\mathbf{A}$. Toutefois, la solution de SVD est très sensibles aux bruits du flot optique. Nous avons donc recours à l'algorithme RANSAC [35] pour traiter les valeurs aberrantes. L'algorithme analogue pour estimer FOE est illustrée dans Algorithme 1.

La robustesse de la méthode proposée est validé par le flot synthétique en ajoutant le bruit Gaussien blanc. Les mêmes image réelles sont aussi testées par cette méthode. Leur résultats sont affiché à la Figure 3.4 et à la Figure 3.5. La figure 3.4(b) nous montre aussi la region de faute sélectionnée par RANSAC (les régions noires), cette sélection est cohérent avec la qualité du flot optique affiché par la figure 3.5(a).

3.3 Estimation de FOE basée sur la « c-velocité » inversée

À partir de la présentation ci-dessus sur la « c-velocité », la structure de scène peut être extraite à l'aide du flot optique et le FOE. Comme l'ego-mouvement est aussi importante

Input: Pixel database $\Re$, N_iterations, Threshold, ε, N_mini_samples
Output: Inliers, FOE

```
begin
    ε = ∞ ;
    iterations = 0 ;
    Inliers = {∅} ;
    while iteration < N_iterations do
        samples = RandomlyChooseSamples(ℜ, 2) ;
        FOE_hypotheses = Resolve(samples) ;
        Inliers_hypotheses = {∅} ;
        for Each pixel P(x, y) ∈ ℜ do
            if NAE(FlowVector, Vector(FOE → P)) < Threshold then
                Inliers_hypotheses = Add(P) ;
            end
        end
        if Size(Inliers_hypotheses) > N_mini_samples then
            FOE_hypotheses = Resolve(Inliers) ;
            Error_hypotheses = Error(FOE_hypotheses, Inliers) ;
            if Error_hypotheses < ε then
                ε = Error_hypotheses ;
                FOE = FOE_hypotheses ;
                Inliers = Inliers_hypotheses ;
            end
        end
    end
end
```

Algorithm 1: Estimation du FOE avec RANSAC

que la géométrie de scène, nous pensons à l'extraction de l'ego-mouvement translational (FOE) à l'aide de la structure de scène. Selon la « c-velocité », la projection d'un plan (route, obstacles, ou bâtiments) dans l'image « c-velocité » est une ligne. Mais un mauvais FOE peut déformer cette ligne. Et cette ligne est plus fine lorsque le FOE utilisé est plus proche de la vraiment localisation de FOE. Du coup la « c-velocité » inversée consiste à estimer FOE en optimisant la représentation du plan dans les image de « c-velocité ». La première étape est donc l'extraction de la structure de scène. Differentes méthodes peuvent être utilisées pour terminer cette tâche. Par exemple, le système de LIDAR permet de mesurer les informations 3D. Un système de stéréovision comme la « v-disparité » est aussi possible d'extraire les plans. Notre système monoculaire - la « c-velocité » - permet aussi d'extraire la structure de scène. De plus, nous pouvons combiner la « c-velocité » et la « c-velocité » inversée pour améliorer à la fois la localisation du FOE et l'extraction de plans.

A partir de la « c-velocité », la représentation d'un plan dans l'image « c-velocité » est une ligne. Et cette ligne dispersera lorsque le FOE est mal positionné. Apparemment, la dispersion de la ligne augmente lorsque le FOE est plus loin du vrai FOE. En sachant

