HAL
open science

Vision stéréoscopique par ordinateur pour la détection et le suivi de cibles pour une application automobile

Julien Morat

To cite this version:

Julien Morat. Vision stéréoscopique par ordinateur pour la détection et le suivi de cibles pour une application automobile. Interface homme-machine [cs.HC]. Institut National Polytechnique de Grenoble - INPG, 2008. Français. NNT : . tel-00343675

HAL Id: tel-00343675

https://theses.hal.science/tel-00343675

Submitted on 2 Dec 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

N^o attribué par la bibliothèque
|_|_|_|_|_|_|_|_|_|_|

THÈSE

pour obtenir le grade de
DOCTEUR DE L'INPG

Spécialité : Imagerie, Vision et Robotique
préparée au laboratoire **LJK**
dans le cadre de l'Ecole Doctorale :
Mathématiques, Sciences et Technologies de l'Information, Informatique

---

présentée et soutenue publiquement par

Julien Morat

1er Juillet 2008

Vision stéréoscopique par ordinateur pour la détection et le suivi de cibles pour une application automobile.

Directeur de thèse : Radu Horaud

JURY

Président :	**James Crowley**
Rapporteurs :	**Jean-Marc Lavest**
	Didier Aubert
Examinateurs :	**Frédéric Devernay**
	Sebastien Cornou
	Radu Horaud

Table des matières

Table des matières **iii**

1 Introduction **1**
1.1 Les enjeux 1
1.2 Les systèmes d'aide à la conduite 1
1.2.1 Les prestations 2
1.2.2 Les systèmes actuellement sur le marché 2
1.2.3 Les principales technologies de perception 4
1.3 La stéréo-vision : pour quelle prestation ? 6
1.3.1 La problématique 6
1.4 Structure de la thèse 7
1.4.1 Chapitre 2 : Calibrage d'une paire de caméras embarquées 8
1.4.2 Chapitre 3 : Rectification 8
1.4.3 Chapitre 4 : Mise en correspondance stéréoscopique . . . 8
1.4.4 Chapitre 5 : Segmentation des obstacles 8
1.4.5 Chapitre 6 : Mesure de déplacement 3-D 9
1.5 Notations 9
1.5.1 Les repères 9
1.5.2 Les vecteurs 9
1.5.3 Les matrices 10
1.5.4 Les opérateurs 10

2 Calibrage d'une paire de caméras embarquées **11**
2.1 Les modèles de caméra 12
2.1.1 La fonction de projection perspective linéaire 12
2.1.2 La distorsion non-linéaire 14
2.1.3 Modèle générique 17
2.1.4 Paramètres extrinsèques 20
2.2 Une paire de caméras 24
2.2.1 Géométrie interne de la tête stéréoscopique 25
2.2.2 Contrainte épipolaire 26
2.2.3 Reconstruction 3-D 28
2.3 Méthodologie d'évaluation de la qualité du calibrage 31
2.3.1 Critères de qualité 31
2.3.2 Méthodologie d'évaluation 32
2.3.3 Résultats 36
2.4 Calibrage du véhicule au cours de son cycle de vie 39
2.4.1 Perte de calibrage 40

2.4.2 Comportement de la géométrie face aux vibrations dues au roulement . . . 45
2.4.3 Détecter une perte du calibrage . . . 50
2.4.4 Calibrer « à la volée » . . . 50
2.4.5 Évaluation du calibrage en ligne . . . 52
2.4.6 Résultats . . . 53
2.5 Conclusion . . . 54

3 Rectification 57
3.1 Interprétation géométrique . . . 58
3.2 Matrices de rectification . . . 59
3.2.1 A partir de la matrice fondamentale . . . 60
3.2.2 A partir des paramètres de calibrage . . . 60
3.3 Variété des rectifications . . . 62
3.3.1 Degrés de liberté . . . 62
3.3.2 Interprétation géométrique . . . 62
3.4 Les surfaces d'iso-disparité . . . 63
3.4.1 Le cas générique . . . 63
3.4.2 Rectification fronto-parallèle . . . 64
3.4.3 Variété des surfaces d'iso-disparité . . . 65
3.4.4 Intersection des plans d'iso disparité . . . 67
3.5 Choix d'une rectification . . . 68
3.5.1 Déformation minimale . . . 69
3.5.2 Suivant un plan 3-D . . . 69
3.5.3 Suivant le plan de la route . . . 72
3.6 Conclusion . . . 73

4 Mise en correspondance stéréoscopique 75
4.1 Algorithmes de mise en correspondance . . . 76
4.1.1 Le principe . . . 76
4.1.2 Une implantation optimisée . . . 78
4.1.3 Le calcul de corrélation . . . 78
4.1.4 La fonction d'agrégation de la corrélation . . . 78
4.1.5 La mise à jour . . . 80
4.1.6 Affinage de la disparité . . . 80
4.1.7 Les critères de qualité . . . 81
4.1.8 Volume de reconstruction tronqué . . . 82
4.1.9 Résultats . . . 84
4.2 Calcul de l'orientation locale de la surface . . . 87
4.2.1 Le principe . . . 89
4.2.2 Implantation optimisée . . . 93
4.2.3 Validations . . . 98
4.3 Conclusion . . . 102

5 Segmentation des obstacles 103
5.1 État de l'art . . . 104
5.1.1 Apparence . . . 104
5.1.2 Géométrie . . . 105
5.1.3 Mouvement . . . 108
5.2 Le tamis à obstacles . . . 109

5.2.1 Principe du tamis à obstacles 110
5.3 Résultats 112
5.4 Conclusion 113

6 Mesure du déplacement 3-D des cibles pour une application automobile 117
6.1 État de l'art de la mesure de déplacement 3-D par vision 118
6.1.1 Systèmes monoculaires et mouvements 3-D rigides (*Structure-from-motion*) 119
6.1.2 Systèmes monoculaires et mouvements 3-D non rigides . . 119
6.1.3 Systèmes multi-caméra pour la mesure de mouvements 3-D rigides (*Stereo-Motion*) 120
6.1.4 Systèmes multi-caméra et mouvements 3-D non rigides (flux de scène) 121
6.1.5 Carte de profondeur dynamique 122
6.2 Méthode proposée pour la mesure de mouvement par stéreo-vision 122
6.2.1 Suivi 2-D par l'algorithme de suivi de Lucas & Kanade . 123
6.2.2 Les nombreuses extensions de Lucas & Kanade 125
6.2.3 Implantation multi-résolution fournie dans *OpenCV* . . . 129
6.2.4 Utiliser les deux images 136
6.2.5 Vecteur de paramètres **p** 137
6.2.6 Intégrer la relation entre distance et taille apparente . . . 140
6.2.7 Utilisation de régions d'intérêt 141
6.3 Validation 145
6.3.1 Évaluation quantitative 145
6.3.2 Évaluation qualitative 151
6.4 Conclusion 151

7 Conclusions et perspectives 155
7.1 Le bilan 155
7.2 Contributions 156
7.3 Perspectives 157

8 Annexe 159
8.1 Disparité et coût sous-pixelique 159
8.2 Algorithmes *LK* étendus à la stéréoscopie 161
8.2.1 Version *ELK* 161
8.2.2 Version *SLK* 164

Bibliographie 167

Chapitre 1

Introduction

Cette thèse présente les recherches effectuées au sein de *Renault*, à propos de la stéréo-vision par ordinateur pour une application d'aide à la conduite. Le premier objectif de ces travaux est d'identifier les difficultés qu'un constructeur automobile doit relever afin de pouvoir intégrer un système stéréoscopique dans un véhicule de série. Le deuxième objectif est d'apporter une solution, ou des éléments de solution, aux problèmes identifiés.

1.1 Les enjeux

Les statistiques montrent que 10 millions de personnes dans le monde sont chaque année impliquées dans un accident de la route. 20 à 30% d'entre elles sont sévèrement blessées, et 400 000 sont tuées [OEC]. La première action des constructeurs automobiles fut de réduire les conséquences d'une collision en améliorant la sécurité passive. Ainsi, la ceinture de sécurité, les sacs gonflables de sécurité et autres mesures ont permis de réduire énormément la proportion et la gravité des accidents entre 1960 et 1990 [CDA07]. Cependant, la sécurité passive semble commencer à montrer ses limites. Pour améliorer encore la sécurité, il est possible d'intervenir en amont de l'accident. Des recherches récentes montrent que l'inattention et la défaillance du jugement du conducteur sont les principales sources d'accidents et que des systèmes d'aide à la conduite peuvent réduire significativement le nombre et la gravité des accidents [GBS05].

Pour cette raison, et aussi pour des raisons de confort, les acteurs de l'industrie automobile sont de plus en plus demandeurs de systèmes « intelligents ». Cet intérêt croissant se traduit par une recherche très active dans le domaine des ITS[1], regroupant des domaines très distincts, comme la localisation, la planification et la perception de l'environnement.

1.2 Les systèmes d'aide à la conduite

Le marché propose aujourd'hui de nombreux systèmes d'aide à la conduite. Dans cette partie, nous commençons par décrire brièvement l'éventail des prestations les plus répandues. Ensuite, nous voyons quelles actions peuvent ou

[1] Intelligent Transportation System

doivent être déclenchées pour fournir la prestation. Finalement, nous présentons les principales technologies de perception.

1.2.1 Les prestations

Dans cette partie, nous décrivons les prestations les plus répandues. Nous nous limitons à celles mises en œuvre en cours de circulation, par opposition à celles dédiées au parking.

Remarque : Les noms que nous employons pour désigner les prestations sont les plus usités, mais la dénomination peut varier suivant les constructeurs.

La détection de franchissement de marquage routier , parfois appelée *LKA* (pour *Lane Keeping Assist*), a pour but de prévenir les changements de voie involontaires.

La régulation de distance est un système qui maintient le véhicule porteur à distance du véhicule précédent. Le plus souvent couplé au régulateur de vitesse, ce système est en mesure de ralentir la vitesse du véhicule, si celui-ci se rapproche trop du précédent. Aussi appelé régulateur de vitesse adaptatif, il est le plus souvent nommé par l'acronyme « ACC », qui signifie *Adaptative Cruise Control*, ou *Automatic Cruise Control* selon les constructeurs. Initialement limité aux hautes vitesses pour des raisons technologiques, l'*ACC* se voit désormais étendu aux basses vitesses (*LSF* pour *Low Speed Following*), voire jusqu'à l'arrêt complet (*Stop & Go*).

La prévention de collision se déclenche en cas de collision imminente. Ce genre de système est utile lorsque le véhicule précédent freine ou qu'un obstacle surgit subitement. Il existe autant de noms qu'il existe de constructeurs : *Presafe* pour *Daimler*, *Pre-crash Safety System* pour *Lexus* et *Collision Warning and Brake Support* pour *Volvo*. La stratégie engagée en cas de collision varie d'un système à l'autre. Sommairement, deux types de systèmes se dégagent : ceux qui prennent la main sur le conducteur, et ceux qui se contentent de l'avertir. L'éventail des différentes stratégies possibles est examiné un peu plus loin dans la partie.

1.2.2 Les systèmes actuellement sur le marché

Bien que le premier système ACC expérimental ait été installé sur une *Ford Zodiac sedan* en 1971 [tie05], ce système n'était, il y a quelques années encore, qu'une vitrine technologique réservée aux plus grandes marques. Mais aujourd'hui, les systèmes intelligents commencent réellement à investir le marché, preuve en est le nombre croissant de marques (voir le tableau 1.1) en équipant leurs véhicules.

Remarque : Les constructeurs automobiles ne sont généralement pas les créateurs des systèmes de perception. Ces systèmes sont, pour la plupart, conçus et fabriqués par des équipementiers.

Tab. 1.1: Quelques systèmes « intelligents » disponibles sur le marché.

Dénomination commerciale	**Prestations**	**Marque**	**Système de perception**
Presafe 2	Pré-Crash	Mercedes-Benz	RADAR
Collision Mitigation Braking System	Pré-Crash		RADAR
Automatic Driving Assist[2]	ACC, Pré-Crash	SUBARU	RADAR, Stéréo-vision
Dynamic Laser Cruise Control	ACC, LKA, Pré-Crash	LEXUS	RADAR, Stéréo-vision
Adaptive cruise control with Collision Warning and Brake Support	ACC, Pré-Crash	VOLVO	LIDAR
City Safety	LSF, Pré-Crash	VOLVO	RADAR
Audi	LSF, Collision Warning	Audi	RADAR
Active Cruise Control	ACC, Stop & Go	BMW	RADAR
Lane Departure Warning	Lane Departure Warning	BMW	Caméra
ACC, collision warning	ACC, Collision Warning	Ford	RADAR
ACC, Collision Warning and/or Avoidance (CWA)	ACC, Collision Warning, Pré-Crash	TOYOTA	LIDAR
Intelligent Cruise Control, Intelligent Brake Assist	ACC, Collision Warning, Pré-Crash	NISSAN	LIDAR
Collision Mitigation System	ACC, Collision Warning		RADAR

1.2.3 Les principales technologies de perception

Les équipementiers[3] proposent un éventail de capteurs (voir la figure 1.1) ou de systèmes permettant de remplir la fonction de perception de l'environnement. Nous distinguons deux types de capteurs : les capteurs actifs et passifs. Les capteurs actifs irradient la scène pour ensuite récolter le signal réfléchi. Les capteurs passifs ne font que collecter le signal présent naturellement dans la scène. Le RADAR et le LIDAR sont des capteurs actifs. Les caméras, qu'elles travaillent dans le domaine du visible, du proche infrarouge ou de l'infrarouge lointain sont des capteurs passifs.

(a) Radar *TRW* (b) Lidar *Hella* (c) Caméra thermique *raytheon*

FIG. 1.1: Les différentes technologies de perception de l'environnement du véhicule.

Le RADAR

Le radar est un capteur dit actif. Comme ceux de cette catégorie, il fournit une excellente mesure de positionnement et de vitesse. Lorsqu'il se déplace, il distingue aisément les cibles en mouvement du reste de l'environnement. Par contre, à faible vitesse, cette segmentation est plus hasardeuse, ce qui le rend inadapté aux prestations fonctionnant à basse vitesse. Les véhicules ne sont autorisés à émettre que sur deux gammes de fréquences, qui donnent des caractéristiques différentes aux RADAR. La plus haute (77GHz) offre de très bonnes performances, mais ne peut être utilisée qu'à haute vitesse à cause du champ de vue étroit. Elle convient aux applications PreCrash. Son prix élevé en fait un capteur réservé aux véhicules hauts de gamme. La gamme de fréquences inférieure (24GHz) correspond à des capteurs moins chers, pour lesquels la limite de champ est plus proche. Mais la plus grande ouverture de champ offre la possibilité de réaliser des applications comme le suivi à basse vitesse. Aujourd'hui, certains véhicules haut de gamme doivent s'équiper des deux types de RADAR, afin de couvrir aussi bien les champs proches et lointains, d'où un surcoût important.

[3]Equipementier : Industriel fabriquant des équipements, qui peuvent être très divers et variés. On rencontre surtout le terme dans le domaine automobile.

Le LIDAR

Le LIDAR est également un capteur actif. Acronyme de l'expression anglo-saxonne « Light Detection and Ranging », il permet de mesurer des distances ou des vitesses en se basant sur l'analyse des propriétés d'une lumière laser émise et réfléchie vers l'émetteur. Les LIDAR généralement employés dans les applications automobiles balayent la scène de 1 à 4 faisceaux. Chacun de ces faisceaux permet d'estimer un plan de coupe de la scène. Le LIDAR propose un bon compromis entre le coût et les performances. D'autre part, il convient aussi bien à des applications à haute vitesse qu'à basse vitesse. Ce capteur est cependant limité par sa fréquence d'échantillonnage, ce qui en interdit l'utilisation pour les applications de sécurité de type pré-crash [WDH+07].

La vision

Les systèmes basés sur la vision utilisent une ou plusieurs caméras pour capter l'environnement. A la différence du RADAR et du LIDAR, de telles technologies n'irradient pas la scène avec un signal connu. Cette caractéristique impose que le signal perçu par le capteur soit traité afin d'en tirer une information pertinente sur l'environnement. La richesse de l'information fournie par le capteur est un avantage, mais rend le traitement complexe.

Nous distinguons trois types de systèmes basés sur la vision, qui se distinguent par la nature du capteur.

Le premier type, très rudimentaire, utilise un capteur composé d'un très petit nombre de cellules photo-sensibles. Ce type de capteur est très simple et donc très bon marché. Mais l'information qu'il fournit reste très limitée (intensités en quelques points de la scène), et donc limite son usage pour des applications très simples, comme le *Lane Departure Warning System* de *Citroën*.

Le deuxième type se base sur une caméra pour percevoir l'environnement routier. L'analyse des d'éléments d'apparence, tels que les textures, les contours permet de détecter la présence d'objets connus dans l'image. L'analyse du flux optique porte également une information riche. Grâce à l'analyse de la séquence d'images, il est possible de détecter une collision imminente. Par exemple, la société *MobilEye* propose un système de détection de lignes (Lane Departure Warning) et de détection de collisions (Advanced Warning System) [GBS05] utilisant uniquement le flux d'images d'une caméra.

Le troisième type consiste à coupler deux caméras pour avoir une vision stéréoscopique de la scène. La paire d'images, au prix d'un lourd traitement, sert d'obtenir une reconstruction 3-D de la scène.

Ce dernier type est le plus prometteur. En effet, il peut fournir non seulement les prestations d'un système monoculaire, mais permet en plus d'obtenir une mesure de profondeur.

Les conditions de faible visibilité (brouillard, pluie battante, neige...) limitent les performances d'un système basé vision. Ce qui n'est pas un réel problème puisqu'elles limitent également le conducteur.

Néanmoins, il reste quelques points freinant l'utilisation d'un tel système de perception. Ainsi, la puissance de calcul requise explique que le RADAR et le LIDAR soient actuellement préférés à la vision. Cependant, l'augmentation très rapide de la puissance de calcul (par exemple l'*IMAPCAR* de *NEC*) vont, dans un futur proche, rendre très compétitif le prix de tels systèmes.

1.3 La stéréo-vision : pour quelle prestation ?

Comme nous l'avons déjà mentionné, à l'instar des capteurs actifs, la stéréo-vision permet de reconstruire l'environnement 3-D, mais avec des caractéristiques différentes. Le RADAR et le LIDAR ont prouvé leur efficacité en environnement autoroutier, lorsque le véhicule porteur et les cibles potentielles se déplacent à grande vitesse. Dans ce cas de figure, il est facile de séparer les éléments en mouvements des parties statiques de la scène. Pour ce genre d'applications, les systèmes basés sur la vision n'atteignent pas les performances du RADAR et du LIDAR. Les deux prochains objectifs des acteurs de l'industrie automobile sont d'étendre ces systèmes à des vitesses plus basses et d'en réduire le coût. Une nouvelle génération de RADAR et de LIDAR semble contenter les aspects techniques du premier défi, mais ces appareils restent onéreux. Les systèmes basés sur la vision semblent proposer une bonne alternative, pour des caractéristiques similaires, et un coût moindre. D'autre part, certaines prestations ne sont accessibles qu'aux technologies basées sur la vision, comme la détection de marquage routier ou encore la distinction des différents types de cible (piéton, véhicule).

En résumé, la stéréo-vision peut devenir un système compétitif aux capteurs actifs à condition de l'utiliser à basse vitesse, ou pour des prestations de type « détection de piétons ».

Dans le cadre de cette thèse, nous avons choisi de prendre comme exemple d'application l'ACC étendu aux basses vitesses (voire jusqu'à l'arrêt). Nous utilisons aussi le terme LSF (pour *Low Speed Following*) pour le désigner.

Typologie de la scène

L'environnement routier pour le LSF correspond à un milieu urbain, ou périurbain. Le système ne doit pas dépendre de l'infrastructure routière. La vitesse relative entre le véhicule porteur et la cible peut atteindre 100 km/h. Le plus petit objet à percevoir est un cycliste de 50 cm de large sur 1 m de haut. Le système doit être efficace de 3 m à 50 m de distance. Pour être en mesure de détecter et de suivre les véhicules, même dans les virages, le système doit couvrir un champ d'environ 40 degrés. Les contraintes d'asservissement imposent une estimée de la distance précise à $\pm 5\%$, et une estimée de la vitesse précise à ± 2.5 km/h.

1.3.1 La problématique

Parmi toutes les technologies de perception qui pourraient équiper les véhicules de demain, *Renault* s'est intéressé à la stéréo-vision. Cette technologie promet des caractéristiques très attractives. Pouvant servir à plusieurs prestations, la stéréo-vision présente des caractéristiques techniques tout à fait comparables à d'autre capteur pour un prix inférieur. D'autre part, les techniques de vision permettent des prestations qu'aucun autres capteurs actuel ne permet (comme différencier un piéton d'un arbre).

Aucun des systèmes décrits dans la section précédente (par vision) ne remplit les fonctionnalités exigées par un système de suivi. Nous proposons donc un système dédié à cette tâche, qui permettra de dépasser en termes de fonctionnalités et de performances les systèmes basés sur d'autres technologies que la

vision.

La vision par ordinateur, et plus particulièrement la stéréo-vision, est une discipline jeune. Dans le cas d'applications industrielles, et particulièrement pour l'industrie automobile, elle est encore mal maîtrisée. L'intégration d'un système stéréoscopique à bord d'un véhicule de série soulève de nombreux problèmes, dont certains sortent du cadre purement académique.

Les deux objectifs majeurs de cette thèse sont :

1. identifier les difficultés à résoudre pour pouvoir utiliser un système stéréoscopique dans une application d'aide à la conduite ;
2. lorsque c'est possible, apporter les éléments permettant de donner l'approche permettant de surmonter ces difficultés.

Dans ce cadre, les contributions majeures de la thèse porte sur :
- la multi-rectification pour la segmentation des obstacles
- la mesure précise de vitesse relative par stéréo-vision et
- l'implantation d'un système complet.

1.4 Structure de la thèse

L'objectif majeur de cette thèse est d'analyser les difficultés liées aux applications possibles de la stéréo-vision embarquée sur véhicule. Comme exemple, nous avons choisi de traiter un système de suivi de véhicule. L'utilisation d'un tel système comporte de nombreux aspects, allant du calibrage aux techniques de suivi. Pour des raisons pratiques évidentes, nous avons scindé le système en une succession de processus. Ainsi découpé, le processus de détection et de suivi de cibles comporte les aspects suivants :

1. le calibrage,
2. la rectification,
3. la mise en correspondance stéréoscopique,
4. la segmentation des obstacles et
5. la mesure de vitesse relative des obstacles ou des cibles.

La relation entre tous ces aspects est décrite par la figure 1.2. Dans la suite de cette thèse, nous décrivons chacun des aspects sous la forme de briques fonctionnelles.

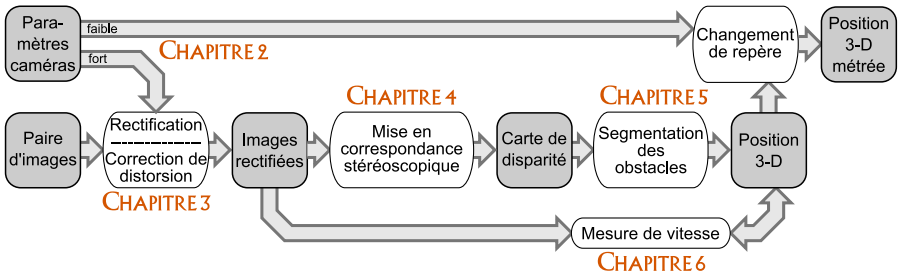


FIG. 1.2: Le processus de détection d'obstacle par stéréovision.

Pour chacune des briques, nous expliquons quel est son rôle, quelles sont les approches proposées par la littérature pour remplir la fonction, et finalement quelle approche nous avons envisagée.

1.4.1 Chapitre 2 : Calibrage d'une paire de caméras embarquées

Pour pouvoir utiliser un système de stéréo-vision comme outil de mesure, il est indispensable d'en connaître les propriétés physiques et géométriques. Construire un système avec les caractéristiques voulues serait trop coûteux, c'est pourquoi il est préférable de le calibrer après l'avoir construit. Le calibrage consiste a utiliser le système, en environnement contrôlé ou non, afin d'en déduire ses propres paramètres. La précision avec laquelle les paramètres sont estimés influent sur le fonctionnement finale du système. D'autre part, les paramètres de calibrage peuvent évoluer avec des constantes de temps différente : courtes (vibrations), moyennes (température) et longue (usure), et il peut être nécessaire d'auto-calibrer le le système au cours de son utilisation.

Dans le chapitre traitant du calibrage, nous commençons par présenter les outils mathématiques, ainsi que quelques méthodes de calibrage proposées par la littérature. Puis nous présentons une nouvelle méthode d'évaluation permettant de mettre en avant les relations qui existe entre le calibrage et la qualité de fonctionnement du système.

1.4.2 Chapitre 3 : Rectification

La rectification est une étape préliminaire permettant de simplifier considérablement le processus de mise en correspondance stéréoscopique. Cette opération consiste à déformer la paire d'images de manière à simuler une configuration simplifiée de caméras. La rectification dépend de la configuration réelle des caméras. Nous expliquons comment calculer la rectification. D'autre part, nous expliquons également qu'il existe une infinité de rectifications valides pour configuration de caméras, et nous étudions quelles sont les choix possibles et leurs conséquences.

1.4.3 Chapitre 4 : Mise en correspondance stéréoscopique

La mise en correspondance stéréoscopique, aussi appelée appariement, est le processus qui fourni, pour chaque point d'une image de la paire, son correspondant dans l'autre image. C'est une étape clef du processus de reconstruction 3-D. En première partie du chapitre, nous présentons un algorithme de la littérature et son implantation optimisée. Nous détaillons quels sont les points importants dans le cas d'applications automobiles. En deuxième partie, nous présentons une extension de cet algorithme, de manière à obtenir simultanément la reconstruction de la surface et une estimation de son orientation locale, information très utile dans le cas d'applications automobiles.

1.4.4 Chapitre 5 : Segmentation des obstacles

L'application automobile qui nous intéresse, à savoir le suivi de véhicule, nécessite la mise en place d'un processus spécifique à la mesure de vitesse. Ce pro-

cessus requiert des informations synthétiques sur les véhicules cibles, telles que leur taille apparente et leur position 3-D. En amont de la chaîne de processus, le processus de mise en correspondance stéréoscopique fournit une reconstruction 3-D de l'environnent. Cette information nécessite d'être traitée pour fournir des cibles. Ce traitement intermédiaire s'appelle l'« extraction d'obstacles » ou « segmentation des obstacles ». Il permet d'extraire de la reconstruction 3-D les informations de taille et position des cibles qui seront ensuite traitées par le processus de mesure de vitesse.

1.4.5 Chapitre 6 : Mesure de déplacement 3-D

Les capteurs utilisés en vision ne permettent pas une mesure directe de la vitesse. Il faut donc mettre en œuvre des méthodes spécifiques permettant non seulement de mesurer le déplacement 3-D, mais aussi de suivre une cible au cours du temps. La solution la plus intuitive consiste à mettre en correspondance au cours du temps les détections successives fournies par le processus de segmentation des obstacles. Mais, avec cette approche, l'enchaînement des processus engendre une accumulation des erreurs qui rend l'estimée de la vitesse inexploitable. Nous proposons donc une nouvelle approche basée sur la mesure directe du flux de scène.

1.5 Notations

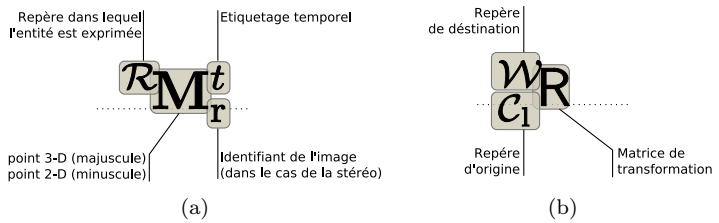


Fig. 1.3: Le canevas des notations des vecteurs (a) et des matrices (b).

1.5.1 Les repères

Les repères sont notés $\mathcal{R}_{\mathrm{w}}$, en style calligraphié, indicé de l'initial du repère. Par simplicité et pour éviter l'accumulation d'indices, les repères sont parfois noté seulement par l'initial calligraphiée, comme par exemple le repère monde (en anglais *world*), noté $\mathcal{W}$.

1.5.2 Les vecteurs

Les vecteurs et toutes les entités s'exprimant sous la forme d'un vecteur sont notés en **gras**. Dans le cas spécifique des points, les points 2-D sont notés en minuscule (ex : $\mathbf{m}$), et les points 3-D en majuscule (ex : $\mathbf{M}$). Si un point est exprimé dans un repère particulier, celui-ci sera annoté en exposant à gauche du point (cf. 1.3(a)). En indice à droite est noté l'identifiant dont est issu l'entité.

Par exemple, un point de l'image droite (en anglais *right*) sera noté $\mathbf{m}_{\mathrm{r}}$. En exposant à droite est noté l'étiquetage temporelle de la mesure, si celui-ci est significatif. Les points $\mathbf{m}_{\mathrm{r}}^{t}$ et $\mathbf{m}_{\mathrm{r}}^{t+1}$ sont deux observations du même points à des instants de temps consécutifs.

1.5.3 Les matrices

Les matrices et toutes les fonctions s'écrivant sous forme matricielle sont en style *Sans Serif* (ex : P) (cf. 1.3(b)). Dans la suite du document, nous avons :

F qui désigne la matrice fondamentale,
H qui désigne une homographie,
I qui désigne la matrice identité,
K qui désigne la matrice des paramètres intrinsèques,
R qui désigne une matrice de rotation,

(Cf. l'ouvrage de *Forsyth* et *Ponce* [FP02] traitant de la géométrie projective.)

1.5.4 Les opérateurs

L'opérateur $\sim$ signifie l'égalité à un facteur d'échelle près. c'est à dire :

$$\begin{pmatrix} a \\ b \\ 1 \end{pmatrix} \sim \begin{pmatrix} ca \\ cb \\ c \end{pmatrix} \text{ avec } c \neq 0. \tag{1.1}$$

Dans le cas d'opération sur les matrices, les opérateurs $\times$ et . désignent le produit vectoriel et le produit scalaire.

Chapitre 2

Calibrage d'une paire de caméras embarquées

Pour pouvoir utiliser un système de stéréo-vision comme outil de mesure, il est indispensable d'en connaître les propriétés physiques et géométriques. Comme l'illustre la figure 2.1, il y a deux raisons à cela. La première est que les images acquises doivent être rectifiées, afin de rendre le processus stéréoscopique rapide et robuste (se reporter au chapitre suivant pour plus de détails). Mais la deuxième raison, la plus importante, est que la reconstruction d'un point 3-D à partir de points 2-D repose sur la connaissance de la fonction de projection, qui donne à partir des coordonnées 3-D d'un point les coordonnées 2-D de sa projection dans chaque image.

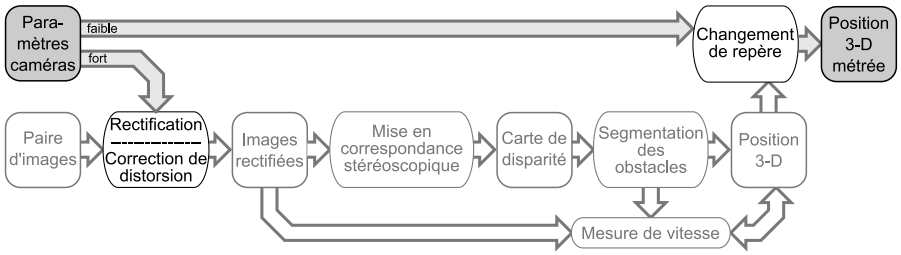


Fig. 2.1: Les paramètres issus du calibrage décrivent la configuration de la paire de caméras. Ils sont nécessaires au calcul de la rectification et de la reconstruction 3-D.

Les deux premières parties de ce chapitre sont consacrées aux modèles mis en œuvre pour caractériser les fonctions de projections d'une paire de caméra. Parmi la multitude de formulations [Tsa87, Bey92] fournies par la littérature, nous détaillons un modèle de caméra unique (le modèle sténopé couplé au modèle de distorsions non-linéaire), qui s'avère suffisant dans le cas des caméras que nous utilisons. Nous présentons ensuite comment étendre ce modèle au cas stéréoscopique. En effet, le couplage de deux modèles sténopés (un par caméra) peut suffire à représenter le système stéréoscopique. Nous présentons deux autres formulations plus adaptées à la stéréo-vision, l'une étant plus intuitive et l'autre

adaptée aux applications automobiles (grand volume de reconstruction).

Les deux dernières parties de ce chapitre traitent des méthodes de calibrage. Comme deux systèmes ne sont jamais totalement identiques, un calibrage est nécessaire pour obtenir les paramètres réels d'un système. Après une description succincte de l'approche classique de calibrage [Bou], nous présentons une nouvelle méthodologie d'évaluation de ce calibrage. En effet, une mauvaise estimation des paramètres peut introduire un biais dans le calcul de la carte de disparité, voire le rendre impossible. Identifier et comprendre les conséquences de telles imprécisions est donc primordial pour un constructeur automobile. Nous étudierons également comment calibrer un système stéréoscopique pendant son utilisation.

2.1 Les modèles de caméra

Les modèles mis en oeuvre dans un système stéréoscopique n'étant qu'une extension de ceux utilisés pour les systèmes monoculaires, nous commençons par présenter les outils permettant de modéliser une unique caméra.

Le terme *caméra* est issu du latin et signifie *chambre*, pour *chambre photographique*. Il désigne un appareil de prises de vue animées. Le fonctionnement d'une caméra est très similaire à celui d'un appareil photographique. Depuis des siècles, les artistes et mathématiciens ont cherché à comprendre et reproduire les mécanismes de l'effet de perspective, qui « aplatit » le monde sur un plan, à la manière du perspectographe dépeint dans la figure 2.2. Aujourd'hui, la géométrie, et plus particulièrement la géométrie projective, est très largement utilisée pour modéliser la fonction qui relie un point 3-D à sa projection 2-D sur le capteur photométrique d'une caméra. En effet, la géométrie projective fournie les outils mathématiques permettant de manipuler les faisceaux de droites rayonnent à partir d'un même point. Cette structure est exactement celle de la caméra sténopé dont tous les faisceaux de lumière passent par le foyer. Parmi les différents modèles existants, nous avons choisi le modèle sténopé, qui, associé à un modèle de distorsion, propose un bon compromis de précision et de simplicité.

Le modèle sténopé est séparé en une partie intrinsèque (interne au capteur) et une partie extrinsèque (externe au capteur). Dans la partie intrinsèque nous développons la formulation de la fonction de projection, mais aussi de distorsion (bien qu'en dehors du modèle sténopé). La pose (position et orientation) de l'appareil de prise de vue est, quand à elle, modélisée par la partie extrinsèque du modèle.

2.1.1 La fonction de projection perspective linéaire

Comme le montre la figure 2.3, le modèle sténopé, ou *trou d'épingle* (en anglais *pinhole*), est constitué d'un plan image I et d'un centre optique C. La principale caractéristique de ce modèle est que tous les rayons optiques issus d'un point de la scène passent par un unique point C (le clou accroché au mur dans la gravure 2.2). L'intersection du rayon optique avec le plan image I définit le point projeté $\mathbf{m}$.

Avec un système de coordonnées homogènes, la fonction de projection transformant $\mathbf{M}$ en un point de l'image $\mathbf{m}$ est modélisée par la matrice K' de taille

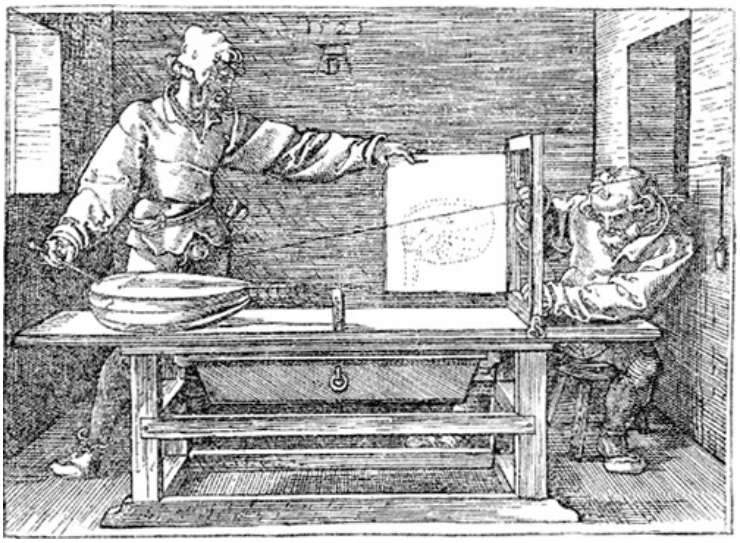


FIG. 2.2: Gravure du peintre et mathématicien *Albrecht Dürer*, mettant en scène un artiste utilisant un perspectographe à œuilleton ; outil auquel il laissera son nom.

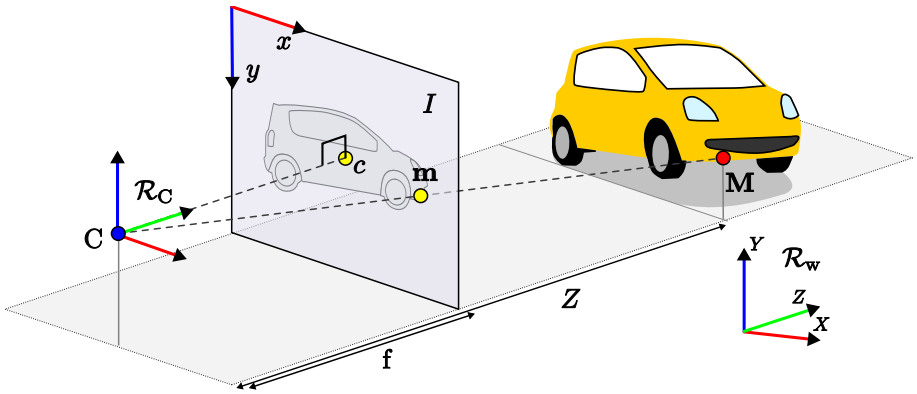


FIG. 2.3: Le modèle sténopé : la projection m du point M est définie par l'intersection de la droite (C, M) et du plan image I.

3×4 telle que :

$$\mathbf{m} \sim \mathsf{K}'\mathbf{M}, \tag{2.1}$$

où les points $\mathbf{m}$ et $\mathbf{M}$ sont exprimés en coordonnées homogènes. Le signe $\sim$ signifie que $\mathbf{m}$ est égal à un facteur d'échelle près. Se référer aux notations décrites en partie 1.5 (p. 9). La fonction se compose d'une étape de projection proprement dite et d'une étape de conversion d'unité. La première étape consiste à projeter le point 3-D $\mathbf{M}$ dans le plan rétinien (équivalent au plan image) comme

suit :

$$\mathbf{m}_{\mathrm{r}} \sim \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \mathbf{M}. \tag{2.2}$$

L'unique paramètre de cette fonction est la *distance focale* f. Le point 2-D $\mathbf{m}_{\mathrm{r}}$ obtenu est exprimé avec les mêmes unités que $\mathbf{M}$. Les coordonnées de ce point dans le plan image (qui est un échantillonnage en pixels du plan rétinien) et les coordonnées du même point dans le plan rétinien sont liées par l'équation suivante :

$$\mathbf{m} \sim \begin{pmatrix} k_x & 0 & c_x \\ 0 & k_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \mathbf{m}_{\mathrm{r}}, \tag{2.3}$$

où (c_x, c_y) sont les coordonnées du point principal (à l'intersection de l'image et de l'axe optique), k_x et k_y les facteurs d'échelle. L'application consécutive des équations 2.2 et 2.3 donne la forme de la matrice K' de l'équation 2.1 :

$$\mathsf{K}' = \begin{pmatrix} k_x & 0 & c_x \\ 0 & k_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} k_x f & 0 & c_x & 0 \\ 0 & k_y f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{2.4}$$

Les paramètres f, k_x, k_y, c_x et c_y définissent les 5 degrés de liberté de la partie intrinsèque du modèle sténopé. La matrice K' peut également se décomposer en deux matrices, dont la première, souvent appelée la matrice intrinsèque et notée K, sert dans certaines situations (par exemple lorsque l'on désire séparer la fonction de rotation de la translation). La matrice K s'écrit :

$$\begin{pmatrix} f k_x & 0 & c_x \\ 0 & f k_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{2.5}$$

2.1.2 La distorsion non-linéaire

Le modèle sténopé approxime la projection par une fonction linéaire (éq. 2.1). Malheureusement, l'écart entre cette approximation et la réalité est généralement non négligeable. Une des conséquences de la distorsion est la courbure apparente des lignes droites (voir la figure 2.4).

Dans ce cas, le modèle sténopé ne suffit plus à décrire le comportement de la caméra. Il est donc nécessaire de prendre en considération les éventuelles déviations. Les approches classiques [Tsa87, Bro71, Bey92, WCH92], que nous détaillons en première partie, formulent la distorsion en ajoutant un terme aux coordonnées (x, y) (éq. 2.1). Une fois la distorsion corrigée, l'image peut être manipulée comme si elle était issue d'une caméra sténopée.

Dans la deuxième partie, nous abordons le cas extrême où la caméra et son optique ne suivent pas une projection centrale. En effet, pour ces optiques particulières, les algorithmes classiques de stéréo-vision ne sont plus applicables. Par conséquent, lorsque la caméra est implantée dans un véhicule, il est important de s'assurer que le pare-brise n'invalide pas le modèle de projection centrale. Dans ce but, nous avons évalué la pertinence de ce modèle sur un système incluant un pare-brise.

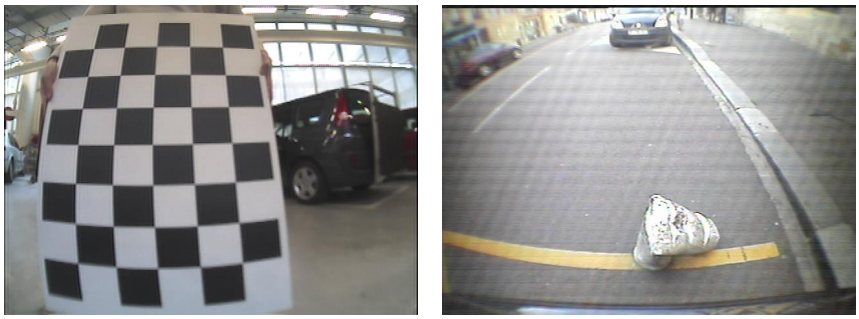


FIG. 2.4: L'optique grand angle équipant cette caméra de recul induit une distorsion en barillet particulièrement importante. Sur les bords de l'image, les carrés de la mire ainsi que le marquage jaune sont courbés.

Distorsions avec modèle sténopé

L'approche la plus utilisée consiste à modéliser les distorsions par un modèle paramétrique qui vient corriger les coordonnées (x, y) du modèle sténopé (éq. 2.1) :

$$x_{\mathrm{d}} = x + \delta_{\mathrm{x}}(x, y) \tag{2.6}$$

$$y_{\mathrm{d}} = y + \delta_{\mathrm{y}}(x, y) \tag{2.7}$$

où (x, y) sont les coordonnées sans distorsion (non-observables) et $(x_{\mathrm{d}}, y_{\mathrm{d}})$ sont les coordonnées avec la distorsion (le point réellement observé). Comme l'indique les équations 2.6 et 2.7, l'erreur de positionnement dépend de la position du point. Les défauts les plus fréquemment rencontrés avec des objectifs standards sont la distorsion radiale, la distorsion tangentielle, et la distorsion prismatique [WCH92]. Le plus souvent [Tsa87, Bey92, Zha00], seule la distorsion radiale est traitée. Ce type de distorsion provient essentiellement d'imperfections dans la courbure de la lentille. Si le déplacement radial est négatif il provoque un effet de « coussinet ». A l'inverse, un déplacement radial positif provoque un effet de « barillet », illustré dans la figure 2.4. Les fonctions de distorsion radiale (une par dimension) sont approximées par les fonctions polynomiales δ_{x} et δ_{y}. Elles s'écrivent comme suit :

$$\delta_{\mathrm{x}}(x, y) = (x - c_x) \sum_{i=1}^{\infty} k_i (x^2 + y^2)^i, \tag{2.8}$$

$$\delta_{\mathrm{y}}(x, y) = (y - c_y) \sum_{i=1}^{\infty} k_i (x^2 + y^2)^i, \tag{2.9}$$

où k_i sont les coefficients de distorsions, et (c_x, c_y) les coordonnées du centre optique. En pratique, une approximation d'ordre 3 ($i = \{1, 2, 3\}$) suffit à modéliser la distorsion radiale de la plupart des caméras. Parfois, des polynômes de degré plus élevé sont nécessaires, tels que ceux utilisés par Lavest *et al.* [LL96]. La distorsion tangentielle [Bro71, WCH92, LL96] et la distorsion prismatique [WCH92] sont dues, pour la première, à un mauvais alignement des lentilles, et pour la seconde à un défaut dans leur parallélisme.