(a) Résultats du FOE estimé par la méthode proposée

(b) Régions inlier et outlier sélectionnées par RANSAC

FIGURE 3.4: Un exemple de l'estimation du FOE sur les image réelles

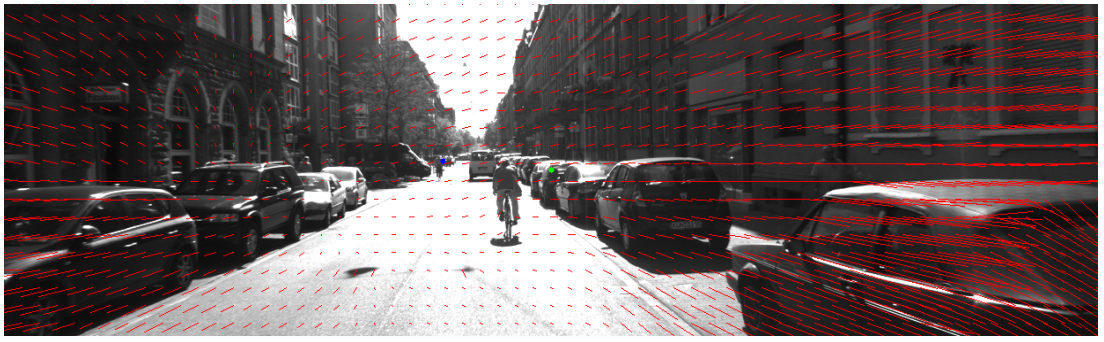
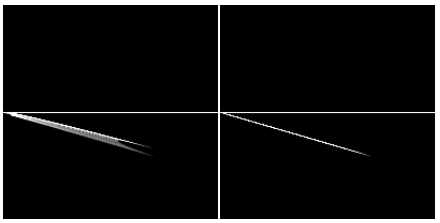


FIGURE 3.5: Un exemple de l'estimation du FOE sur les image réelles

que le FOE est utilisé pour calculer la c-valeur, la dispersion de la ligne est également la dispersion de la c-valeur. Et cette dispersion peut être du coup écrit par l'Equation 3.4. Où c_M est la c-valeur du point m calculées à l'aide du FOE, et $c_{*extsignifient}(w(m))$ est la moyenne de c-valeurs pour tous les points situés sur la même courbe de iso-vitesse de m (voir l'Equation 3.5).

$$D_{FOE}(\Pi) = \sum_{m \in \Pi} (c_m - c_{\text{mean}}(w(m)))^2 \tag{3.4}$$

$$c_{\text{mean}}(w(m)) = \frac{\sum_{\substack{i \in \Pi \\ w(i)=w(m)}} c_i}{n} \tag{3.5}$$



(a) (b)

FIGURE 3.6: Les images « c-velocité » avant (a) et après (b) l'optimisation du FOE

Cette dispersion, qui est prouvé globalement monotone avec la distance entre le FOE utilisé et le vrai FOE, peut ensuite être utilisé comme métrique pour calculer le vrai FOE. L'estimation du FOE est ensuite transformé en un problème de minimisation sur la dispersion $D_{FOE}(Pi)$:

$$\begin{aligned}(x_{FOE}, y_{FOE}) &= \operatorname{argmin} D_{FOE}(\Pi) \\ &= \sum_{m \in \Pi} (c_m(x_{FOE}, y_{FOE}) - c_{\text{mean}}(w(m)))^2\end{aligned} \tag{3.6}$$

Pour résoudre le problème de minimisation, classique algorithmes de gradient peuvent être utilisé. Un résultat de l'optimisation à l'aide de l'algorithme de la plus profonde descente (steepest descent en anglais) est illustré à la Figure 3.6, où nous pouvons comparer la représentation du plan dans l'image « c-velocité » avant et après l'optimisation. Apparemment la déformation de la représentation du plan est beaucoup compensé après l'optimisation.

La méthode est tout d'abord tester sur le flot synthetique pour analyser la résistance au bruit du flot optique, à la rotation du flot optique. Ensuite nous la testons sur les image réelles. Sans vérité de térrain, nous ne peut donc que donner une évaluation qualitative sur l'extrait du FOE. Figure 3.7 affiche le FOE (le point bleu) extraite par notre méthode, on peut le comparer avec le centre de l'image (le point vert). Nous pouvons voir que le champ de flot optique de la route est vers à ce FOE, ce qui prouve de l'exactitude du FOE estimé.