Dans le cas d'optiques aux distorsions plus complexes, comme celles que l'on peut trouver dans les téléphones portables ou les microscopes électroniques (voir la figure 2.5), des modèles plus complexes, comme des modèles à base de splines, ont été développés [CGS+04, HS04].

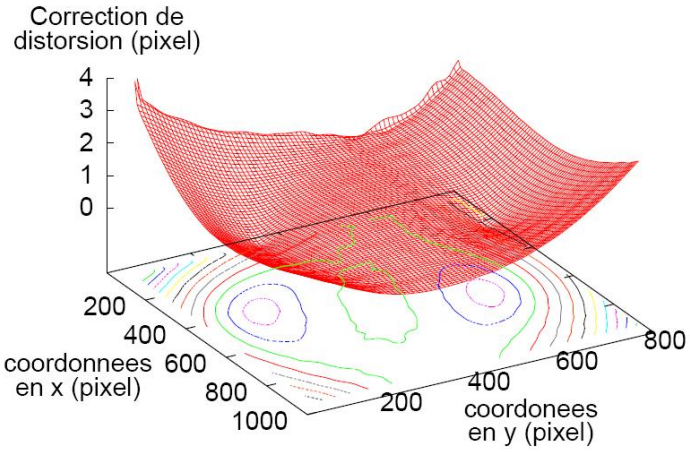


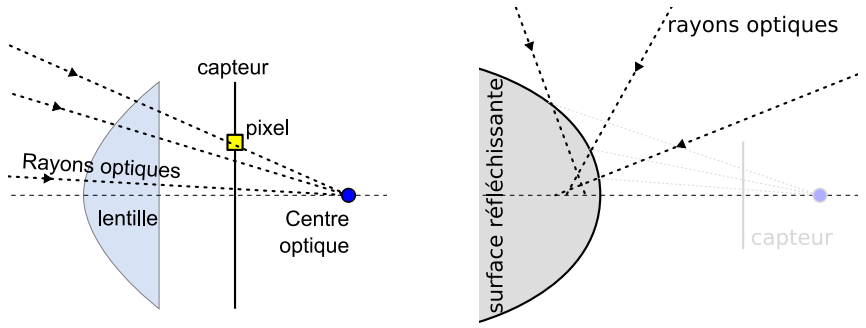
FIG. 2.5: La distorsion d'un microscope électronique n'est ni radiale, ni tangentielle, ni prismatique. La correction de celle ci passe par la définition de nouveaux modèles plus génériques [CGS+04].

Remarque : Ici, nous appelons « modèle de distorsion » la fonction δ de $(x, y) \rightarrow (x_\mathrm{d}, y_\mathrm{d})$, mais certains auteurs appellent du même nom la fonction inverse.

Application de la correction de distorsion sur les images

Le modèle de distorsion que nous présentons ici donne des coordonnées distordues $(x_\mathrm{d}, y_\mathrm{d})$ en fonction de coordonnées sans la distorsion (x, y). A l'aide de la fonction de distorsion $\delta(x, y)$, il est possible de corriger la distorsion d'une image. L'approche classique (implantée dans *OpenCV* [Int01]) consiste à parcourir les pixels de l'image non-distordue pour les remplir. En effet, pour chaque pixel corrigé, le point correspondant dans l'image distordue est obtenu par l'application du modèle de distorsion. Si les coordonnées de ce point ne sont pas entières, le pixel est colorié par une interpolation bilinéaire entre les plus proches voisins.

Pour plus d'efficacité, la fonction de distorsion en chaque pixel $\delta(x, y)$ n'est calculée qu'une fois, et non à chaque nouvelle image fournie par la caméra. En effet, la correction (le déplacement) à effectuer sur un même pixel est toujours identique et peut donc être évaluée une fois, puis appliquée sur chaque nouvelle image. Une manière synthétique de stocker la fonction δ en chaque pixel consiste à construire une « carte de correction », puis de l'appliquer à chaque nouvelle image.



(a) Projection centrale (b) Projection non-centrale

FIG. 2.6: Le modèle perspectif linéaire (a) convient pour la plupart des caméras standards. Mais certaines configurations, comme les très grands angles, ou les caméras catadioptriques (b) requièrent un modèle plus générique.

Remarque : Pour gagner du temps, la correction de la distorsion peut être appliquée en même temps que d'autres transformations de l'image, comme par exemple la rectification (chapitre 3), en additionnant les transformations dans l'étape 2.

2.1.3 Modèle générique

Description

Le modèle perspectif linéaire (cf. 2.1.1), même associé à un modèle de distorsion non-linéaire, ne permet pas de modéliser toutes les caméras. En effet, l'utilisation de ces deux modèles pré-suppose que la caméra et son optique suivent une projection centrale (dépeint sur la figure 2.6(a)), dans laquelle tous les rayons optiques s'intersectent en un point unique : le centre optique. Mais lorsque cette hypothèse n'est pas vérifiée, comme par exemple dans le cas de caméras catadioptriques (voir la figure 2.6(b)), le modèle perspectif linéaire n'est plus applicable.

Il faut alors préférer un modèle plus général, comme celui proposé par Ramalingam *et al.* [RSL05]. Ce dernier représente la caméra par un ensemble de rayons de projection [GN01], contrairement aux modèles paramétriques qui synthétisent la fonction de projection par quelques paramètres (distance focale, coefficients de distorsion, etc.). Une caméra consiste en un ensemble de pixels. Chaque pixel capte la lumière qui se propage le long d'un rayon appelé rayon de projection ou rayon optique. Une caméra est alors complètement modélisée par :

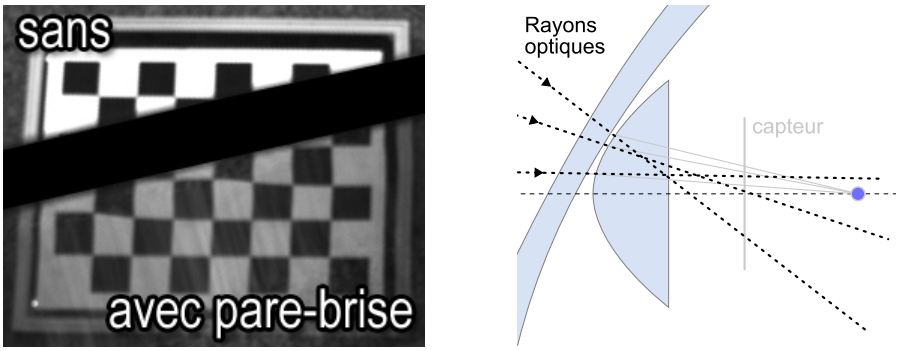
- les coordonnées de ces rayons (en 3-D, données par rapport à un repère local de la caméra) ;
- la correspondance entre pixels et rayons.

Ce modèle général permet de décrire la plupart des types de caméras existants (du moins celles qui opèrent dans le domaine visible), incluant les caméras sténo-

pées (avec des distorsions), les caméras catadioptriques (utilisant des systèmes à miroirs), etc. Le modèle de projection centrale est donc un cas particulier du modèle de projection général, dans lequel tous les rayons optiques s'intersectent en un point unique.

Validation de l'hypothèse de modèle sténopé

Le but est d'étudier le modèle d'un système optique composé de l'association d'un pare-brise et d'une caméra. En effet, le pare-brise se comportant comme un système optique complexe, engendre des distorsions (voir la figure 2.7(a)). Dès lors, il est possible, comme l'illustre la figure 2.7(b), que l'hypothèse de projection centrale ne soit pas valide, empêchant de fait l'utilisation d'un modèle sténopé. Bien que le modèle sténopé soit largement utilisé dans le domaine automobile, l'hypothèse de projection centrale n'a, à notre connaissance, jamais été vérifiée en présence d'un pare-brise.



(a) (b)

Fig. 2.7: La distorsion engendrée par le pare-brise (a) peut invalider l'hypothèse de projection centrale (b).

Nous présentons l'expérimentation mise en œuvre pour comprendre le modèle de projection de la caméra lorsque'elle observe la scène au travers du pare-brise. Plus précisément, nous avons vérifié que le pare-brise de notre véhicule d'essai[1] (voir la figure 2.8) ne perturbe pas le système optique au point de le rendre non central. L'expérimentation consiste en une série de calibrages avec et sans pare-brise, sous l'hypothèse d'un modèle à projection centrale. La qualité du calibrage obtenu est ensuite comparée avec et sans pare-brise.

L'acquisition de 3 poses différentes de la mire de calibrage (voir la figure 2.9) suffit à déterminer les rayons optiques associés à chaque pixel de l'image.

La qualité du calibrage ainsi obtenu est déterminée par l'erreur quadratique moyenne (RMS). Cette erreur est caractérisée par la distance d_i entre chaque rayon optique l_i reconstruit et le point $\mathbf{m}_i$ de la mire qui lui correspond. l'erreur RMS s'écrit :

$$RMS = \sqrt{\frac{1}{n}\sum_{i=1}^{n} d_i^2}.$$

[1] Un pare-brise de véhicule Renault VeSatis

Fig. 2.8: La caméra est placée en haut, proche du rétroviseur central.

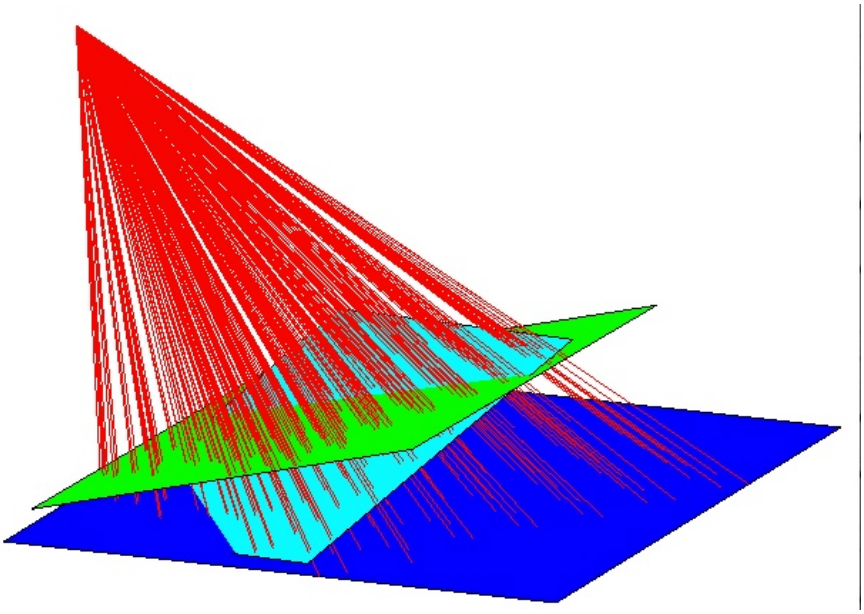


Fig. 2.9: Reconstruction 3-D des trois positions de la mire plane (en vert et bleu) et des rayons optiques (en rouge).

distance focale	6 mm		8 mm	
pare-brise	sans	avec	sans	avec
distance max (cm)	79.22	186.79	121.90	73.46
RMS (cm)	0.0145	0.0303	0.0281	0.0120
***RMS* normalisée**	**0.0183**	**0.0162**	**0.0231**	**0.0163**

TAB. 2.1: Comparaison de l'erreur *RMS* avec et sans pare-brise pour des objectifs de 6 mm et 8 mm de distance focale. L'erreur étant proportionnelle à la distance, elle doit être normalisée par la distance (maximum) de la mire aux caméras.

Deux objectifs, d'une distance focale de 6 et 8mm, ont été testés. L'erreur obtenue (se référer au tableau 2.1) est d'autant plus grande que les points de la mire sont distants des caméras. Pour obtenir une mesure comparable entre les différents calibrages, il est donc nécessaire d'uniformiser l'erreur *RMS* pour les différentes distances des points de la mire. L'erreur étant proportionnelle à la distance, nous divisons l'erreur *RMS* par cette dernière, pour obtenir ce que nous appelons l'erreur *RMS* normalisée.

Retenons que l'erreur *RMS* normalisée est très similaire avec ou sans le pare-brise. En d'autres termes, un modèle à projection centrale (sténopé + distorsion non-linéaire) suffit à modéliser les caméras embarquées. A noter cependant qu'on ne connait pas le modèle de distorsion non-linéaire à mettre en œuvre.

2.1.4 Paramètres extrinsèques

Les paramètres extrinsèques du modèle expriment la transformation liée à la pose de la caméra par rapport au repère monde ($\mathcal{W}$). Car, dans la partie intrinsèque, la fonction de projection perspective (équation 2.1) présuppose que le système de coordonnées, dans lequel sont exprimés les points 3-D, est centré sur la caméra ($\mathcal{C}$). Par la suite, nous aurons besoin d'exprimer les coordonnées des points 3-D dans un référentiel $\mathcal{R}_{\mathrm{W}}$ (système, voiture, monde, etc), différent de $\mathcal{R}_{\mathrm{C}}$. Pour pouvoir appliquer la fonction de projection, il est nécessaire de transformer les coordonnées $\mathbf{M}$ de manière à ce qu'elles soient exprimées dans le repère caméra ($\mathcal{R}_{\mathrm{C}}$). Nous notons ${}^{\mathcal{W}}\mathbf{M}$ un point exprimé dans le repère monde, et ${}^{\mathcal{C}}\mathbf{M}$ le même point exprimé dans le repère de la caméra.

Quelle que soit la transformation, elle peut toujours être décomposée en une rotation suivie d'une translation (voir la figure 2.10). L'ordre dans lequel sont appliquées ces deux opérations est très important car l'opération n'est pas commutative. Dans la suite de cette thèse, la fonction de rotation sera notée R et la fonction de translation $\mathbf{t}$. Un point initialement exprimé dans le repère $\mathcal{W}$ est converti dans le système de coordonnées $\mathcal{C}$ par la transformation suivante :

$$ {}^{\mathcal{W}}\mathbf{M} = {}^{\mathcal{W}}_{\mathcal{C}}\mathsf{R}\ {}^{\mathcal{C}}\mathbf{M} + {}^{\mathcal{W}}_{\mathcal{C}}\mathbf{t} \tag{2.10} $$

où $\mathbf{t}$ est un vecteur à trois composantes, et R est une matrice de taille 3×3.

Remarque : Il y a souvent confusion entre les paramètres définissant la pose de la caméra (position et orientation) et ceux de la fonction de changement de repère (rotation et translation). Bien que ces deux jeux de paramètres

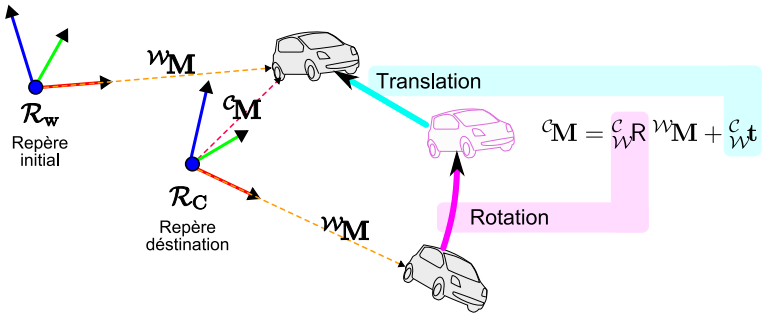


FIG. 2.10: Le changement de repère est une transformation rigide composée d'une rotation puis d'une translation.

dépendent des mêmes caractéristiques, il ne sont pas égaux. La relation entre l'un et l'autre est développée dans la partie 2.1.4 (p.23)

Formulation des rotations

Autant la translation est toujours exprimée sous la forme d'un vecteur à trois composantes, autant il existe une multitude de représentations possibles pour une rotation. Pour en citer quatre : la matrice de rotation, les angles d'*Euler*, le vecteur de *Rodrigues* et les quaternions. Le tableau 2.11 mentionne quelles sont les applications pour lesquelles ces représentations sont adaptées. Dans cette partie nous présentons les trois premières formulations.

représentation	degrés de liberté	applications
la matrice de rotation	9	adaptée au calcul informatique
les angles d'Euler	3	représentation intuitive souvent utilisée par les logiciels de modélisations comme maya ou 3DSMax,
le vecteur de Rodrigues	3	représentation intuitive adaptée aux méthodes de minimisation non-linéaire
les quaternions	4	représentation adaptée aux interpolations

FIG. 2.11: Les différentes représentations de la rotation avec leurs spécificités.

La Matrice de rotation est de taille 3×3, de la forme suivante :

$$\mathrm{R} = \begin{bmatrix} \mathrm{R}_{11} & \mathrm{R}_{12} & \mathrm{R}_{13} \\ \mathrm{R}_{21} & \mathrm{R}_{22} & \mathrm{R}_{23} \\ \mathrm{R}_{31} & \mathrm{R}_{32} & \mathrm{R}_{33} \end{bmatrix}. \tag{2.11}$$

Elle décrit une rotation autour de l'origine du repère. La rotation ayant seulement 3 degrés de liberté, les neuf paramètres de la matrices sont contraints.

Ainsi, les vecteurs colonnes (ou lignes) sont orthogonaux entre eux, induisant la propriété suivante :

$$\mathsf{R}^{\mathsf{T}}\mathsf{R} = I \Leftrightarrow \mathsf{R}^{\mathsf{T}} = \mathsf{R}^{-1} \tag{2.12}$$

Les matrices suivantes (2.13) modélisent les rotations canoniques autour de chacun des trois axes X, Y et Z :

$$\mathsf{R}_{\mathrm{x}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}, \quad \mathsf{R}_{\mathrm{y}} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}, \quad \mathsf{R}_{\mathrm{z}} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.13}$$

Les angles d'Euler décrivent une rotation quelconque comme la composition de trois rotations élémentaires appelées précession (ψ), nutation (θ) et rotation propre (ϕ). Ces trois rotations élémentaires correspondent à la composition $\mathsf{R}_{\mathrm{z}} \times \mathsf{R}_{\mathrm{x}} \times \mathsf{R}_{\mathrm{z}}$, avec R_{z} et R_{x} définies dans l'équation 2.13. Les paramètres de la matrice de rotation donnés par l'équation 2.11 sont :

$$\begin{aligned}
&\mathsf{R}_{11} = \cos\phi\cos\psi - \sin\phi\cos\theta\cos\psi && \mathsf{R}_{12} = -\cos\phi\cos\psi - \cos\phi\cos\theta\sin\psi \\
&\mathsf{R}_{13} = \sin\theta\sin\psi && \mathsf{R}_{21} = \cos\phi\sin\psi + \sin\phi\cos\theta\cos\psi \\
&\mathsf{R}_{22} = -\sin\phi\sin\psi + \cos\phi\cos\theta\cos\psi && \mathsf{R}_{23} = -\sin\theta\cos\psi \\
&\mathsf{R}_{31} = \sin\phi\sin\theta && \mathsf{R}_{32} = \cos\phi\sin\theta \\
&\mathsf{R}_{33} = \cos\theta
\end{aligned}$$

Vecteur de Rodrigues ω est un vecteur à 3 composantes. Il fournit une représentation compacte, efficace et très intuitive des fonctions de rotation. La direction du vecteur définit l'axe de la rotation, et la norme définit la valeur de l'angle. Contrairement à une représentation sous forme matricielle, n'importe quel ensemble des paramètres définissent une et une seule rotation valide. Il n'est donc pas nécessaire de vérifier la validité d'un jeu de paramètres. Ce qui en fait un modèle adapté aux méthodes d'optimisations, qui explorent l'espace des paramètres.

$$\mathsf{R} = \mathsf{I} + \tilde{\omega}\sin\theta + \tilde{\omega}^2(1-\cos\theta), \tag{2.14}$$

avec I la matrice identité, θ la valeur de l'angle, l'axe de rotation défini par le vecteur unitaire $\hat{\omega} = (\omega_x, \omega_y, \omega_z)$ et

$$\tilde{\omega} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}. \tag{2.15}$$

En développant l'équation 2.14, on obtient R en fonction du vecteur unitaire $(\omega_x, \omega_y, \omega_z)$ et de l'angle θ :

$$\begin{aligned}
&\mathsf{R}_{11} = \cos\theta + {\omega_x}^2(1-\cos\theta) && \mathsf{R}_{12} = \omega_x\omega_y(1-\cos\theta) - \omega_x\sin\omega_z \\
&\mathsf{R}_{13} = \omega_y\sin\theta + \omega_x\omega_z(1-\cos\theta) && \mathsf{R}_{21} = omega_z\sin\theta + \omega_x\omega_y(1-\cos\theta) \\
&\mathsf{R}_{22} = \cos\theta + {\omega_z}^2(1-\cos\theta) && \mathsf{R}_{23} = -\omega_x\sin\theta + \omega_y\omega_z(1-\cos\theta) \\
&\mathsf{R}_{31} = -\omega_y\sin\theta + \omega_x\omega_z(1-\cos\theta) && \mathsf{R}_{32} = \omega_x\sin\theta + \omega_y\omega_z(1-\cos\theta) \\
&\mathsf{R}_{33} = cos\theta + {\omega_z}^2(1-\cos\theta)
\end{aligned}$$

Relation entre les paramètres extrinsèques et la pose de la caméra

Comme mentionnée en début de cette partie, il existe une relation entre les paramètres extrinsèques et la pose de la caméra. Mais même si la rotation (respectivement la translation) est formulée de manière similaire à l'orientation (respectivement la position), ces deux données ne sont pas égales. Plus précisément, la position de la caméra (la position de son centre optique C dans le référentiel monde $\mathcal{R}_{\mathrm{W}}$) est fonction de la rotation R et de la translation $\mathbf{t}$. Étant donné que C est au centre du référentiel caméra, la position de la caméra se déduit comme suit :

$$\mathsf{R}\ {}^{\mathcal{W}}C + \mathbf{t} = {}^{\mathcal{C}}0 \tag{2.16}$$

$$\mathsf{R}\ {}^{\mathcal{W}}C = -\mathbf{t} \tag{2.17}$$

$${}^{\mathcal{W}}C = -\mathsf{R}^{\mathsf{T}}\mathbf{t} \tag{2.18}$$

Les différents repères

De la caméra au monde, en passant par le véhicule, chaque élément peut être associé un référentiel. Le référentiel monde $\mathcal{R}_{\mathrm{W}}$ peut être choisi arbitrairement. Il est possible d'utiliser celui du GPS[2], cependant il présente peu d'intérêt pour des applications comme l'ACC[3]. Car pour ce type d'application, seul l'entourage

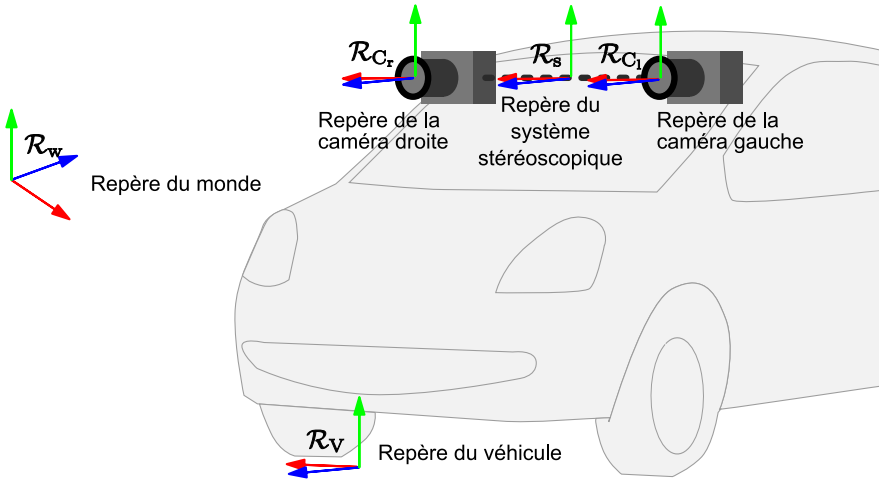


Fig. 2.12: De la caméra au monde, en passant par le véhicule, à chaque élément (physique ou conceptuel) peut être associé un référentiel.

immédiat du véhicule est pertinent, et un repère $\mathcal{R}_{\mathrm{V}}$ attaché au véhicule est plus judicieux. $\mathcal{R}_{\mathrm{V}}$ permet d'exprimer la position et/ou la vitesse des obstacles relativement au véhicule. $\mathcal{R}_{\mathrm{V}}$ est généralement centré sur l'essieu arrière du véhicule, car cette pièce non-suspendue présente l'avantage d'être physiquement identifiable et localisable. Cependant nous préférons aligner ce repère à l'aplomb du pare-choc avant (voir la figure 2.12), car il permet ainsi d'exprimer la distance

[2]Global Positioning System
[3]Adaptative Cruise Control

avant le choc. Comme sur la figure 2.13, nous choisissons d'aligner l'axe X à l'essieu du véhicule, l'axe Z sur l'axe longitudinal (sens du déplacement), et l'axe Y, orthogonal aux deux autres, sur l'axe de la verticale.

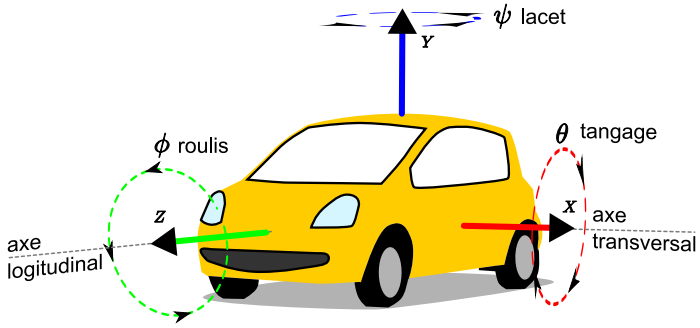


FIG. 2.13: La dénomination des axes du repère véhicule, avec les angles associés.

A la caméra gauche (respectivement droite) est associée un référentiel $\mathcal{R}_{\mathrm{C_l}}$ (respectivement $\mathcal{R}_{\mathrm{C_r}}$), nécessaire à la fonction de projection (se référer à la partie 2.1.1). Comme nous le détaillons dans la partie suivante (2.2.1), il est avantageux d'introduire un nouveau référentiel $\mathcal{R}_{\mathrm{S}}$ propre au système stéréoscopique.

2.2 Une paire de caméras

Lors de la projection d'un point 3-D sur le capteur photosensible de la caméra, l'information de profondeur est perdue. C'est à dire qu'à un point de l'image correspond une infinité de points 3-D, répartis le long du rayon optique (la droite 3-D passant par le centre optique et le point image). Grâce à l'ajout d'une deuxième caméra, la reconstruction de la profondeur devient possible en calculant l'intersection des deux rayons optiques (un par caméra) correspondant à un même point 3-D. Par conséquent, il est nécessaire d'exprimer la géométrie des deux capteurs dans un même référentiel. Deux instances du modèle sténopé (que nous venons de détailler) suffiraient à modéliser ce couple de caméras. Mais, avec cette représentation, les deux caméras seraient considérées indépendamment l'une de l'autre. Cette représentation présente deux autres inconvénients. Tout d'abord, elle ne permet pas d'exprimer, et donc de comprendre, la relation physique liant les deux capteurs (ligne de base, orientation relative, etc.). De plus, le modèle sténopé n'est pas utilisé par toutes les méthodes de calibrage.

Dans cette partie, nous allons donc présenter deux autres formulations d'un système stéréoscopique permettant de résoudre les deux limitations sus-mentionnées. La première est une formulation originale intuitive. Elle reprend les paramètres intrinsèques du modèle sténopé, et remanie les paramètres extrinsèques de manière à séparer les paramètres internes au système (relation entre les deux caméras) de ceux externes au système (position de la tête stéréo dans le monde). Cette représentation permet de comprendre plus facilement la relation géométrique qui existe entre les deux capteurs.

La deuxième formulation, très répandue dans la littérature, exprime directement la relation entre les deux images, appelée contrainte épipolaire. La prise en

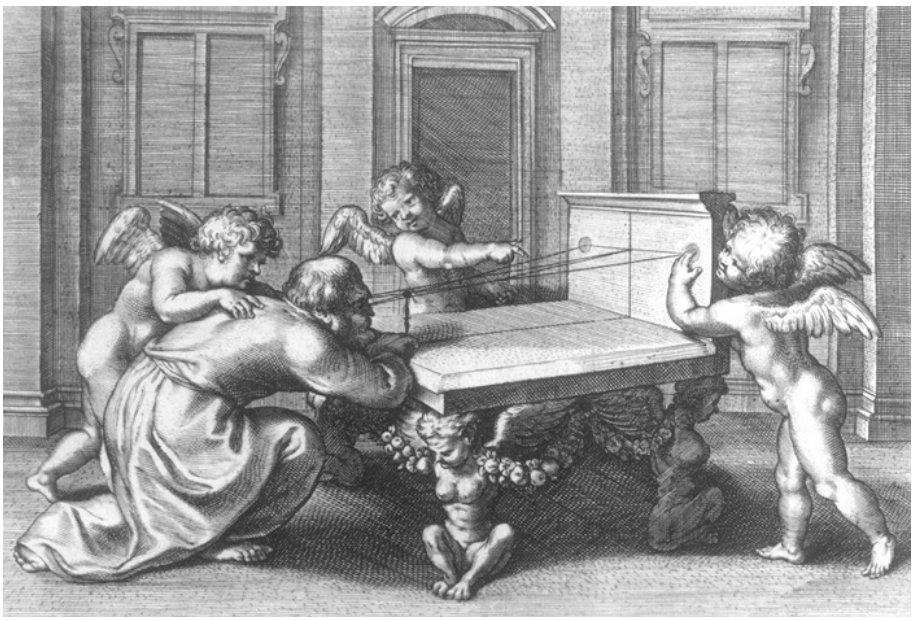


FIG. 2.14: Gravure du peintre *Rubens*, pour le compte du mathématicien *Aguilon*, mettant en avant les propriétés de la stéréoscopie.

compte de cette contrainte est très importante, puisqu'elle améliore et accélère le processus stéréoscopique.

2.2.1 Géométrie interne de la tête stéréoscopique

Comme nous l'avons vu dans la partie précédente, les paramètres extrinsèques du modèle sténopé définissent la fonction de changement de repère transformant des points de $\mathcal{R}_{\mathrm{W}}$ vers $\mathcal{R}_{\mathrm{C}}$. Ces paramètres informent sur la position et l'orientation de chaque caméra par rapport au référentiel monde. Cette information permet notamment d'analyser les déplacements des caméras au cours du temps. D'un point de vue technique, il est intéressant de différencier les mouvements du système (mouvement normal du véhicule) d'une variation de la configuration interne du système. Pour caractériser la géométrie interne de la tête stéréoscopique, nous définissons un nouveau référentiel $\mathcal{R}_{\mathrm{S}}$ permettant de mettre en évidence la relation entre les deux caméras. Une possibilité est de définir la première caméra comme référence, et de caractériser la deuxième par rapport à la première. Toutefois, la ligne de base (le segment reliant les deux centres optiques $\mathrm{C_l}$ et $\mathrm{C_r}$), correspondant à l'armature rigide reliant les deux caméras, semble fournir une référence plus appropriée. Notre nouveau système de paramètres défini à partir de la ligne de base se divise en 6 paramètres internes et 6 (3+3) paramètres supplémentaires pour définir la pose du système.

Comme le dépeint la figure 2.15, les 6 paramètres internes sont :

b : la longueur de la ligne de base,

ϕ_i : les angles de lacet de chaque caméra,

ψ_i : les angles de roulis de chaque caméra,

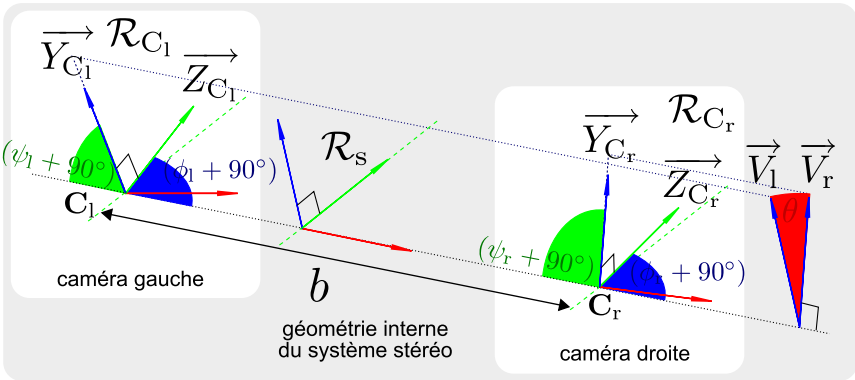


FIG. 2.15: Géométrie interne d'une tête stéréoscopique. Les angles de lacet ϕ et de roulis ψ de chaque caméra sont exprimés par rapport à la ligne de base. L'angle de tangage θ est un angle relatif entre les deux caméras.

θ : l'angle de tangage relatif d'une caméra par rapport à l'autre.

Le repère externe du système est arbitrairement centré sur la ligne de base, avec son axe X aligné avec cette dernière. Il reste à fixer la verticale du repère $\mathcal{R}_S$. Nous choisissons de l'aligner sur celui de la caméra gauche. Les paramètres internes que nous venons de décrire peuvent s'écrire en fonction des paramètres extrinsèques de la façon suivante suivante :

$$b = \|C_r - C_l\|, \qquad \overrightarrow{B} = \frac{C_r - C_l}{b},$$
$$\phi_i = \arccos(\overrightarrow{Z_{C_i}} \cdot \overrightarrow{B}) - 90°, \qquad \psi_i = \arccos(\overrightarrow{Y_{C_i}} \cdot \overrightarrow{B}) - 90°,$$
$$\overrightarrow{V_i} = \frac{\overrightarrow{Z_{C_i}} \times \overrightarrow{B}}{\|\overrightarrow{Z_{C_i}} \times \overrightarrow{B}\|}, \qquad \theta = \arccos(\overrightarrow{V_l} \cdot \overrightarrow{V_r}),$$

où tous les angles sont exprimés en degrés ou de force, $\times$ est l'opérateur du produit vectoriel et $\cdot$ l'opérateur de produit scalaire. L'origine $O_{\mathcal{S}}$, et les axes $(\overrightarrow{X_{\mathcal{S}}}, \overrightarrow{Y_{\mathcal{S}}}, \overrightarrow{Z_{\mathcal{S}}})$ du repère $\mathcal{R}_S$ (voir la figure 2.15), exprimés dans le repère monde, s'écrivent :

$$^{w}O_{\mathcal{S}} = \frac{^{w}C_r - {}^{w}C_l}{2}, \qquad \overrightarrow{X_{\mathcal{S}}} = \overrightarrow{B},$$
$$\overrightarrow{Y_{\mathcal{S}}} = \overrightarrow{V_l} \qquad \overrightarrow{Z_{\mathcal{S}}} = \overrightarrow{X_{\mathcal{S}}} \times \overrightarrow{Y_{\mathcal{S}}}.$$

2.2.2 Contrainte épipolaire

La géométrie épipolaire est applicable à tout système couplant deux caméras, et ce, quelque soit le modèle utilisé. Elle exprime la relation qui existe entre les coordonnées en pixels d'un point dans une image et son correspondant dans l'autre. Nous commençons par expliquer l'aspect géométrique de cette contrainte, avant de développer l'aspect algébrique, formalisé par la matrice fondamentale.

Le point de vue géométrique

Comme le montre la figure 2.16, les centres optiques $\mathrm{C_l}$ et $\mathrm{C_r}$, le point 3-D $\mathbf{M}$ et ses projections $\mathbf{m_l}$ et $\mathbf{m_l}$ sont tous contenus dans un plan Ω, appelé plan épipolaire (en violet sur la figure 2.16). Étant donnés $\mathrm{C_l}$ et le point 2-D $\mathbf{m_l}$, on peut construire le rayon optique issu de $\mathbf{m_l}$, c'est à dire la demi-droite partant du centre optique $\mathrm{C_l}$ et passant par $\mathbf{m_l}$. Par définition, le point 3-D $\mathbf{M}$ appartient au rayon optique issu de $\mathbf{m_l}$. Par conséquent, la projection $\mathbf{m_r}$ de $\mathbf{M}$ dans l'image I_r se situe nécessairement sur la projection du rayon optique. Cette demi-droite de l'image, notée l_r est appelé droite épipolaire, et c'est elle qui définit la contrainte épipolaire.

Quel que soit $\mathbf{M}$, le plan épipolaire Ω qui lui est associé contient la ligne de base (le segment reliant $\mathrm{C_l}$ et $\mathrm{C_r}$). Étant donné que les droites épipolaires sont l'intersection de Ω avec les plans images, alors toutes les droites épipolaires d'une image passent par un unique point e : l'épipole. Les épipoles, notés e_l et e_r, correspondent à l'intersection de la ligne de base avec les deux plans images.

Remarque : Notons que, même si un épipole appartient au plan image, il est souvent en dehors du cadre de l'image (du champ de vue).

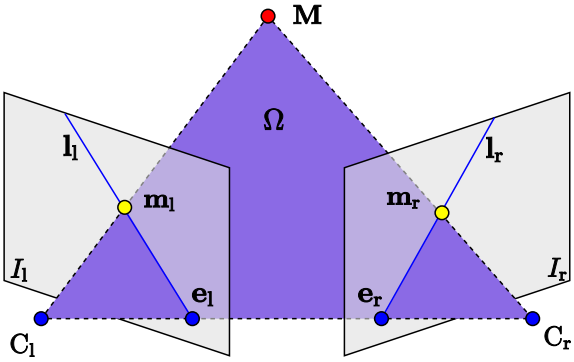


FIG. 2.16: La contrainte épipolaire exprime le fait que pour un point $\mathbf{m_l}$ donné dans une image, son correspondant dans l'autre image se situe nécessairement sur la droite épipolaire.

Le point de vue algébrique : la matrice fondamentale

La matrice fondamentale F est l'expression algébrique de coplanarité des points $(\mathbf{m_l}, \mathbf{m_r}, \mathrm{C_l}, \mathrm{C_r})$, appelé contrainte épipolaire. Pour calculer F, commençons par décrire les matrices de projections $\mathsf{P_l}$ et $\mathsf{P_r}$. Pour simplifier les calculs, nous imposons que le repère du système de coordonnées de référence coïncide avec celui de la caméra de droite (on comprend facilement que la matrice fondamentale est indépendante du choix du système de coordonnées euclidien) :

$$\mathsf{P_l} = \mathsf{K_l}[\mathsf{R}\mathbf{t}] \qquad \text{et} \qquad \mathsf{P_r} = \mathsf{K_r}[I0] \tag{2.19}$$

et les coordonnées de projections de $\mathbf{M}$ dans les deux images sont :

$$\mathbf{m}_\mathrm{l} \sim \mathsf{P}_\mathrm{l}\mathbf{M} \qquad \text{et} \qquad \mathbf{m}_\mathrm{r} \sim \mathsf{P}_\mathrm{r}\mathbf{M}. \tag{2.20}$$

En éliminant $\mathbf{M}$ des équations précédentes, nous obtenons :

$$\mathbf{m}_\mathrm{l}{}^\mathsf{T}\mathsf{F}_\mathrm{lr}\mathbf{m}_\mathrm{r} = 0 \qquad \text{avec} \qquad \mathsf{F}_\mathrm{lr} = \mathsf{K}_\mathrm{l}{}^{-\mathsf{T}}[\mathbf{t}]_\times \mathsf{R}\mathsf{K}_\mathrm{r}{}^{-1} \tag{2.21}$$

où $[\mathbf{t}]_\times$ est la matrice antisymétrique définie par $[\mathbf{t}]_\times\mathbf{x} = \mathbf{t} \times \mathbf{x}$ quel que soit le vecteur $\mathbf{x}$. Cette matrice possède quelques propriétés intéressantes : puisque $\det[\mathbf{t}]_\times\mathbf{x} = 0$, alors $\det \mathsf{F}_\mathrm{lr} = 0$ et F_lr est de rang 2. De plus F est définie à un facteur d'échelle près puisque l'équation 2.21 est toujours valable quand on multiplie F_lr par un facteur quelconque. Nous pouvons également montrer qu'en échangeant le rôle des deux caméras (c'est à dire en fixant la première comme référence), l'équation 2.21 devient $\mathbf{m}_\mathrm{r}{}^\mathsf{T}\mathsf{F}_\mathrm{rl}\mathbf{m}_\mathrm{l} = 0$ et que $\mathsf{F}_\mathrm{rl} \sim \mathsf{F}_\mathrm{lr}{}^\mathsf{T}$. Pour de plus amples informations sur la matrice fondamentale, se référer à l'ouvrage de *Hartley* et *Zisserman* [HZ04].

Dans le cas où les caméras suivent un modèle projectif linéaire (ou qu'une correction de distorsion permet de se ramener à ce cas), des paires de points suffisent à calculer la matrice F. Ce cas à été largement traité dans la revue des algorithmes de calibrage faible par Zhang [Zha96] et Salvi *et al.* [SAP01], et nous ne nous étendrons pas sur le sujet. Nous mentionnons simplement l'existence de la méthode calibrage faible robuste aux données aberrantes de Torr *et al.* [TM97].

2.2.3 Reconstruction 3-D

La reconstruction consiste à calculer la position du point 3-D à partir de ses images. L'approche classique, dite de « triangulation », consiste à calculer l'intersection dans l'espace des rayons optiques associés aux points dans les images. Il existe plusieurs niveaux de reconstruction [Fau92] qui dépendent des modalités de calibrage (calibrage fort, calibrage faible, rectification, etc). Un système entièrement étalonné peut permettre une reconstruction euclidienne, alors qu'une simple rectification des images (via un calibrage faible) n'autorise qu'une reconstruction projective [Fau92].

Reconstruction projective

Si le système est faiblement calibré, on doit se limiter à une reconstruction projective, de l'espace. C'est à dire que la géométrie de la scène reconstruite est connue à une collinéation (homographie) de l'espace près. Dans cet espace, il n'y a ni notion de distances, ni de rapports de distances. L'invariant fondamental de l'espace projectif est le birapport. Pour plus de détails, se référer à *Projective Reconstruction from Line Correspondences* [Har94].

Triangulation euclidienne

La reconstruction euclidienne n'est possible qu'avec la connaissance de tous les paramètres du système (une des possibilités pour les obtenir est de calibrer fortement le système). L'approche classique de reconstruction, dite de « triangulation », consiste à calculer l'intersection dans l'espace des rayons optiques associés aux points dans les images.

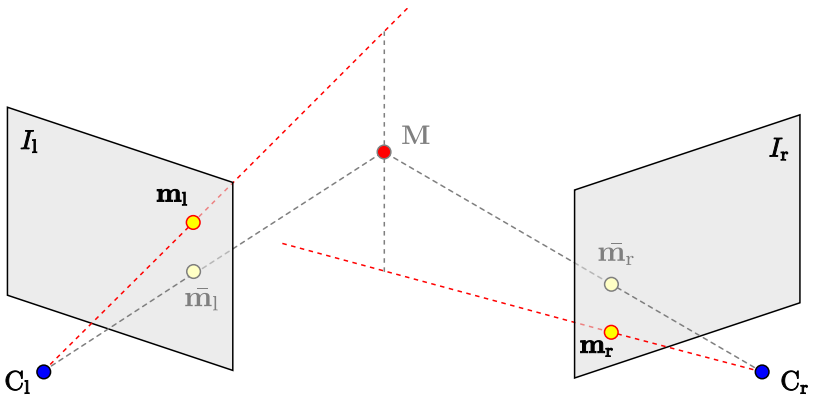


FIG. 2.17: Lorsque la contrainte épipolaire n'est pas respectée, les rayons optiques issus des observations $\mathbf{m}_l$ et $\mathbf{m}_r$ ne s'intersectent pas. Il est donc impossible de reconstruire le point 3-D $\mathbf{M}$ correspondant aux observations. Toutefois, il est possible d'en définir un, plus approximatif, dont les projections $\bar{\mathbf{m}}_l$ et $\bar{\mathbf{m}}_r$ soient proches des observations initiales.