En fait, en dehors d'utiliser la champs de vecteur comme indicateur visuel pour évaluer le FOE, une autre manière est de comparer l'image « c-velocité » initial (construits avec un FOE supposé situé dans le centre de l'image) avec l'image « c-velocité » finale (construit avec l'extrait du FOE). Une telle comparaison peut être trouvé dans la Figure 3.8. Nous pouvons facilement constater que la représentation du plan (la ligne) est beaucoup plus fine après l'optimisation.

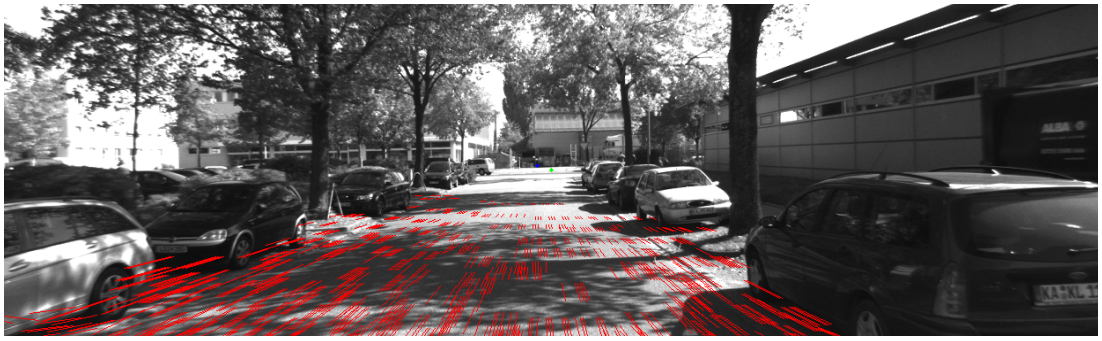
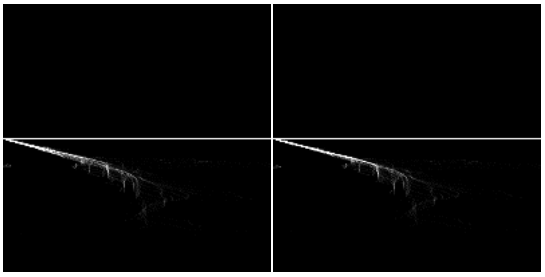


FIGURE 3.7: FOE extrait (le point bleu) par la méthode proposée



(a) (b)

FIGURE 3.8: Les images « c-velocité » avant (a) et après (b) l'optimisation du FOE

4

Détection d'objet par c-vélocité

Dans l'approche « c-velocité », une espace de vote est conçu pour chaque catégorie de plan. En outre, chaque espace de vote est traitée indépendamment des autres (ceux qui correspondent à d'autres modèles de plan) qui rend plus compliquée la sélection de maxima. Quand un pixel vote dans chaque espace de vote quel que soit son modèle de plan. En théorie, son contribution de vote fait émerger un pic seulement dans l'espace de vote correspondant à son modèle de plan. Toutefois, dans la pratique, même si nous avons flot optique précis et le vrai FOE, le bruit de la part d'autres modèles, ce que nous appelons "perturbation inter-module", rend la sélection de maxima plus difficile à exécuter. Du coup, nous proposons ici de rendre le processus de détection du plan plus robuste par traiter tous les espaces de vote en même temps. À cette fin, nous adaptons une méthode pour diviser itérativement les histogrammes afin de réduire la contribution des voteurs qui appartiennent à d'autres modèles de plan.

Cette histogramme divising méthode a été proposé pour segmenter les images en couleur [36]. Cette approche considère tout d'abord une région initial comme entrée dans l'image avec toutes les histogrammes de couleurs. Ensuite il determine un seuil haut et un seuil bas pour un pic le plus significant en considerant toutes les histogrammes. Les pixels correspondants sont ensuite enlevés à partir de la région courante. La segmentation est itérativement faite jusqu'à sans pic. Une disavantage de cet algorithme est que les caracterisques RGB sont très redondants. Mais dans notre cas, sauf sur les limites de plan, les espaces de vote ne sont pas corrélées. Du coup nous pouvons éviter les inconvénients connus de techniques Ohlander. L'algorithme détaillé pour détecter différentes catégories de plan est donnée à la Figure 4.1.