Mais quand la contrainte épipolaire n'est pas respectée, les rayons optiques associés ne s'intersectent pas (figure 2.17) et il n'existe aucune reconstruction stricte possible. En pratique, à cause d'imperfections de calibrage ou de mise en correspondance, il est rare que les rayons optiques s'intersectent vraiment. Le point 3-D est alors reconstruit « au mieux », en choisissant le point 3-D le « plus proche ». Cette notion de « proximité » varie selon les méthodes de triangulation. La première des trois méthodes de triangulation que nous présentons [HZ04], qui est aussi la plus intuitive, consiste à trouver le segment le plus court reliant les deux rayons optiques et d'en choisir le milieu comme point de reconstruction. Pour trouver ce point, chaque rayon optique est parcouru afin de minimiser la distance entre ces points situés sur les deux rayons optiques. Soient $\mathbf{M}$ le point reconstruit, C_j les centres optiques, V_j les rayons optiques, et s_j l'ordonnée de chaque point sur chaque rayon. Ce calcul de minimisation est direct dans le cas où il n'y a que deux caméras.

La deuxième méthode, linéaire, consiste à résoudre le système d'équations données par les projections suivantes :

$$\begin{cases} x_l = \dfrac{m_{l00}X + m_{l01}Y + m_{l02}Z + m_{l03}}{m_{l20}X + m_{l21}Y + m_{l22}Z + 1} \\ y_l = \dfrac{m_{l10}X + m_{l11}Y + m_{l12}Z + m_{l13}}{m_{l20}X + m_{l21}Y + m_{l22}Z + 1} \\ x_r = \dfrac{m_{r00}X + m_{r01}Y + m_{r02}Z + m_{r03}}{m_{r20}X + m_{r21}Y + m_{r22}Z + 1} \\ y_r = \dfrac{m_{r10}X + m_{r11}Y + m_{r12}Z + m_{r13}}{m_{r20}X + m_{r21}Y + m_{r22}Z + 1} \end{cases} \tag{2.22}$$

où m_{ijk} est l'élément (j, k) de la matrice de projection de la caméra i, x_l, y_l, x_r et y_r sont les coordonnées 2-D des points projetés, et X, Y et Z sont les coordonnées 3-D du point reconstruit. Le système d'équation 2.22, se factorise

comme suit :

$$\begin{cases} 1 - m_{\mathrm{l}03} = (m_{\mathrm{l}00} - x_{\mathrm{l}} m_{\mathrm{l}20})X + (m_{\mathrm{l}01} - x_{\mathrm{l}} m_{\mathrm{l}21})Y + (m_{\mathrm{l}02} - x_{\mathrm{l}} m_{\mathrm{l}22})Z \\ 1 - m_{\mathrm{l}13} = (m_{\mathrm{l}10} - y_{\mathrm{l}} m_{\mathrm{l}20})X + (m_{\mathrm{l}11} - y_{\mathrm{l}} m_{\mathrm{l}21})Y + (m_{\mathrm{l}12} - y_{\mathrm{l}} m_{\mathrm{l}22})Z \\ 1 - m_{\mathrm{r}03} = (m_{\mathrm{r}00} - x_{\mathrm{r}} m_{\mathrm{r}20})X + (m_{\mathrm{r}01} - x_{\mathrm{r}} m_{\mathrm{r}21})Y + (m_{\mathrm{r}02} - x_{\mathrm{r}} m_{\mathrm{r}22})Z \\ 1 - m_{\mathrm{r}13} = (m_{\mathrm{r}10} - y_{\mathrm{r}} m_{\mathrm{r}20})X + (m_{\mathrm{r}11} - y_{\mathrm{r}} m_{\mathrm{r}21})Y + (m_{\mathrm{r}12} - y_{\mathrm{r}} m_{\mathrm{r}22})Z \end{cases} \tag{2.23}$$

Exprimé sous forme matricielle, le système d'équations 2.23 s'écrit :

$$\underbrace{\begin{bmatrix} 1 - m_{\mathrm{l}03} \\ 1 - m_{\mathrm{l}13} \\ 1 - m_{\mathrm{r}03} \\ 1 - m_{\mathrm{r}13} \end{bmatrix}}_{\mathbf{b}} = \underbrace{\begin{bmatrix} m_{\mathrm{l}00} - x_{\mathrm{l}} m_{\mathrm{l}20} & m_{\mathrm{l}01} - x_{\mathrm{l}} m_{\mathrm{l}21} & m_{\mathrm{l}02} - x_{\mathrm{l}} m_{\mathrm{l}22} \\ m_{\mathrm{l}10} - y_{\mathrm{l}} m_{\mathrm{l}20} & m_{\mathrm{l}11} - y_{\mathrm{l}} m_{\mathrm{l}21} & m_{\mathrm{l}12} - y_{\mathrm{l}} m_{\mathrm{l}22} \\ m_{\mathrm{r}00} - x_{\mathrm{r}} m_{\mathrm{r}20} & m_{\mathrm{r}01} - x_{\mathrm{r}} m_{\mathrm{r}21} & m_{\mathrm{r}02} - x_{\mathrm{r}} m_{\mathrm{r}22} \\ m_{\mathrm{r}10} - y_{\mathrm{r}} m_{\mathrm{r}20} & m_{\mathrm{r}11} - y_{\mathrm{r}} m_{\mathrm{r}21} & m_{\mathrm{r}12} - y_{\mathrm{r}} m_{\mathrm{r}22} \end{bmatrix}}_{\mathsf{A}} \underbrace{\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}}_{\mathbf{X}}.$$

Ce système sur-contraint peut alors être résolu de la manière suivante :

$$\mathsf{A}^{\#}\mathbf{b} = \left(\mathsf{A}^{\mathsf{T}}\mathsf{A}\right)^{-1}\mathsf{A}^{\mathsf{T}}\mathbf{b}.$$

La dernière méthode de triangulation, est une moindres carrés non-linéaire dont le but est de trouver un point le plus « proche », non plus dans l'espace euclidien (comme c'est le cas pour la première méthode), mais dans l'espace image. La distance est donc cette fois caractérisée par l'erreur de reprojection (pour plus de détail se référer à la partie suivante). Une méthode de minimisation non-linéaire, telle que celle de *Levenberg-Marquardt*, est adaptée dans ce cas, car optimale sous l'hypothèse d'un bruit gaussien sur la mesure 2-D. La fonction $f(\mathbf{M})$ à minimiser est :

$$f(\mathbf{M}) = \sum_{j=\{\mathrm{l},\mathrm{r}\}} d(\mathbf{m}_j, \bar{\mathbf{m}}_j)^2, \tag{2.24}$$

avec $d()$ la fonction de distance 2-D et $\bar{\mathbf{m}}_j = \mathsf{P}_j\mathbf{M}$ la projection de $\mathbf{M}$ dans l'image j. La distance d élevée au carré, est égale à $d_x^2 + d_y^2$ (avec d_x et d_y de l'équation 2.25).

Erreur de reprojection

Lorsque que les rayons optiques ne s'intersectent pas, le point $\mathbf{M}$ reconstruit ne peut pas appartenir aux deux rayons. Par conséquent, les projections 2-D $\bar{\mathbf{m}}_{\mathrm{l}}$ et $\bar{\mathbf{m}}_{\mathrm{r}}$ ne correspondent pas aux points 2-D $\mathbf{m}_{\mathrm{l}}$ et $\mathbf{m}_{\mathrm{r}}$, initialement mesurés. Cet écart entre les points reprojetés $\bar{\mathbf{m}}_i$ et les points $\mathbf{m}_i$ mesurés est appelé erreur de reprojection, et s'exprime par le vecteur colonne à deux composantes d :

$$d = \begin{bmatrix} d_x \\ d_y \end{bmatrix} = \left[\mathbf{m}\right] - \left[\bar{\mathbf{m}}\right] \tag{2.25}$$

avec $\bar{\mathbf{m}} = \mathsf{P}\mathbf{M}$ la projection de $\mathbf{M}$ dans l'image. L'erreur de reprojection est d'autant plus importante que les rayons optiques passent loin l'un de l'autre, signe que la contrainte épipolaire est d'autant moins respectée.

2.3 Méthodologie d'évaluation de la qualité du calibrage

La connaissance des paramètres géométriques du système stéréoscopique est nécessaire pour pouvoir extraire des mesures 3-D à partir d'images. Ces paramètres pourraient être spécifiés par le concepteur, et il serait à la charge du fabricant (ou fournisseur) de les respecter. Mais atteindre la précision de montage requise serait trop coûteux. L'alternative consiste à monter le système « au mieux », puis de l'étalonner, afin d'obtenir avec précision les paramètres effectifs. Cette opération est souvent appelée calibrage (*calibration* est le terme anglais). Le calibrage, ou étalonnage, permet d'estimer la fonction qui projette un point 3-D du monde sur l'image. Cette fonction, comme par exemple celle associée au modèle sténopé, fait intervenir un ensemble de paramètres. Dans ce cas, le calibrage revient à estimer cet ensemble de paramètres. La littérature fournit un très grand nombre de travaux sur cette thématique [Bro71, Tsa87, Bey92, WCH92, LL96, ZFD97, Zha00].

Les performances atteintes par les différentes méthodes de calibrage sont variables. Elles dépendent de plusieurs facteurs, tels que le modèle, l'algorithme mis en œuvre, la précision de la mire, sa configuration, etc. Pour un constructeur automobile, il est primordial d'identifier et de comprendre ce qui peut limiter la performance d'un système stéréoscopique. La pauvreté de la littérature sur les performances liées au calibrage nous a poussé à étudier le problème. Dans cette optique, nous présentons une méthodologie d'évaluation des algorithmes de calibrage. Cette méthodologie passe inévitablement par la définition de critères de qualité, que nous décrivons en première partie, permettant de comparer les différents calibrages. Nous décrivons ensuite le coeur de cette approche qui, basée sur des simulations, permet de tester et d'isoler l'impact que chaque facteur a sur la qualité finale du calibrage. Et finalement, nous présentons les résultats que nous avons obtenus sur notre système, et dégageons les points importants à mettre en place pour obtenir un calibrage de qualité. Ces travaux ont été publiés dans la conférence *Intelligent Vehicles* [MDCG07].

2.3.1 Critères de qualité

Nous choisissons trois critères pertinents pour évaluer la qualité du calibrage.

Le premier critère est l'erreur commise sur l'estimation de chaque paramètre du système. Il est obtenu par la différence entre les paramètres réels et ceux estimés à l'aide du calibrage (voir la figure 2.18(a)). Il permet de repérer les paramètres qui sont mal estimés.

Le deuxième critère est l'erreur de reconstruction. Celle-ci est quantifiée en comparant les coordonnées réelles des points d'échantillonnage à celles obtenues via la reconstruction utilisant les paramètres calibrés (voir la figure 2.18(b)). Dans notre comparaison, nous avons choisi d'examiner le vecteur d'erreur, plutôt que d'évaluer une distance euclidienne. Cette information, plus riche, permet de différentier un biais longitudinal, d'un biais latéral, qui n'aurait pas le même impact sur l'application. Pour synthétiser l'erreur commise sur l'ensemble des points d'échantillonnage, nous calculons l'erreur quadratique moyenne (*RMS*) sur chacune des 3 coordonnées. La mesure de qualité de reconstruction obtenue dépend donc évidemment de la qualité du calibrage, mais aussi de l'ensemble

des points d'échantillonnage utilisés pour l'évaluation. Il est donc important de choisir cet ensemble de manière représentative de l'application visée.

Le dernier critère est l'erreur de reprojection (voir les figures 2.17 et 2.18(c)). Comme expliqué dans la partie 2.2.3, celle ci est due à une violation de la contrainte épipolaire. De la même façon que pour l'erreur de reconstruction, l'erreur de reprojection est évaluée sur un ensemble de points d'échantillonnage. En résulte un ensemble de vecteurs d'erreur à deux composantes. Cependant, lorsque les deux caméras sont placées l'une à coté de l'autre, l'erreur de reprojection selon l'axe x n'est pas significative, et nous n'affichons que l'erreur de reprojection selon l'axe y. Si cette dernière dépasse les 0.5 pixels, c'est le signe d'une violation de la contrainte épipolaire suffisante pour mettre les algorithmes de stéréo-corrélation en échec (voir 2.2.2 p.26 pour plus de détails). Autre similitude avec le critère précédent, l'erreur de reprojection étant dépendante de l'échantillonnage, il est important de bien choisir celui-ci. Pour des raisons pratiques évidentes, dans nos expérimentations, nous prendrons les mêmes échantillons pour l'évaluation de l'erreur de reconstruction et de reprojection.

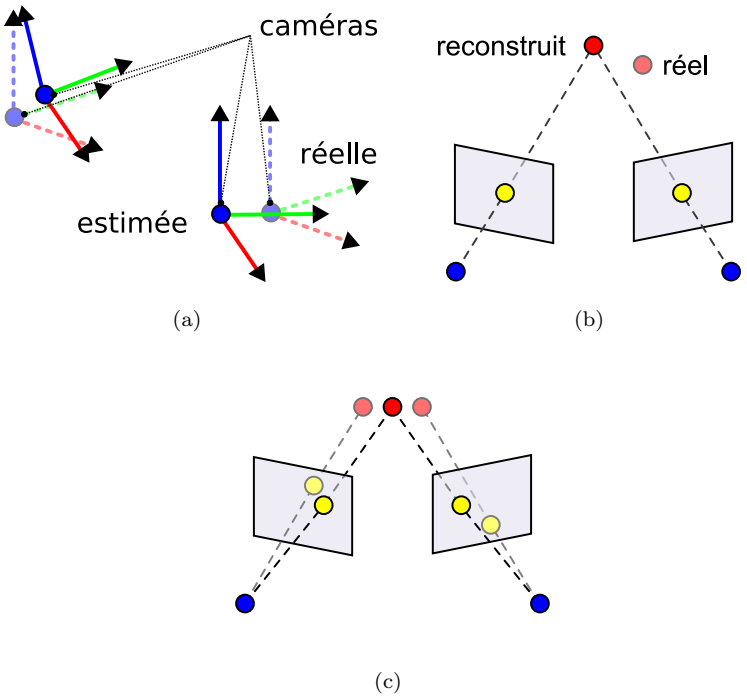


FIG. 2.18: Les trois critères de qualité d'un calibrage sont : la précision des paramètres estimés (a), et les erreurs de reconstruction (b) et de reprojection (c) commises avec ces paramètres. Voir la figure 2.17 pour plus de détail sur l'erreur de reprojection.

2.3.2 Méthodologie d'évaluation

Avant de décrire la méthodologie d'évaluation du calibrage, commençons par examiner et partitionner le processus de calibrage en lui même. Il existe

deux types de calibrage : fort et faible. Seul le calibrage fort permet d'obtenir une reconstruction métrique, nécessaire aux applications automobiles. Nous n'examinerons donc que ce dernier. Toutes les méthodes de calibrage fort font intervenir un étalon (une mire). Cet étalon doit non seulement être connu, mais aussi être facilement détecté dans les images (comme c'est le cas de la mire damier de *OpenCV* [Int01], figure 2.4). Il existe néanmoins un moyen de calibrer sans mire, à la condition de connaître certains des paramètres intrinsèques, comme les focales et les points principaux [HZ04]. Cependant, calibrer à l'aide d'une mire donnera toujours de meilleurs résultats, et sera toujours préféré.

Comme le schématise la figure 2.18, le calibrage fort consiste en une série d'acquisitions de la mire (par exemple le damier de la figure 2.4), sous une ou plusieurs postures. Un algorithme de détection permet ensuite d'extraire automatiquement les éléments de la mire (points, coins, cercles, lignes, etc.) dans les images. La précision de la détection varie selon le motif de la mire et la méthode de détection. Typiquement, les coins du damier de la figure 2.4 sont détectés avec une précision de l'ordre de 0.1 pixel. Enfin, à partir des points 2-D détectés dans les images et des points 3-D de la mire, l'algorithme de calibrage estime les paramètres du système.

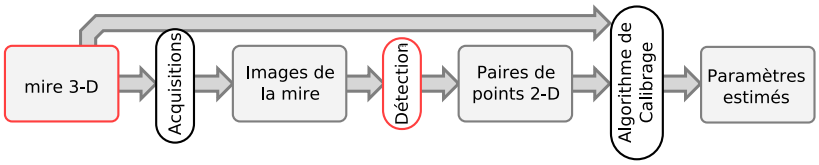


FIG. 2.19: Le calibrage utilise une série d'acquisitions de la mire pour estimer les paramètres du modèle choisi.

Même si le mécanisme et la procédure de calibrage varient sensiblement d'un algorithme à l'autre, les principaux facteurs influençant la qualité demeurent :

1. la précision de construction de la mire,
2. la configuration géométrique de la mire, c'est à dire la forme et la position de la mire dans la scène,
3. la configuration géométrique du système, comme par exemple la distance entre les caméras, les distances focales, etc.
4. la précision de détection de la mire dans les images,
5. et de l'algorithme de calibrage mis en œuvre.

La méthodologie d'évaluation que nous présentons dans cette partie a pour but de quantifier la qualité maximale pouvant être atteinte par un système donné et en fonction des facteurs listés précédemment. L'idée est de fournir un outil d'aide à la conception.

Une approche consisterait à envisager le problème de manière analytique. Mais l'aspect extrêmement complexe des algorithmes de calibrage (non-linéaires, non déterministes, ...) rend ce type d'approche très difficile, voir impossible. Nous préférons donc une approche plus empirique, consistant à évaluer directement la qualité atteinte par un système calibré sous certaines conditions. Cette méthode s'articule en trois grandes étapes. La première étape consiste à calibrer le système, la deuxième à le faire fonctionner sur des échantillons connus, et la

troisième à quantifier la qualité atteinte. Même s'il est possible, et recommandé, d'appliquer cette évaluation sur un prototype physique [Tsa87], nous conduisons l'évaluation à l'aide de simulations. En effet l'évaluation sur simulation s'affranchit des limites de l'évaluation sur des données réelles, à savoir que dans le cas de données réelles :

- tous les paramètres ne peuvent pas être maîtrisés,
- les manipulations à mettre en œuvre sont fastidieuses.

Étape 1 : simuler la procédure de calibrage

Lors de cette étape, nous simulons les trois processus (acquisition, détection et calibrage) et les données intermédiaires intervenant lors d'un calibrage (voir la figure 2.18).

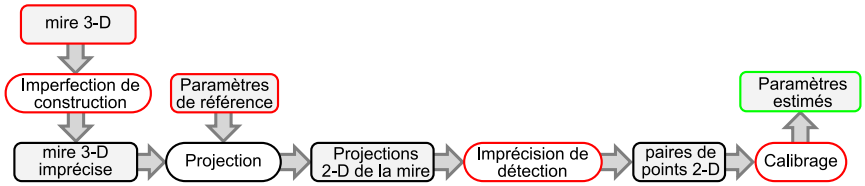


FIG. 2.20: Étape 1 : Simulation du processus de calibrage (voir la figure 2.18) en prenant en compte toutes les sources potentielles d'imperfections.

Comme l'indique la figure 2.20, la première donnée à simuler est la mire 3-D. Pour simplifier, nous la synthétisons sous forme d'un ensemble de points 3-D (voir la figure 2.21(a)). En pratique, la mire, aussi précise soit elle, n'est jamais parfaite. Nous simulons également cette « imperfection » de construction en ajoutant un bruit blanc gaussien aux coordonnées de chaque point 3-D de la mire (voir la figure 2.21(b)). Un autre point important est la répartition des points de la mire dans la scène. Nous détaillerons ce dernier aspect dans l'analyse des résultats, en partie 2.3.3.

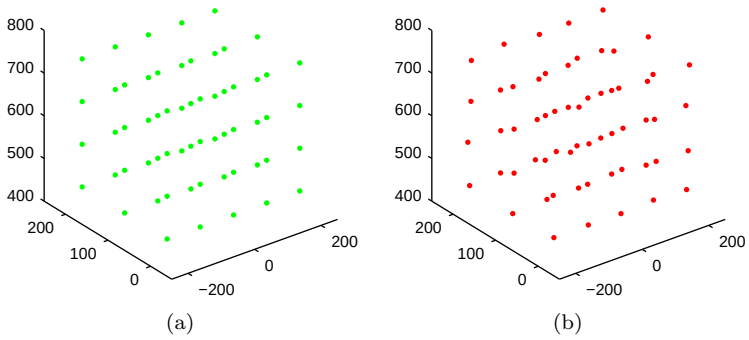


FIG. 2.21: La mire synthétique est constituée d'un ensemble de points 3-D (a) répartis régulièrement, puis (b) bruités.

Ensuite, pour simuler chaque caméra, nous appliquons la fonction de projection donnée par le modèle sténopé sur les points 3-D de la mire synthétique. Les distorsions radiale et tangentielle peuvent être ajoutées. Les paramètres de la projection, que nous notons paramètres de référence sur la figure 2.20, sont choisis de façon aussi représentative que possible du système à évaluer.

La détection automatique des points de la mire dans les images étant imprécise, nous la simulons par l'ajout d'un bruit blanc gaussien aux coordonnées 2-D de chaque point.

Enfin, les paires de points 2-D et 3-D associés sont utilisés par l'algorithme de calibrage pour estimer les paramètres du système.