La méthode est à la fois testé par les images synthétique et les images réelles (voir la Figure 4.2 et la Figure 4.3).

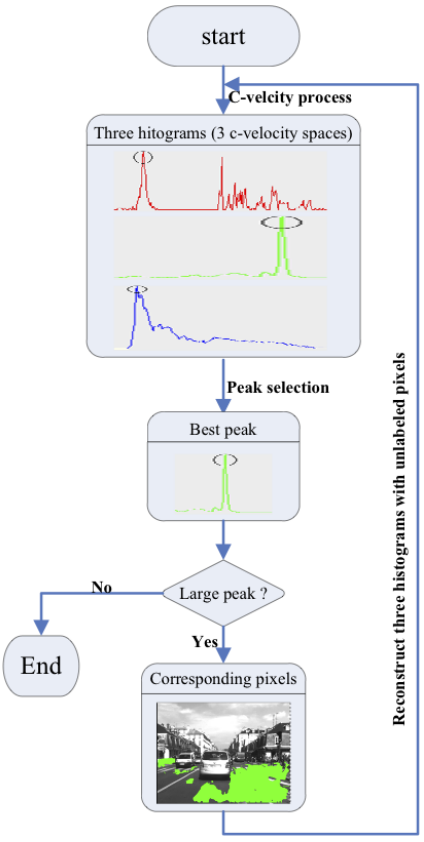
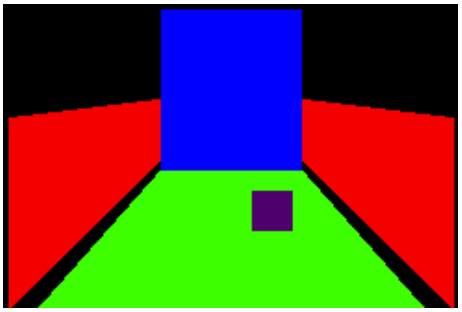
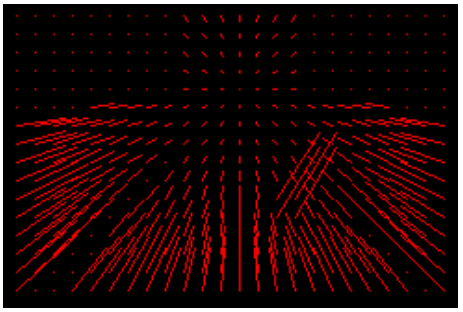


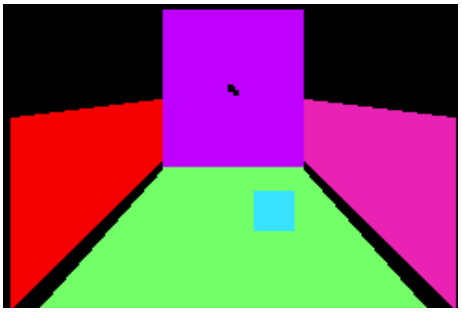
FIGURE 4.1: Schema of Ohlander-like algorithm



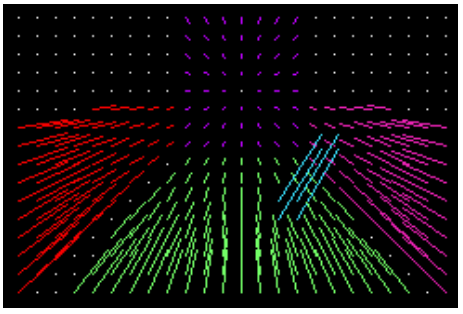
(a) Scene synthétique



(b) Flot optique synthétique

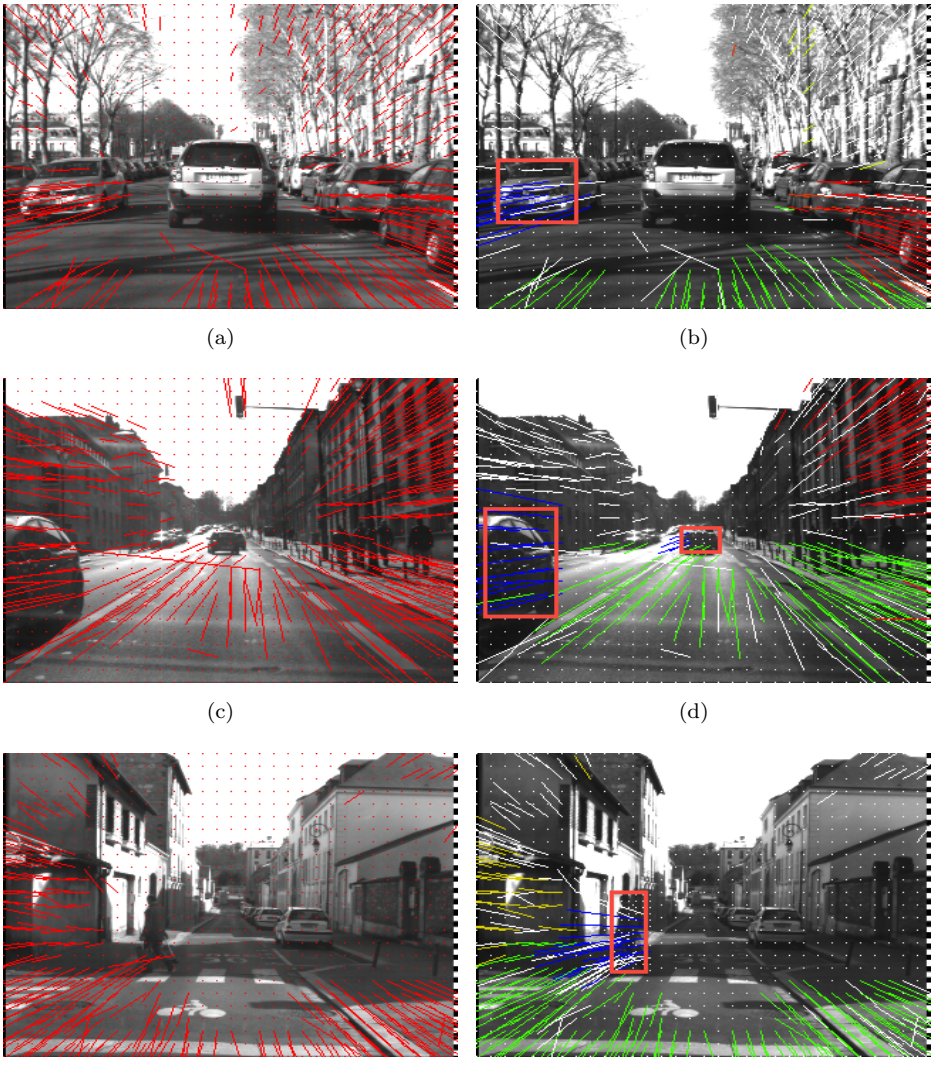


(c) Plans détectés et marqués par couleurs



(d) Flot optique marqué par differents couleurs

FIGURE 4.2: Resultat sur l'image synthétique



(a) (b)

(c) (d)

(e) (f)

FIGURE 4.3: Resultats sur l'image réelle

Conclusion et perspectives

La recheche de cette thèse est focalisé sur l'estimation de la structure de scène à partir d'une caméra embarquée sur un véhicule mobile à l'aide d'une méthode monoculaire - « c-velocité », qui considère que la scène 3D peut être approximée par un ensemble de formes paramétrées (en l'occurrence des surfaces planaires qui correspondent à des objets de type bâtiments, route, piéton, etc.). Cette méthode est cumulative et exploite certaines propriétés des courbes d'iso-vitesse. Mais cette méthode, pour être efficace, requiert un calcul assez précis du flot optique et une bonne estimation de la position du FOE. Du coup on a proposé trois approches de calculs de FOE. Et aussi nous avons travaillé sur une amélioration du flot optique en exploitant des connaissances a priori qui proviennent notamment d'autres capteurs. Par ailleurs, nous avons aussi proposé une modification de l'approche « c-velocité » d'origine afin d'anticiper une éventuelle coopération mouvement/stéréo à travers un jumelage avec la « v-disparité ».