Nous avons donc bien simulé toutes les sources potentielles d'imperfections, à savoir :

- la précision de construction de la mire,
- la configuration géométrique de la mire,
- la configuration géométrique du système,
- la précision de détection de la mire dans les images,
- et l'algorithme de calibrage mis en œuvre.

Étape 2 : simuler le système après calibrage

La deuxième étape consiste à simuler un système pendant son fonctionnement (voir la figure 2.1), afin de pouvoir l'évaluer par la suite.

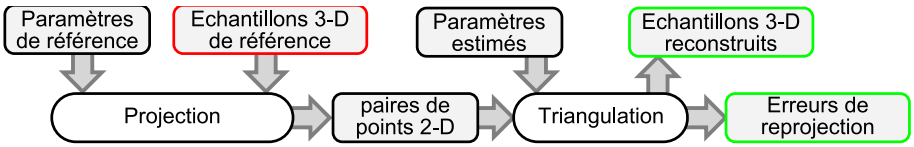


Fig. 2.22: Étape 2 : Simulation du processus de stéréovision (voir la figure 2.1).

Tout d'abord nous échantillonnons la zone de couverture du système, c'est à dire le volume dans lequel le système peut opérer (délimité par les pointillés rouges sur la figure 2.24). L'ensemble des échantillons est constitué de nombreux points 3-D répartis dans le volume. L'évaluation sera restreinte aux limites proches et lointaines du volume de travail, qui représentent des cas extrêmes.

Tout d'abord, nous simulons le capteur comme dans l'étape 1. A la différence de l'étape 1 aucune n'est mire qui est observée, mais une scène représentative.

Puis, chaque algorithme mis en œuvre lors du fonctionnement (rectification, mise en correspondance, détection d'obstacles et conversion vers des unités métriques) peut être source d'erreurs. Mais seules nous intéressent les erreurs engendrées par une mauvaise qualité de calibrage. Nous simplifions donc tous ces algorithmes par un seul algorithme de triangulation appliqué sur les paires de points 2-D issus de la simulation du capteur. La triangulation utilise les paramètres calibrés et non les paramètres réels. C'est cet écart entre paramètres réels et estimés qui engendre des imprécisions.

En résumé, la simulation du système consiste en la simulation du capteur et la triangulation (voir la figure 2.22).

Étape 3 : évaluer

La dernière étape (voir la figure 2.23) est le calcul des trois critères de qualité, comme décrit dans la partie 2.3.1. L'évaluation sera restreinte aux limites proches et lointaines du volume de travail (en vert sur la figure 2.24), qui représentent des cas extrêmes.

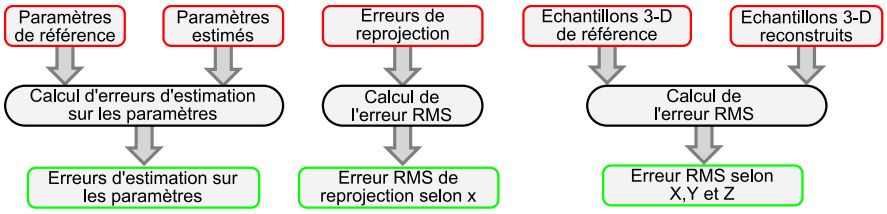


FIG. 2.23: Étape 3 : Évaluation suivant les trois critères définis dans 2.3.1.

2.3.3 Résultats

Dans les résultats que nous présentons ici, nous avons choisi une configuration de système proche de celle de notre véhicule d'essai, à savoir une distance focale de 6 mm, avec 640×480 pixels de résolution répartis sur un capteur $1/3''$ (correspondant à une caméra *Dragonfly* de *Point Grey Research* montée avec un objectif 6 mm *Pentax*). Le centre optique est choisi au centre du capteur et la distorsion est négligée. Les caméras sont placées à 40 cm l'une de l'autre, regardant toutes les deux vers l'avant du véhicule, et strictement d'aplomb.

Pour calibrer le système nous utilisons une mire 3-D et l'algorithme fourni par la librairie *OpenCV* [Int01], développé par Bouguet [Bou]. Nous avons imaginé une mire damier de 4 m de large et 2 m de haut, faisant face aux caméras. Placée sur des rails, elle pourrait s'éloigner d'une distance comprise entre 3.5 m et 30 m. L'ensemble des points 3-D résultant d'une telle mire seraient répartis régulièrement dans le cube bleu ciel de la figure 2.24. Dans les différentes expérimentations qui suivent, la loi normale simulant l'imprécision de construction est de moyenne nulle et d'écart-type $\sigma_{3D} = 1$ mm (sauf mention contraire). Celle modélisant l'erreur de détection dans les images est de moyenne nulle et d'écart-type $\sigma_{2D} = 0.5$ pixel (sauf mention contraire).

Le système est évalué sur 4800 points échantillons répartis uniformément à 31.5 m de distance, sur toute la largeur du volume utile (volume vert sur la figure 2.24). L'erreur *RMS* est exprimée en cm pour l'erreur de reconstruction, et en pixel pour l'erreur de reprojection.

Nous nous sommes attachés à étudier en particulier la qualité du calibrage en fonction de la précision de construction de la mire, de la précision de détection de la mire et de la géométrie de la mire.

Qualité en fonction de la précision de construction de la mire

En premier lieu, nous étudions la conséquence d'une construction imparfaite de la mire. Cette imperfection se traduit par une différence entre les coordonnées supposées des points de la mire, et les coordonnées réelles de ces mêmes points. Nous modélisons cette imperfection par un bruit blanc gaussien σ_{3D}

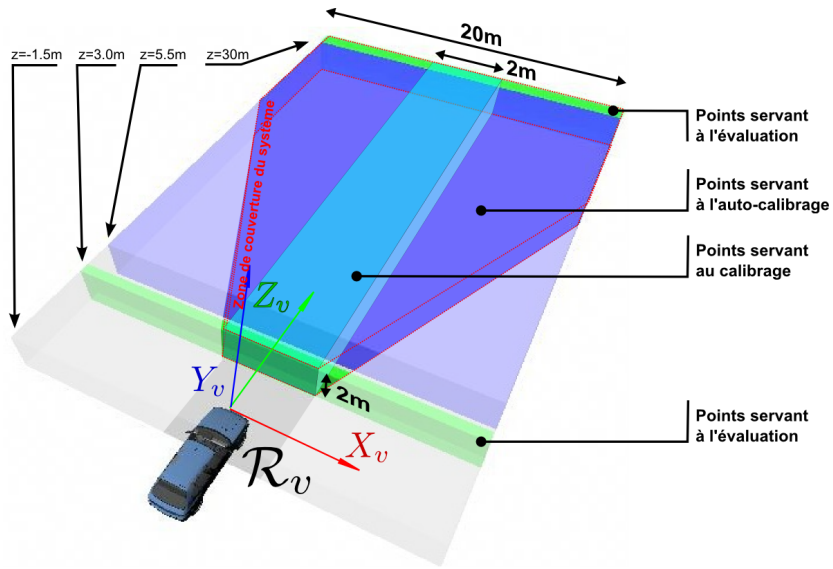


FIG. 2.24: Les points 3-D servant au calibrage sont répartis dans un parallélépipède de 4m de large sur 2 m de haut et 24.5 m de profondeur (en bleu clair), alors que les points 3-D servant à évaluer le calibrage (en vert) sont répartis aux extrémités proches et lointaines du volume utile (à 3 et 30 m de distance) sur toute la largeur de la scène (20 m).

s'ajoutant aux coordonnées euclidiennes des points 3-D de la mire. Les résultats listés dans le tableau 2.2 ont été obtenus en utilisant la configuration expliquée précédemment, à l'exception du bruit de mesure σ_{2D}, qui est négligé.

σ_{3D} (cm)	0.1	1	$10(0,3\%)$
RMS_X (cm)	0.05	0.74	$2.82(0,1\%)$
RMS_Y (cm)	0.01	0.23	$2.14(0,1\%)$
RMS_Z (cm)	0.45	4.37	$23.3(0,7\%)$
RMS_y (pixel)	0.0008	0.0273	0.0309
f_x^r (pixel)	0.015	0.148	2.216
b (cm)	-0.001	-0.011	-0.148
p (°)	-0.0006	-0.002	-0.007
r^l (°)	0.002	0.059	0.073
r^r (°)	0.002	0.058	0.075
y^l (°)	0.0004	0.1173	0.0375
y^r (°)	0.0009	0.1262	0.1355

TAB. 2.2: Les trois critères de qualité du calibrage en fonction de l'imperfection σ_{3D} de la mire. La configuration du système est détaillée en début de partie 2.3.3, à l'exception du bruit de mesure σ_{2D}, qui est négligé.

Le résultat important à retenir du tableau 2.2 est qu'une mire précise au cm

suffit à calibrer le système. Notons cependant que l'imprécision est simulée par un bruit blanc gaussien, et ce type de bruit est très bien géré par les méthodes de calibrage basées sur une optimisation aux moindres carrés.

Qualité en fonction de la précision de détection de la mire

En deuxième lieu, nous étudions la conséquence d'une détection imprécise de la mire. Nous modélisons cette imperfection de détection par un bruit blanc gaussien $\sigma_{2\mathrm{D}}$ qui s'ajoute aux coordonnées des points projetés de la mire. Les résultats listés dans le tableau 2.3 ont été obtenus sur la même configuration que la simulation précédente, excepté $\sigma_{2\mathrm{D}}$, qui varie entre 0.1 et 1 pixel. $\sigma_{3\mathrm{D}}$ (précision de construction de la mire) est fixé à 1 mm.

$\sigma_{2\mathrm{D}}$ (pixel)	0.1	0.5	1
RMS_X (cm)	0.03	0.04	0.20
RMS_Y (cm)	0.006	0.015	0.200
RMS_Z (cm)	0.257	0.396	$5.949(0,2\%)$
RMS_y (pixel)	0.0018	0.0019	0.0174
f^l (pixel)	-0.01	-0.022	-0.05
b (mm)	-0.01	0.06	-0.19
p (°)	-0.0006	-10^{-5}	-0.007
r^l (°)	-0.003	-0.006	-0.032
r^r (°)	-0.003	-0.006	-0.032
y^l (°)	-0.004	0.008	-0.087
y^r (°)	-0.006	0.011	-0.110

Tab. 2.3: Qualité du calibrage en fonction de la précision de détection des points dans les images ($\sigma_{2\mathrm{D}}$). La configuration du système est détaillée en début de partie 2.3.3.

Le résultat important à retenir du tableau 2.3 est qu'une mire précise au cm suffit à calibrer le système. Notons que l'imprécision est simulée par un bruit blanc gaussien, et ce type de bruit est très bien géré par les méthodes de calibrage basées sur une optimisation au moindre carré. Le résultat serait probablement différent dans le cas d'un biais systématique. En effet, la méthode de calibrage testée ici est optimale en présence d'un bruit blanc gaussien sur les points détectés dans les images.

Qualité en fonction de la géométrie de la mire

En dernier lieu, nous désirons vérifier qu'une mire plus petite donne une qualité de calibrage similaire. Pour cela, nous comparons la mire telle que décrite en début de cette partie à une mire réduite, plus aisée à fabriquer. La petite mire couvre un volume allant de 4.5 m à 7 m de distance, soit une profondeur de 2.5 m.

Les résultats listés dans le tableau 2.4 mettent en évidence l'apport d'une mire couvrant tout le volume utile. En effet, une mire ne couvrant que les 2.5 premiers mètres du volume de reconstruction donne une qualité de calibrage inférieure, puisque l'erreur *RMS* de reconstruction selon l'axe Z, passe de 0.38

Profondeur de la mire (m)	$4.5 \rightarrow 7$	$4.5 \rightarrow 30$
RMS_X (cm)	0.44	0.030
RMS_Y (cm)	0.22	0.02
RMS_Z (cm)	$9.38(0,3\%)$	0.38
RMS_y (pix)	0.003	0.011
f^r (pix)	-0.531	-0.133
b (mm)	0.19	-0.374
p (°)	-0.009	-0.007
r^l (°)	0.003	-0.008
r^r (°)	0.006	-0.009
y^l (°)	-0.138	-0.118
y^r (°)	-0.131	-0.100

TAB. 2.4: Évaluation du calibrage en fonction de la géométrie de la mire. Les points de la mire sont espacés de 50 cm les uns des autres, pour une total de 270 points pour la petite mire (colonne 1) et de 2340 points pour la grande mire (colonne 2).

à 9.38 cm. Ce résultat rejoint la constatation de Beyer *et al.* [Bey92], selon laquelle la qualité de la reconstruction est meilleure dans le volume occupé par la mire.

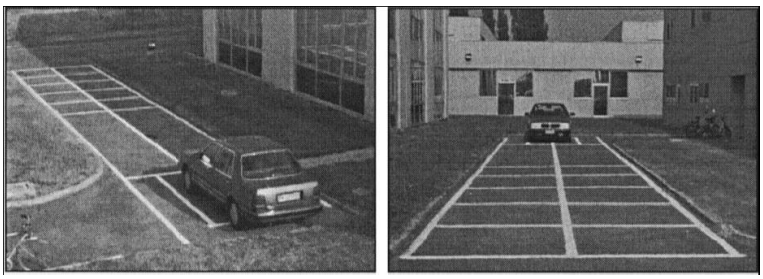
Conclusion

Dans tous les scenari d'imprécision, c'est l'erreur *RMS* en profondeur qui est la plus fortement perturbée. Les erreurs RMS_X et RMS_Y restent raisonnables (< 3 cm). L'erreur *RMS* de reprojection reste très largement inférieure au demi-pixel, ce qui ne mettrait pas les algorithmes de mise en correspondance en défaut. Le principal défi reste à obtenir une bonne précision sur l'estimation des profondeurs, essentiellement à 30 m de distance. Les détecteurs précis à 0.5 pixel donnent des résultats tout à fait satisfaisants.

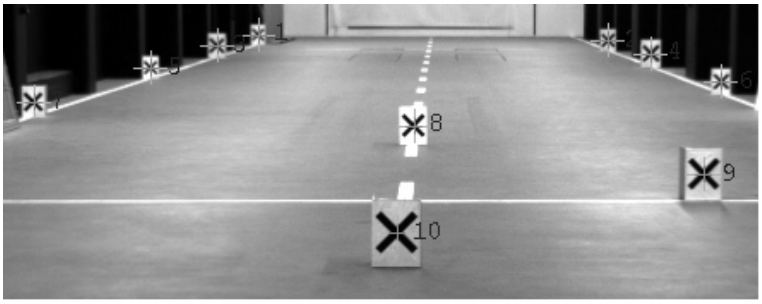
Par contre, et notre expérience le confirme, il est très important de calibrer tout le volume de reconstruction, même si le nombre de points lointains est beaucoup plus réduits que le nombre de points proches (voir la figure 2.25).

2.4 Calibrage du véhicule au cours de son cycle de vie

Dans la partie précédente, nous avons étudié la qualité de calibrage qu'il était possible d'atteindre avec des méthodes basées sur des mires. Une fois calibré, le système stéréoscopique installé à bord du véhicule endure des conditions extrêmes : des vibrations, des chocs et des températures extrêmes. Aussi bon soit le calibrage, et en dépit des efforts de construction, la configuration du système peut être modifiée par ces perturbations extérieures, rendant les paramètres calibrés erronés. C'est la première question que nous nous sommes posés : est ce que même soumise à de fortes vibrations sur de courtes périodes, la tête stéréo



(a)



(b)

FIG. 2.25: La grille au sol sert à calibrer les paramètres extrinsèques du système (a) Argos [MR01] et (b) Nedevschi *et al.* [NMO+06].

reste calibrée ?

Ensuite, sur le long terme, il est évident que les paramètres risquent de changer. Pour qu'un système puisse équiper un véhicule de série, il doit au au moins être capable de détecter une perte du calibrage [NMO+06], et dans le meilleur des cas de se corriger automatiquement. Les constructeurs automobiles ont bien compris cette problématique, et la littérature fournit plusieurs articles expliquant comment « auto-calibrer » un système stéréo embarqué à bord d'un véhicule. Ces méthodes sont le plus souvent des adaptations de la technique très populaire d'ajustement de faisceaux [TMHF99]. Mais la technique est mal adaptée au cas d'un système embarqué à bord d'une automobile, car les trajectoires rectilignes que suit le véhicule produisent un cas dégénéré pour ce type d'approche. D'autre part, la nécessité de conserver une reconstruction métrique impose le choix de certaines contraintes.

2.4.1 Perte de calibrage

Évaluation de l'impact d'un décalibrage sur le processus de stéréo-vision

Nous savons que si les paramètres viennent à s'écarter de leur estimation initiale, alors le processus de stéréo-vision peut être biaisé, voire peut échouer. Mais quel est l'impact d'une variation des paramètres, et est ce que tous les paramètres ont la même influence ? Pour répondre à cette question, nous repre-

nons l'approche décrite dans l'étape 2 de la partie 2.3 (voir la figure 2.22), dans laquelle nous simulons un système sur un ensemble d'échantillons. Mais cette fois les paramètres de référence (réels) et les paramètres altérés sont intervertis : les paramètres de référence servent à la triangulation et les paramètres altérés servent à simuler le capteur (voir la figure 2.26). Enfin, nous reprenons exacte-

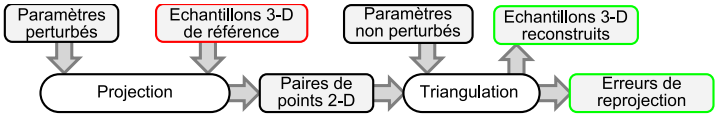


FIG. 2.26: Adaptation de la simulation du système (voir la figure 2.22) afin d'évaluer les conséquences d'un décalibrage.

ment l'approche décrite dans l'étape 3 de la partie 2.3 (voir la figure 2.23) pour évaluer la qualité du système ainsi perturbé.

Résultats

Pour les résultats présentés ci-après, nous avons repris la même configuration de capteur que dans la partie 2.3.3, à savoir une distance focale de 6 mm, avec 640×480 pixels de résolution répartis sur un capteur $1/3''$ (correspondant à une caméra *Dragonfly* de *Point Grey* montée avec un objectif 6 mm *Pentax*). Le centre optique est choisi au centre du capteur et la distorsion est négligée. Les caméras sont placées à 40 cm l'une de l'autre, regardant toutes les deux vers l'avant du véhicule, et strictement d'aplomb.

Dans les résultats qui suivent, nous nous limitons à l'analyse de trois paramètres seulement, mais la méthode d'évaluation est applicable à tous les paramètres. Nous évaluons les pertes de performance consécutives à une variation de la distance focale, puis une variation de tangage, et finalement une variation de lacet. Nous commentons les résultats obtenus. Toutes les variations sont appliquées à la caméra droite, mais les résultats sont transposables à la caméra gauche.

Examinons l'erreur de reconstruction suite à une variation de 0.5% de la distance focale. La figure 2.27 présente cette erreur sous forme de vecteurs pour des échantillons couvrant tout le volume utile.

déviation	$f^l + 0.5\%$	
distance de l'échantillonnage (m)	4.5	31.5
RMS_X (cm)	5	265
RMS_Y (cm)	1.5	11.2
RMS_Z (cm)	13	528
RMS_y (pixel)	0.2	0.04

TAB. 2.5: Dégradation des performances du système en fonction suite à la modification de 0.5% de la distance focale. Les 4800 échantillons 3-D servant à l'évaluation sont répartis à 4.5 m et 31.5 m de distance.

Les valeurs des erreurs sont résumées par les erreurs *RMS* de reconstruction

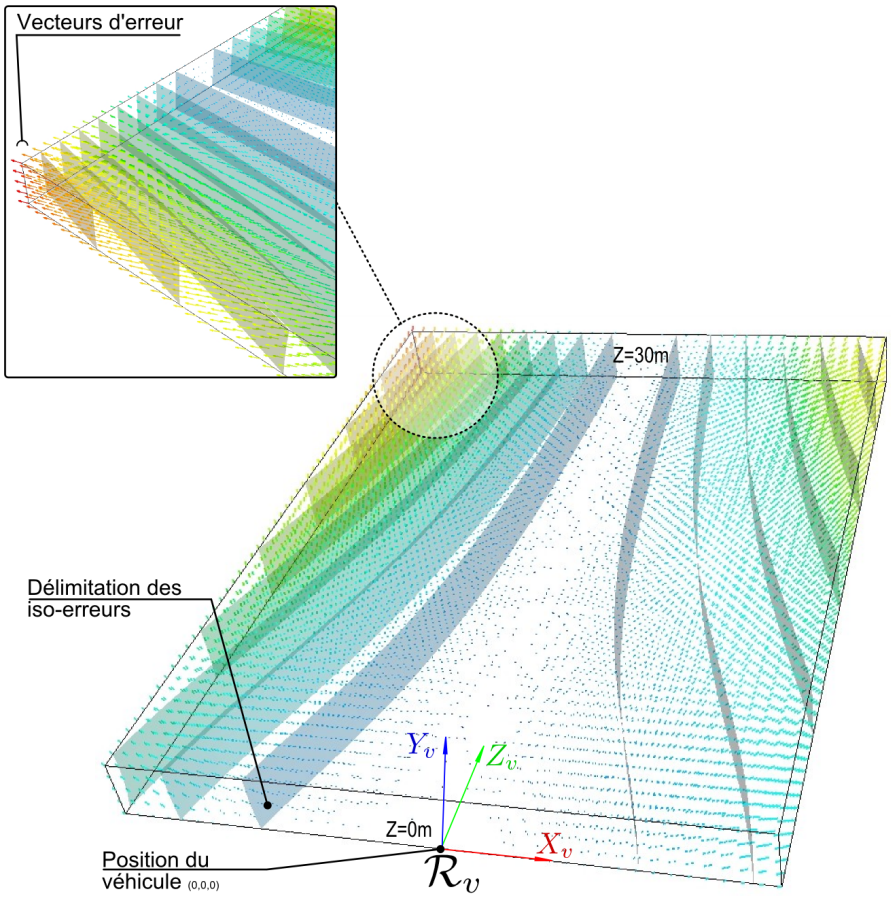


FIG. 2.27: Erreur de reconstruction suite à une déviation de 0.5% de la distance focale de la caméra droite. Les vecteurs représentent l'erreur de reconstruction. Du bleu au rouge, l'amplitude de l'erreur est croissante. Les bandes représentent les iso-surfaces de l'amplitude des vecteurs.

et de reprojection dans le tableau 2.5. Remarquons qu'une petite variation de la distance focale (0.5% → 4 pixels) a un impact très fort sur la reconstruction, puisque l'erreur *RMS* est supérieure à 5 m, à une distance de 30 m. Par contre, l'erreur de reprojection est peu affectée par ce paramètre.

Examinons maintenant les conséquences d'une variation de l'angle de tangage. La figure 2.28 présente cette erreur sous forme de vecteurs pour des échantillons couvrant tout le volume utile.

Les valeurs des erreurs sont résumées par les erreurs *RMS* de reconstruction et de reprojection dans le tableau 2.6. A l'inverse de la distance focale, l'angle de tangage influe peu sur la qualité de la reconstruction, puisque une variation de 0.5° induit seulement une erreur *RMS* de 20 cm à 31.5 m de distance. Par contre, l'erreur de reprojection est très affectée par ce paramètre, puisque pour la même déviation nous constatons une erreur *RMS* d'environ 3.5 pixels. Rappelons qu'une erreur de reprojection supérieure à 0.5 pixels met en défaut le

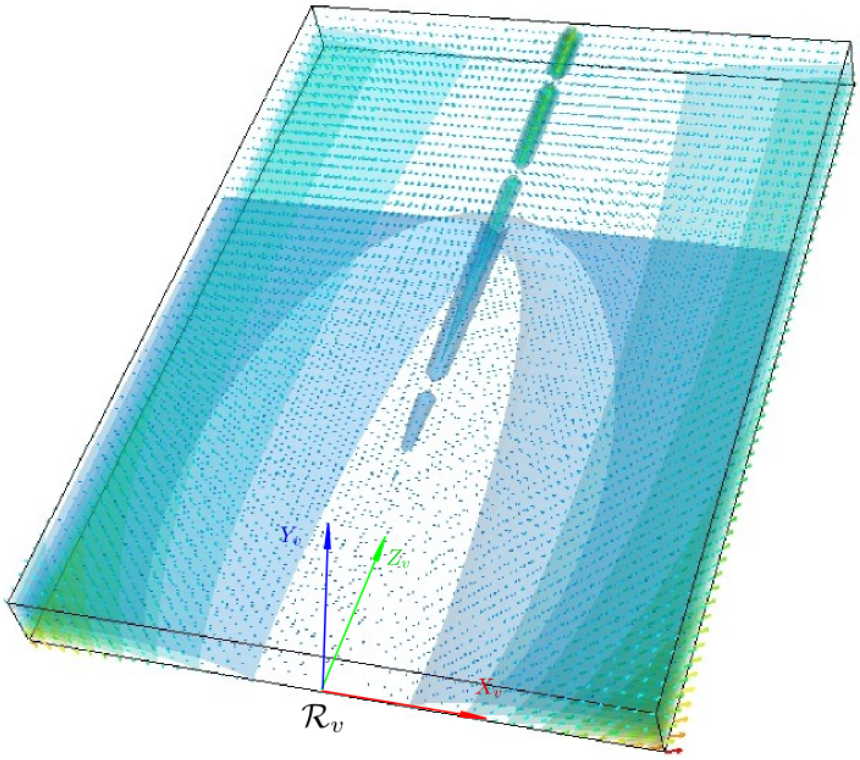


FIG. 2.28: Erreur de reconstruction suite à une déviation de 0.5° de l'angle de tangage de la caméra droite. Les vecteurs représentent l'erreur de reconstruction. Du bleu au rouge, l'amplitude de l'erreur est croissante. Les bandes représentent les iso-surfaces de l'amplitude des vecteurs.

déviation	$p^r + 0.5°$	
distance de l'échantillonnage (m)	4.5	31.5
RMS_X (cm)	1.03	10.2
RMS_Y (cm)	2.6	14.4
RMS_Z (cm)	2.7	20.8
RMS_y (pixel)	3.54	3.49

TAB. 2.6: Dégradation des performances du système en fonction d'une variation de 0.5 degrés de l'angle de tangage. Les 4800 échantillons 3-D servant à l'évaluation sont répartis à 4.5 m et 31.5 m de distance.

processus de corrélation stéréoscopique.

Examinons finalement les conséquences d'une variation de l'angle de lacet. La figure 2.29 présente cette erreur sous forme de vecteurs pour des échantillons couvrant tout le volume utile.

Le tableau 2.7 montre, qu'à l'inverse de l'angle de tangage, l'angle de lacet

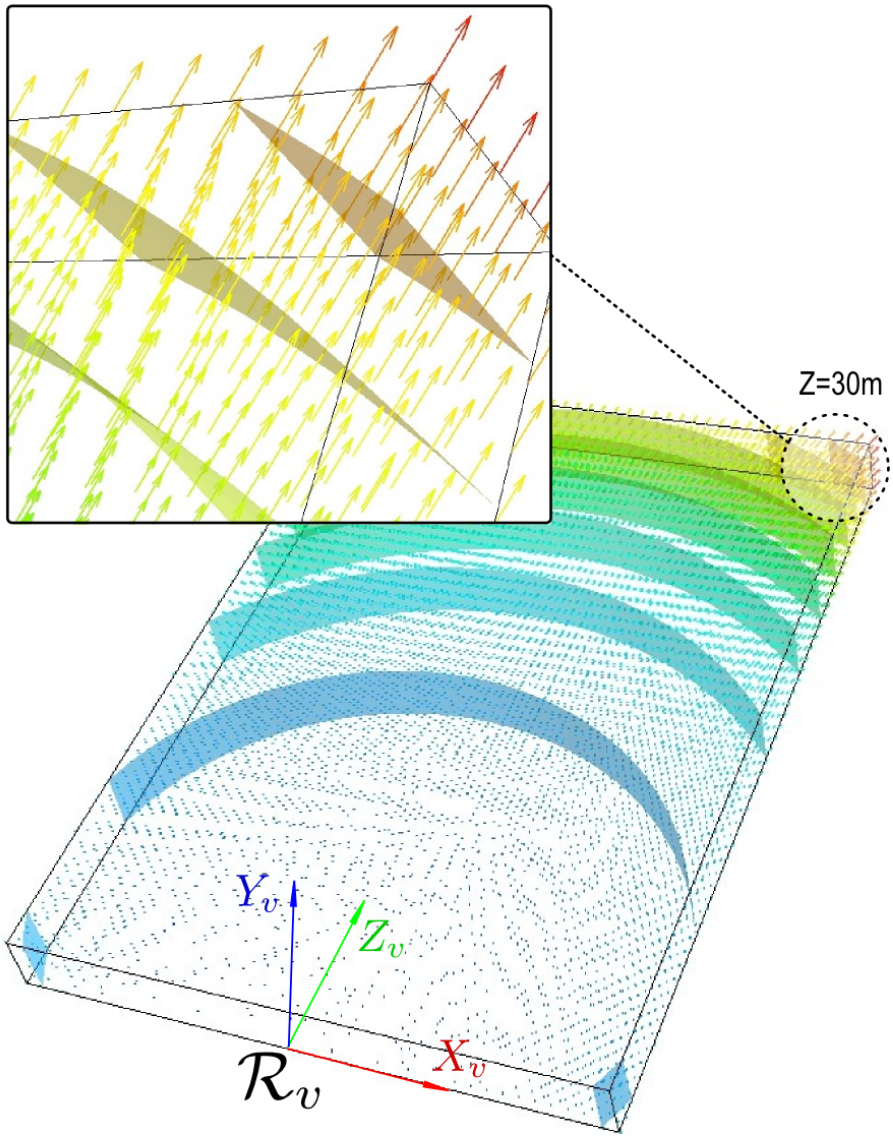


FIG. 2.29: Erreur de reconstruction suite à une déviation de 0.5° de l'angle de lacet de la caméra droite. Les vecteurs représentent l'erreur de reconstruction. Du bleu au rouge, l'amplitude de l'erreur est croissante. Les bandes représentent les iso-surfaces de l'amplitude des vecteurs.

influe énormément sur la qualité de la reconstruction. Une variation de 0.5° induit une erreur *RMS* de presque 1 m à 4.5 m de distance et jusqu'à 14 m d'erreur à 31.5 m de distance. Par contre, l'erreur de reprojection reste insignifiante, puisque pour la même déviation nous constatons une erreur *RMS* d'environ 0.1 pixels.

En conclusion, nous avons examiné l'influence de trois des nombreux pa-

déviation	$y^r + 0.5°$	
distance de l'échantillonnage (m)	4.5	31.5
RMS_X (cm)	22	570
RMS_Y (cm)	8.7	31.3
RMS_Z (cm)	74.6	1480
RMS_y (pixel)	0.11	0.02

TAB. 2.7: Dégradation des performances du système en fonction d'une variation de 0.5 degrés de l'angle de lacet. Les 4800 échantillons 3-D servant à l'évaluation sont répartis à 4.5 m et 31.5 m de distance.

ramètres du système. Nous avons montré que chacun de ces paramètres agit différemment sur les performances du système. En effet, là où une modification de l'angle de tangage met en défaut les algorithmes de stéréo-corrélation sans trop altérer la précision de reconstruction, une modification de la distance focale ou de l'angle de lacet mène à une reconstruction erronée (voir très erronée) sans perturber les algorithmes de stéréo-corrélation.

2.4.2 Comportement de la géométrie face aux vibrations dues au roulement

Comme nous venons de le démontrer, le système est affecté par le dérèglement des paramètres. Et donc, installer une tête stéréoscopique dans un véhicule n'a aucun sens si ces paramètres varient constamment. Nous voulions vérifier que cela n'arrive pas rapidement lorsque le véhicule roule en conditions extrêmes. Pour cela, nous avons mesuré les paramètres du système alors qu'il était soumis à de fortes contraintes. Pour des raisons pratiques, nous nous sommes limités aux paramètres extrinsèques des caméras.

Expérimentation 1

Toute la difficulté de cette expérimentation réside dans le fait que le véhicule doit être soumis à des contraintes (vibrations de roulage) tout en filmant une mire. Nous avons placé le véhicule porteur des caméras sur un banc capable de simuler un roulage, et placé des mires en face du véhicule. N'ayant pas la possibilité d'utiliser une vraie mire 3-D (comme celle de la figure 2.25), nous avons choisi une mire en damier. Mais l'estimation des paramètres extrinsèques à l'aide d'une seule mire plane s'avère très imprécise. Nous avons donc amélioré la méthode afin de pouvoir utiliser 2 mires planes disposées à des profondeurs différentes comme une mire 3-D (voir les figures 2.30 et 2.31).

Bien que chaque mire soit connue, leur position relative ne l'est pas. Pour pouvoir les utiliser comme une seule mire 3-D, nous avons donc dû imaginer une technique permettant de retrouver cette transformation rigide en même temps que les paramètres extrinsèques des caméras. Cette technique, détournée

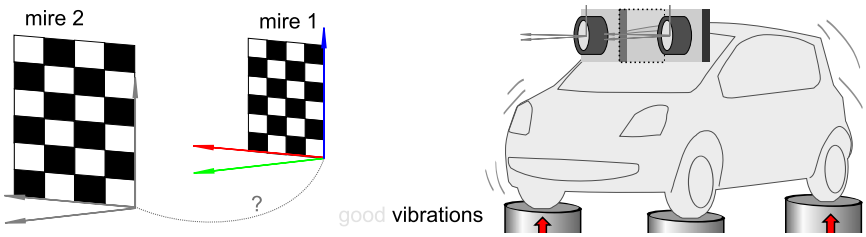


FIG. 2.30: (a) Les deux mires placées à des profondeurs différentes, sont filmées depuis le véhicule, alors qu'il est sollicité par le banc. La paire d'images 000 de la séquence (c) permet d'obtenir un placement relatif d'une mire par rapport à l'autre. Dans le reste de la séquence les deux mires sont utilisées comme une seule mire 3-D.

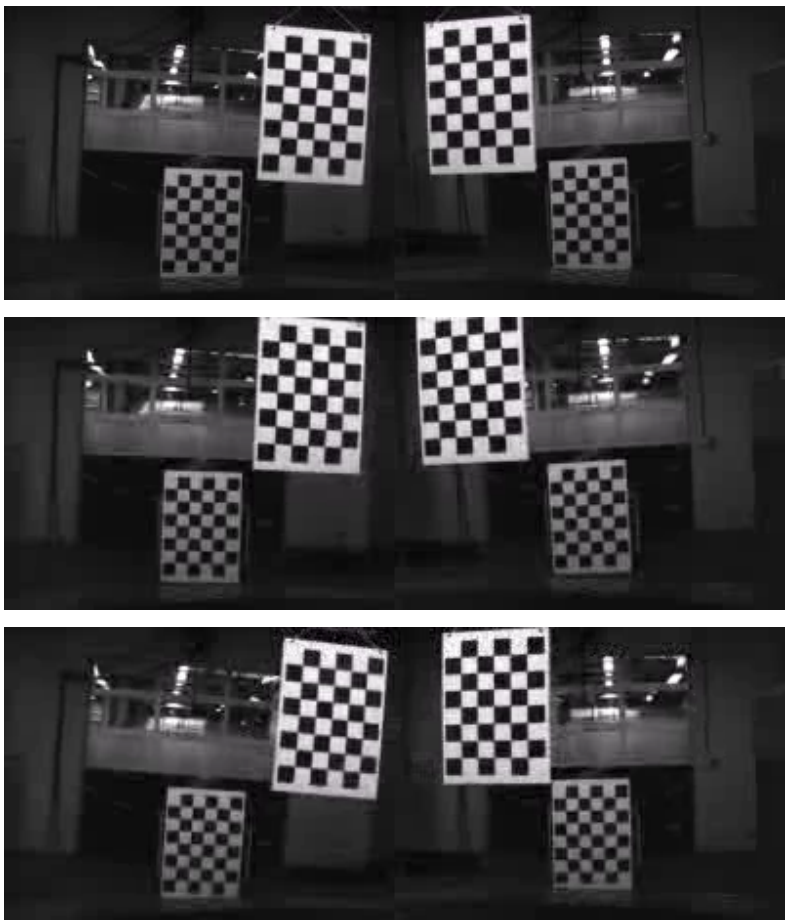


FIG. 2.31: Extraits de la séquence de paires d'images des 2 mires pendant l'excitation.

de l'ajustement de faisceaux, consiste à minimiser la fonction suivante :

$$\min_{(p_2,p_\mathrm{l},p_\mathrm{r})} \sum_{s=\{\mathrm{l},\mathrm{r}\}} \left[\sum_{i=1}^{n} \Big[\mathbf{m}_{si} - P\big(k_s, t(\mathbf{p}_s, \mathbf{M}_i)\big) \Big]^2 + \sum_{j=n+1}^{2n} \Big[\mathbf{m}_{sj} - P\big(k_s, t(\mathbf{p}_s, t(p_2, \mathbf{M}_j))\big) \Big]^2 \right], \tag{2.26}$$

avec :
- $i \in \{1, \ldots, n\}$ l'indice des points de la première mire,
- $j \in \{n+1, \ldots, 2n\}$ l'indice des points de la deuxième mire,
- $\mathbf{M}_k$ le $k^{\text{ème}}$ point 3-D,
- $\mathbf{m}_{sk}$ la projection du point $\mathbf{M}_k$ dans l'image s,
- t la fonction changement de repère,
- P la fonction de projection,
- $\mathbf{p}_2$, $\mathbf{p}_\mathrm{l}$ et $\mathbf{p}_\mathrm{r}$ les paramètres de pose de la seconde mire, de la caméra gauche et de la caméra droite,
- et k_l et k_r les paramètres intrinsèques des caméras.

La minimisation à été effectué par la méthode d'optimisation non-linéaire *Levenberg-Marquardt*, fournie dans la librairie *C_MinPack* (*MinPack* traduite en *C*).

Résultats 1

La séquence de 1500 paires d'images que nous avons traitée correspond à quelques secondes d'une route de campagne à 110km/h. Pour stabiliser les résultats, nous avons déterminé la position relative des mires sur la première paire d'images seulement. Puis, sur le reste de la séquence, nous avons utilisé un calibrage avec les points 3-D obtenus sur la première paire d'images. Pour améliorer la lisibilité, les paramètres extrinsèques de la paire de caméras sont convertis en paramètres internes à la tête stéréoscopique (cf. 2.2.1). Les graphiques de la figure 2.32 reprennent l'estimation de trois de ces paramètres pour les 1500 acquisitions.

Les graphiques (a) et (b) de la figure 2.32 montrent clairement le gain de précision apportée par notre approche (en vert) comparativement à la méthode n'utilisant qu'une mire plane. Les variations des angles de tangage et de roulis sont estimées inférieures à 1/10°. Une telle variation n'est pas de nature à rendre le système défaillant, puisqu'elle représente à peu près 0.5 pixel d'erreur de reprojection. Par contre, l'estimation de l'angle de lacet montre des variations d'environ 0.5°, ce qui peut engendrer une erreur de reconstruction de 14m à 31.5 m de distance (se reporter au tableau 2.7).

Ce dernier résultat, concernant l'estimation du lacet, est dû à un bruit de mesure. En effet, cette expérimentation s'avère inappropriée pour estimer cet angle. Dans une deuxième expérimentation, nous allons chercher à évaluer ce paramètre plus finement.

Expérimentation 2

Pour mesurer les variations de l'angle de lacet d'une caméra à l'intérieur de la tête stéréoscopique, nous avons mené une nouvelle expérimentation. Au

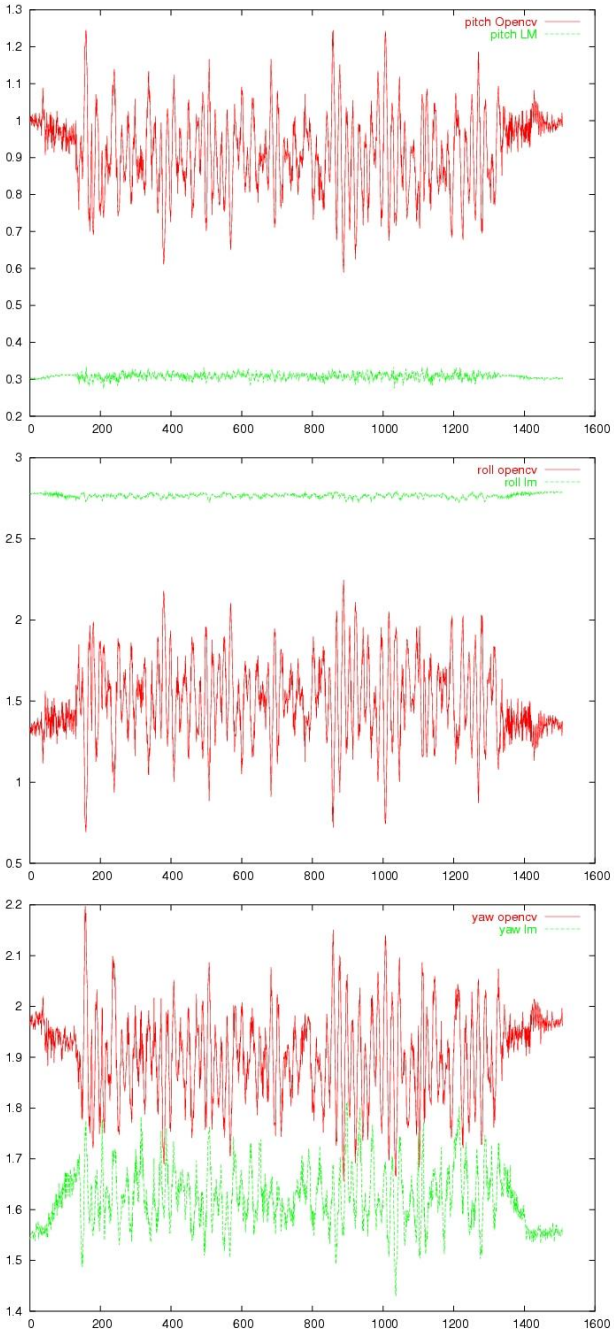


FIG. 2.32: Comparaison de l'estimation du tangage relatif, roulis et lacet (de haut en bas) avec la méthode à une mire plane (en rouge) ou avec une mire 3-D composée de deux mires planes (en vert). Sur l'axe horizontal sont répartis les acquisitions au cours du temps, et sur l'axe vertical exprime la valeur estimée de l'angle en degrés.

lieu de mesurer indépendamment les paramètres de chaque caméra, nous avons directement mesuré leurs mouvements relatifs en filmant l'une avec l'autre (voir la figure 2.33). La translation du centre de la mire dans l'image permet d'obtenir directement l'angle de lacet.

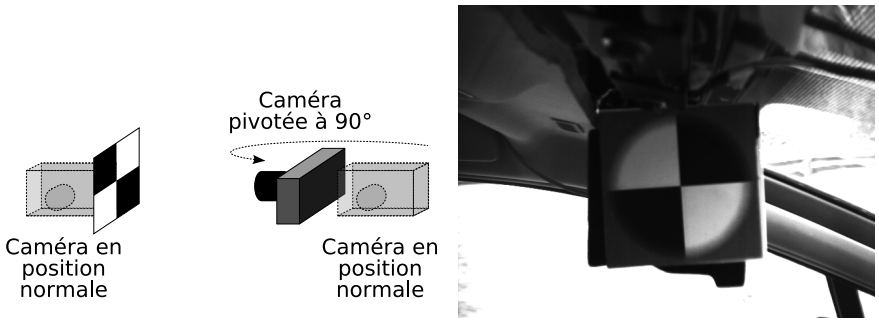


FIG. 2.33: La mire placée en lieu et place d'une des deux caméras est filmée par l'autre caméra, alors que le véhicule roule. Le déplacement de la mire dans l'image permet d'obtenir avec précision certains des paramètres extrinsèques de la caméra.

Résultats 2

Les déplacements mesurés au cours de cette expérimentation sont de l'ordre de 0.2 pixels. Une telle variation suivant l'axe x correspond à une variation de 0.1° suivant l'axe de lacet. La même variation suivant l'axe y correspondrait à une variation de 0.1° suivant l'axe de tangage (axe de roulis si la caméra était en position normale). De telles variations ne sont pas de nature à perturber le système.