Malgré des résultats encourageants, notre travail a mis au jour un certain nombre de pistes qu'il serait intéressant de poursuivre.

Tout d'abord, nous devons continuer la combinaison de « v-velocité » et « v-disparité ». Parce que la « v-velocité » peut d'une part confirmer la détection dans l'image « v-disparité » et d'autre part présenter des informations de mouvement pour la détection de l'objet mobile.

Concernant la méthode d'estimation du flot optique proposé, seul le modèle routier (plan horizontal) est testé pour l'estimation. Nous croyons que les autres modèles (bâtiments ou obstacles) peuvent être également appliquées avec cette méthode. Nous pouvons aussi travailler sur la combinaison de l'approche « c-velocité » et l'approche « c-velocité » inversée, parce qu'il permet de détecter le FOE et d'extraire la structure de scène de manière itérativement.

Enfin, une implémentation matérielle est envisagé pour monté sur un véhicule intelligent pour un test en temps réel. Comme la construction et l'accumulation des espaces de vote sont indépendants pour les différents types d'objet, nous pensons qu'un parallélisme au niveau du matériel n'est pas difficile à réaliser.

Bibliographie

[1] Michael Darms, Paul Rybski, Christopher R. Baker, and Christopher Urmson. Obstacle detection and tracking for the urban challenge. *IEEE Transactions on Intelligent Transportation Systems*, 10(3) :475–485, September 2009.

[2] M. Perrollaz, J.-D. Yoder, A. Nègre, A. Spalanzani, and C. Laugier. A visibility-based approach for occupancy grid computation in disparity space. *Intelligent Transportation Systems, IEEE Transactions on*, 13(3) :1383–1393, Sept 2012.

[3] Thien-Nghia Nguyen, B. Michaelis, A. Al-Hamadi, M. Tornow, and M. Meinecke. Stereo-camera-based urban environment perception using occupancy grid and object tracking. *Intelligent Transportation Systems, IEEE Transactions on*, 13(1) : 154–165, March 2012.

[4] C.G. Keller, M. Enzweiler, M. Rohrbach, D. Fernandez Llorca, C. Schnorr, and D.M. Gavrila. The benefits of dense stereo for pedestrian detection. *Intelligent Transportation Systems, IEEE Transactions on*, 12(4) :1096–1106, Dec 2011.

[5] J. Fritsch, T. Kuhnl, and F. Kummert. Monocular road terrain detection by combining visual and spatial information. *Intelligent Transportation Systems, IEEE Transactions on*, 15(4) :1586–1596, Aug 2014.

[6] R. Gopalan, Tsai Hong, M. Shneier, and R. Chellappa. A learning approach towards detection and tracking of lane markings. *Intelligent Transportation Systems, IEEE Transactions on*, 13(3) :1088–1098, Sept 2012.

[7] J.M.A. Alvarez and A.M. Ĺopez. Road detection based on illuminant invariance. *Intelligent Transportation Systems, IEEE Transactions on*, 12(1) :184–193, March 2011.

[8] A. Bensrhair, M. Bertozzi, A. Broggi, P. Miche, S. Mousset, and G. Toulminet. A cooperative approach to vision-based vehicle detection. In *Intelligent Transportation Systems, 2001. Proceedings. 2001 IEEE*, pages 207–212, 2001.

[9] Thomas Kalinke, Christos Tzomakas, and Werner V. Seelen. A texture-based object detection and an adaptive model-based classification. In *in Procs. IEEE Intelligent Vehicles Symposium'98*, pages 341–346, 1998.

[10] Shashi D. Buluswar and Bruce A. Draper. Color machine vision for autonomous vehicles. *Engineering Applications of Artificial Intelligence*, 11(2) :245 – 256, 1998.

[11] V.N. Vapnik. An overview of statistical learning theory. *Neural Networks, IEEE Transactions on*, 10(5) :988–999, Sep 1999.

[12] S. Munder and D.M. Gavrila. An experimental study on pedestrian classification. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(11) :1863–1868, Nov 2006.

[13] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning : data mining, inference and prediction.* Springer, 2 edition, 2008.

[14] João Gama and Pavel Brazdil. Cascade generalization. *Machine Learning*, 41(3) : 315–343, 2000.

[15] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893 vol. 1, June 2005.

[16] Chang Huang and R. Nevatia. High performance object detection by collaborative learning of joint ranking of granules features. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 41–48, June 2010.

[17] P. Viola, M.J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 734–741 vol.2, Oct 2003.

[18] M.A. Turk and A.P. Pentland. Face recognition using eigenfaces. In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR '91., IEEE Computer Society Conference on*, pages 586–591, Jun 1991.

[19] R. Labayrade, D. Aubert, and J. P Tarel. Real time obstacle detection in stereovision on non flat road geometry through "v-disparity" representation. In *Intelligent Vehicle Symposium, 2002. IEEE*, volume 2, pages 646–651 vol.2, 2002.

[20] Manolis I. A. Lourakis and Stelios C. Orphanoudakis. Visual detection of obstacles assuming a locally planar ground. pages 527–534, 1997.

[21] Todd Williamson. *A High-Performance Stereo Vision System for Obstacle Detection*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, September 1998.

[22] Zhongfei Zhang, R. Weiss, and A.R. Hanson. Qualitative obstacle detection. pages 554–559, 1994.

[23] Dieter Koller, Quang Luong, and Jitendra Malik. Binocular stereopsis and lane marker flow for vehicle navigation :. Technical report, Berkeley, CA, USA, 1994.

[24] A. Wedel, H. Badino, C. Rabe, H. Loose, U. Franke, and D. Cremers. B-spline modeling of road surfaces with an application to free-space estimation. *Intelligent Transportation Systems, IEEE Transactions on*, 10(4) :572–583, Dec 2009.

[25] S. Bouchafa and B. Zavidovique. C-velocity : A cumulative frame to segment objects from egomotion. *Pattern Recognition and Image Analysis*, 19(4) :583–590, 2009.

[26] Gruyer D., Royère C., du Lac N., Michel G., and Blosseville J.M. Sivic and rt-maps interconnected platforms for the conception and the evaluation of driving assistance systems. In *Proceedings of the ITS World Congress*, 2006.

[27] Berthold K.P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1–3) :185 – 203, 1981.

[28] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 1981 DARPA Image Understanding Workshop*, pages 121–130, 1981.

[29] Simon Baker, Daniel Scharstein, J.P. Lewis, Stefan Roth, MichaelJ. Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1) :1–31, 2011. ISSN 0920-5691.

[30] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving ? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[31] Qiong Nie, Samia Bouchafa, and Alain Merigot. Model-based optical flow for large displacements and homogeneous regions. In *IEEE International Conference on Image Processing, ICIP 2013, Melbourne, Australia, September 15-18, 2013*, pages 3865–3869, 2013.

[32] Li Xu, Jiaya Jia, and Y. Matsushita. Motion detail preserving optical flow estimation. In *Computer Vision and Pattern Recognition (CVPR)*, 2010.

[33] G. Le Besnerais and F. Champagnat. Dense optical flow by iterative local window registration. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 1, pages I–137–40, Sept 2005.

[34] T. Brox and J. Malik. Large displacement optical flow : Descriptor matching in variational motion estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(3) :500–513, March 2011.

[35] Martin A. Fischler and Robert C. Bolles. Random sample consensus : A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6) :381–395, June 1981.

[36] Ron Ohlander, Keith Price, and D. Raj Reddy. Picture segmentation using a recursive region splitting method. *Computer Graphics and Image Processing*, 8(3) : 313 – 333, 1978.