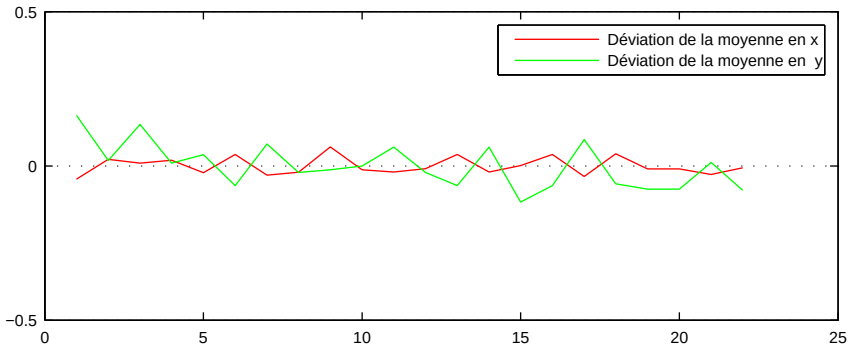


FIG. 2.34: Écart à la moyenne des coordonnées du centre le la mire (présentée dans la figure 2.33). Le déplacement d'environ 0.2 pixel correspondrait, au maximum, à une variation d'angle de 0.1°.

2.4.3 Détecter une perte du calibrage

Détecter une perte du calibrage est essentiel, surtout si le système doit agir sur les commandes du véhicule. Une solution serait de calibrer à la volée et de continuellement mettre à jour les paramètres. Mais autant la puissance de calcul que le nombre d'acquisitions nécessaires rendent irréaliste ce type d'approche.

Il existe néanmoins un critère facilement calculable qui informe sur la validité des paramètres de calibrage. Ce critère est le respect de la contrainte épipolaire. En effet, certains dérèglements ont un très fort impact sur le respect de cette contrainte. C'est le cas par exemple d'une variation de tangage (se référer à l'erreur de reprojection RMS_y dans le tableau 2.6). Détecter une incohérence de la contrainte épipolaire permet de détecter certains décalibrages.

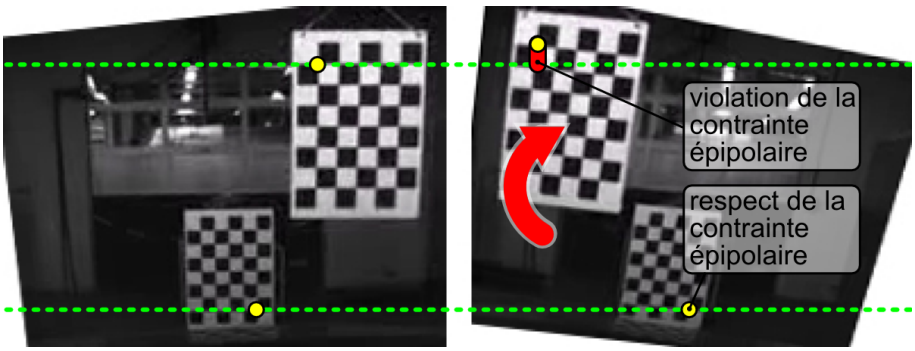


FIG. 2.35: La violation de la contrainte épipolaire est due à une perte de calibrage. Ici, une rotation de quelques degrés autour de l'axe de roulis engendre une violation (en rouge) de la contrainte épipolaire (matérialisé par la ligne en pointillés verte).

Pour cela, il y a plusieurs moyens. Le premier consiste à vérifier le bon fonctionnement des algorithmes de mise en correspondance. En effet, comme l'algorithme de mise en correspondance stéréoscopique se base fortement sur cette contrainte, une contrainte erronée a toutes les chances de faire échouer le processus. Un autre moyen consiste à appliquer des algorithmes de mise en correspondance en relâchant la contrainte épipolaire (voir la figure 2.35). Si les appariements de points trouvés ne respectent pas la contrainte connue, alors cela signifie que cette dernière est fausse et que le système est décalibré.

2.4.4 Calibrer « à la volée »

Les techniques d'ajustement de faisceaux

Le principe du calibrage à la volée, plus souvent appelé calibrage en ligne, est d'utiliser la scène, et non une mire, pour étalonner le système. Comme le montre la figure 2.36, l'idée est de mettre à jour les paramètres à partir d'une accumulation d'appariements de points 2-D.

Les méthodes de calibrage en ligne sont, pour la plupart, basées sur les techniques d'ajustement de faisceaux (*bundle adjustment* en anglais). Cette approche tente de résoudre conjointement l'estimation des paramètres caméras et

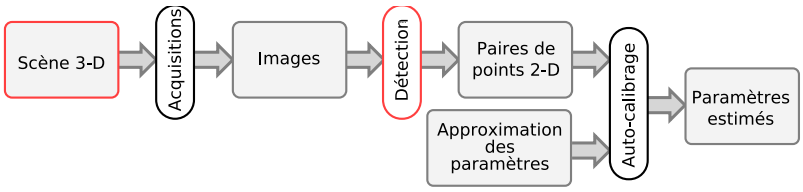


FIG. 2.36: Des paires de points détectées dans les images naturelles permettent de mettre à jour les paramètres de calibrage.

la reconstruction des points 3-D observés. Le problème est résolu par la minimisation d'une fonction d'erreur. Une méthode d'optimisation non-linéaire permet de surmonter le caractère complexe de la fonction d'erreur, ainsi que la nature éparse des données. La fonction d'erreur se base sur l'erreur de reprojection et s'écrit :

$$E(\{\mathbf{p}_i\},\{\mathbf{M}_j\},\{\mathbf{m}_{ij}\}) = \sum_{i=1}^{n}\sum_{j=1}^{m} d(P(\mathbf{p}_i,\mathbf{M}_j),\mathbf{m}_{ij})^2, \tag{2.27}$$

avec :

n : nombre de caméras (ou de points de vue différents)
m : nombre de points 3-D observés,
i : le numéro de caméra,
j : le numéro du point 3-D,
$\mathbf{p}_i$: le vecteur des k paramètres (intrinsèques et extrinsèques) de la caméra i,
$\mathbf{M}_j$: le vecteur de coordonnées du $j^{\text{ème}}$ point 3-D,
$\mathbf{m}_i j$: le vecteur de coordonnées de la projection 2-D du point $\mathbf{M}_j$ dans l'image i,
$d(\mathbf{m}_1,\mathbf{m}_2)$: distance euclidienne 2-D entre les deux points $\mathbf{m}_1$ et $\mathbf{m}_2$.

Pour simplifier, notons $\mathbf{X} \in \mathbb{R}^{(n\times k+3\times m)}$ le vecteur de l'ensemble des variables à optimiser, concaténation des paramètres de projections $\mathbf{p}_i$ et des coordonnées des $\mathbf{M}_j$. $\mathbf{X}$ s'écrit :

$$\mathbf{X} = ({\mathbf{p}_1}^\mathsf{T},\ldots,{\mathbf{p}_n}^\mathsf{T},\mathbf{M}_1,\ldots,\mathbf{M}_m). \tag{2.28}$$

De la même manière, notons $\mathbf{x} \in \mathbb{R}^{(3\times m\times n)}$ le vecteur de mesure des n points dans les m images (les points sont supposés visibles dans toutes les images). $\mathbf{x}$ s'écrit :

$$\mathbf{x} = ({\mathbf{m}_{11}}^\mathsf{T},\ldots,{\mathbf{m}_{nm}}^\mathsf{T}). \tag{2.29}$$

Étant donné que cette méthode est largement décrite par la littérature ([TMHF99]), nous ne nous étendons pas sur le sujet. Nous rappelons seulement les deux principales difficultés, à savoir : la nécessité d'initialiser judicieusement le système et l'influence sur les résultats de la nature des données acquises (cas dégénéré). Pour conclure, l'ajustement de faisceaux permet donc d'utiliser des images acquises en cours d'utilisation pour recalibrer le système.

Mais plusieurs facteurs éloignent le système stéréoscopique embarqué du cas d'usage idéal des techniques d'ajustement de faisceaux. Premièrement, l'utilisation d'ajustement de faisceaux pose le problème du facteur d'échelle. En effet,

cette méthode produit une reconstruction 3-D non métrique. L'introduction d'une métrique passe par l'ajout de contraintes sur le système. La littérature propose plusieurs types de contraintes. Par exemple Garcia [Gar01] fait le choix de fixer certains points de la scène en plaçant des marqueurs sur le véhicule [BBF01]. S. Nedevschi *et al.* [NVMG06] préfèrent utiliser les propriétés des marquages routiers.

Deuxièmement, la méthode d'ajustement de faisceaux est très sensible aux faux appariements. Une telle limitation n'est pas envisageable dans le contexte d'applications automobiles. Pour rendre l'auto-calibrage robuste aux faux appariements, V. Lemonde [Lem05] utilise l'algorithme de sélection RANSAC (en anglais *RANdom SAmple Consensus*, pour plus de détails, se référer à [FP02] ou [Mee04]). Dang *et al.* préfèrent une solution basée sur un moindre carré médian (LMS de *least median of squares* en anglais). Nous avons testé des solutions proches de ces deux dernières méthodes. La première, de Torr *et al.* [TM97], utilise RANSAC pour sélectionner le plus grand nombre de paires de points à la géométrie épipolaire cohérente. La deuxième consiste à modifier la fonction d'erreur de l'équation 2.27, pour y intégrer un *M-Estimateur* [Hub81, Tuk77].

2.4.5 Évaluation du calibrage en ligne

Nous présentons ici une méthode d'évaluation des algorithmes de calibrage en ligne. L'approche est très similaire à celle présentée dans la partie sur l'évaluation du calibrage (2.3), et en reprend deux des trois étapes. Les critères de qualité sont aussi les mêmes.

Méthodologie d'évaluation

Le principe du calibrage en ligne (ou auto-calibrage) est d'utiliser la scène, et non une mire, pour étalonner le système. Comme le montre la figure 2.36, l'idée est de mettre à jour les paramètres à partir de l'accumulation d'appariements de points 2-D. La qualité du résultat dépend des points acquis, de la configuration des caméras, etc. Pour évaluer la qualité du calibrage atteinte par les méthodes d'auto-calibrage, nous reprenons la méthode d'évaluation décrite dans la partie 2.3 (p. 31), en remplaçant l'étape de simulation du calibrage (schéma 2.20) par une étape de simulation du processus de calibrage en ligne.

Cette simulation, schématisée par la figure 2.37, nous permet d'évaluer plusieurs facteurs d'imprécision. Prenons la simulation pas à pas.

Tout d'abord nous simulons le comportement du capteur, alors qu'il acquiert la scène. Lors de cette étape, le capteur est simulé avec les paramètres « décalibrés ». Le nombre de points 3-D acquis et leur répartition dans la scène joue évidement un rôle dans la qualité du recalibrage.

Ensuite, nous simulons l'erreur de mise en correspondance. Cette erreur est modélisée par un bruit blanc gaussien ajouté au coordonnées. Pour également simuler un pourcentage de points aberrants, nous ajoutons des points dont les coordonnées sont générées suivant une loi uniforme.

Finalement, nous simulons le processus de calibrage. Ce processus nécessite des paramètres approchés en plus des paires de points 2-D déjà calculées. Ces paramètres servent d'initialisation à la méthode d'optimisation. Comme il doivent être aussi proches que possible des paramètres réels, il est d'usage de prendre les derniers paramètres estimés.

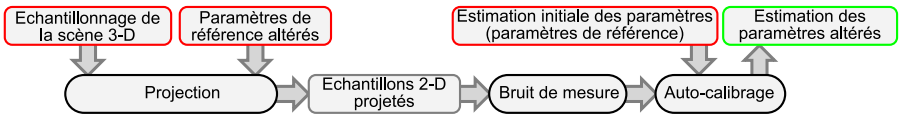


FIG. 2.37: Simulation du processus de calibrage en ligne (voir la figure 2.36) en prenant en compte toutes les sources potentielles d'imprécisions.

Pour résumer, les sources d'imprécision que nous permet d'étudier la méthode sont :

1. le nombre et la répartition des points dans les images,
2. la précision de détection et d'appariement de ces mêmes points,
3. l'algorithme de calibrage en ligne mis en œuvre et
4. le type et l'amplitude de la déviation.

2.4.6 Résultats

Dans les résultats que nous présentons ici, nous reprenons la même configuration (décrite dans la partie 2.3.3 p. 36), à savoir une distance focale de 6 mm, avec 640×480 pixels de résolution répartis sur un capteur $1/3''$ (correspondant à une caméra *Dragonfly* de *Point Grey* montée avec un objectif 6 mm *Pentax*). Le centre optique est choisi au centre du capteur et la distorsion est négligée. Les caméras sont placées à 40 cm l'une de l'autre, regardant toutes les deux vers l'avant du véhicule, et strictement d'aplomb.

La méthode de calibrage en ligne mise en œuvre dans ces expérimentations est une méthode originale reprenant certains des aspects de celle de V. Lemonde [Lem05]. Nous choisissons de fixer les paramètres de projection $\mathbf{p}_j$ correspondant à la position des deux caméras ainsi que certains paramètres intrinsèques. Les paramètres de projections $\mathbf{p}_j$ estimés correspondent à l'orientation et la distance focale de chaque caméra. Nous prenons pour hypothèse que les nouveaux paramètres sont proches de ceux initialement calibrés. Pour intégrer cette hypothèse dans la fonction d'erreur, nous associons un coût proportionnel à l'écart entre les paramètres initiaux et estimés. Ce coût est pondéré par un facteur déterminé a priori. Les erreurs de reprojection sont également podérées, mais par un facteur calculé automatiquement par la technique des *m-estimateurs*. La nouvelle fonction d'erreur s'écrit :

$$\sum_{i=\{l,r\}} \left[\sum_{j=1}^{m} \underbrace{\left[\alpha_j d(P(\mathbf{p}_i, \mathbf{M}_j), \mathbf{m}_{ij})^2\right]}_{\text{erreur de reprojection}} + \sum_{k=1}^{m} \underbrace{\left[\beta_k (\hat{\mathbf{p}}_i(k) - \mathbf{p}_i(k))^2\right]}_{\text{écart aux paramètres initiaux}} \right], \qquad (2.30)$$

avec :

m : nombre de points 3-D,
i : le numéro de caméra ($i \in \{l, r\}$),
j : le numéro du point 3-D,
$\mathbf{p}_i(k)$: le $k^{\text{ème}}$ paramètre à estimer de la caméra i,
$\hat{\mathbf{p}}_i(k)$: le $k^{\text{ème}}$ paramètre connu de la caméra i,
$\mathbf{M}_j$: le vecteur de coordonnées du $j^{\text{ème}}$ point 3-D,

$\mathbf{m}_i j$: le vecteur de coordonnées de la projection 2-D du point $\mathbf{M}_j$ dans l'image i,
$d(\mathbf{m}_1, \mathbf{m}_2)$: distance euclidienne 2-D entre les deux points $\mathbf{m}_1$ et $\mathbf{m}_2$,
α_j : la pondération de l'erreur pour le $j^{\text{ème}}$ point,
β_k : la pondération de l'erreur pour le $k^{\text{ème}}$ paramètre de n'importe quelle caméra.

Nous avons confronté notre méthode de calibrage en ligne à plusieurs type de perte de calibrage, et dans le tableau 2.8 nous présentons celles étudiées dans la partie 2.4.1 (une variation de 0.5° des angles de tangage et de lacet et une variation de 0.5% de la focale d'une des caméras). Rappelons qu'une déviation de l'angle de lacet engendre une erreur de reconstruction importante et une erreur de reprojection faible. A l'inverse, une variation de l'angle de tangage engendre un forte erreur de reprojection, mais une faible erreur de reconstruction. Après calibrage en ligne, nous obtenons les résultats listés dans le tableau. L'élément essentiel qui se dégage de ces données est le fait que l'approche de calibrage en ligne fonctionne plus efficacement pout certains types décalibrages. En effet, la précision de reconstruction est très bonne (de l'ordre de 10^{-4} cm) lorsque le système est recalibré suite à une variation de tangage. Par contre, une variation de l'angle de lacet d'une caméra est très difficile à corriger.

Déviation	0.0	$p \pm 0.5°$	$y^r \pm 0.5°$	$f^r \pm 0.5\%$
RMS_X (cm)	0.0	$7 \cdot 10^{-4}$	$\mathbf{209.9}(\mathbf{6,6}\%)$	$56.5(1,7\%)$
RMS_Y (cm)	0	$2 \cdot 10^{-4}$	$\mathbf{27.6}$	11.0
RMS_Z (cm)	0	0.002	$\mathbf{898}(\mathbf{28}\%)$	$184(5,8\%)$
RMS_y (pix)	0	$2 \cdot 10^{-7}$	$\mathbf{3.49}$	3.45
$f_u^l \times f_v^l$ (pix)	800×800	800×800	800×800	800×799.7
$f_u^r \times f_v^r$ (pix)	800×800	800×799.9	799.9×799.9	799.9×800.2
b (cm)	40.0	39.9	39.9	40.0062
p (°)	0.0	0.50	0.001	0.003
$r^l \vert r^r$ (°)	$0.0 \vert 0.0$	$0.06 \vert 0.06$	$-0.08 \vert -0.08$	$0.12 \vert 0.12$
$y^l \vert y^r$ (°)	$0.0 \vert 0.0$	$0.95 \vert 0.77$	$-0.22 \vert -0.10$	$6.24 \vert 6.20$

TAB. 2.8: Les trois critères de qualité du calibrage en fonction du décalibrage. La configuration du système est détaillée en début de partie 2.3.3. Le calibrage utilise 800 points 3-D répartis dans le volume utile, et mis en correspondance avec une précision de $\sigma_{2\text{D}} = 0.1$pixel.

2.5 Conclusion

Nous avons présenté dans ce chapitre une méthode permettant d'évaluer la qualité d'un calibrage. Toute les aspects de la vie du système sont considérés, en commençant par le calibrage, puis le « décalibrage », et enfin le calibrage en ligne. Cette méthode, basée sur des simulations, permet d'explorer de multiples possibilités, aussi bien les configurations de système que celle du matériel de calibrage.

Les résultats présentés permettent d'identifier les principales difficultés auxquelles il faut faire face pour obtenir et maintenir un système stéréoscopique calibré. Il se dégage clairement que les angles de lacet des caméras doivent faire

l'objet de toutes les attentions, aussi bien pendant l'étape de calibrage que par la suite. En effet, comme le montrent les simulations de décalibrage, l'estimation précise de cet angle est déterminante lors du calcul des distances. Pour obtenir cette estimation avec suffisamment de précision, il est nécessaire de construire des mires couvrant tout l'espace de travail. D'autre part, les simulations de calibrage en ligne montrent que la correction à effectuer face à une variation des angles de lacets semble ardue.

Chapitre 3

Rectification

La rectification est une étape préliminaire permettant de simplifier considérablement le processus de corrélation stéréoscopique (chapitre 4). Dans ce chapitre, nous nous limitons au cas de caméras perspectives placées l'une à côté de l'autre. Dans ce cas, rectifier la paire d'images consiste à déformer ces dernières de manière à rendre les droites épipolaires parallèles à l'axe des ordonnées, simplifiant ainsi l'étape de corrélation stéréoscopique. En effet, après que les deux images aient été rectifiées, un point $\mathbf{m}_\mathrm{l}$ (de l'image I_l) trouve nécessairement son correspondant $\mathbf{m}_\mathrm{r}$ (de l'image I_r) sur la ligne de même ordonnée (voir la figure 3.2). L'intérêt de cette configuration canonique est qu'elle facilite la recherche lors du processus d'appariement des points.

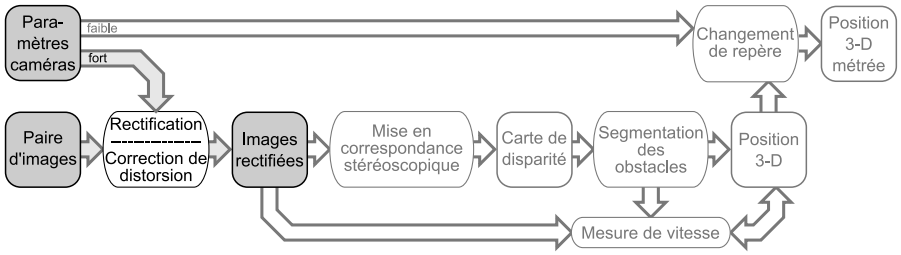


FIG. 3.1: La rectification, étape préliminaire à l'étape de mise en correspondance, se base sur les paramètres estimés du système.

Remarque : L'étape de rectification apporte une simplification, mais n'est pas indispensable au processus de mise en correspondance. En effet, même si elle n'est pas appliquée, la recherche d'appariement s'effectue malgré tout dans l'espace monodimensionnel : le long de la ligne épipolaire. La rectification transforme cette droite épipolaire pour la rendre strictement horizontale, ce qui simplifie son parcours.

Nous commençons par décrire les rectifications sous un aspect géométrique (partie 3.1), puis sous un aspect algébrique. La formulation algébrique, détaillée dans la partie 3.2, aide à établir des rectifications à partir de la matrice fondamentale, des correspondances de points, ou encore des paramètres de calibrage.

FIG. 3.2: Avec une rectification fronto-parallèle, la disparité dans les images gauche et droite (ici entrelacées) est strictement horizontale.

Bien que le calcul des rectifications ne dépende que de la configuration de la paire de caméras, il existe plusieurs rectifications possibles pour une même configuration. Dans la partie 3.3, nous expliquons justement quels sont les degrés de liberté des rectifications. Enfin, dans la partie 3.4 nous détaillons l'impact du choix de l'une de ces rectifications sur la reconstruction de surfaces.

Si les caméras du système stéréoscopique font apparaître des distorsions non linéaires, l'application inverse de cette fonction de distorsion permet de ramener les caméras au modèle perspectif (cf. 2.1.1). Des caméras alignées sur l'axe longitudinal, ou des caméras omnidirectionnelles représentent des configurations géométriques complexes, que nous n'abordons pas. Pour plus de détails sur les méthodes de rectification génériques, se référer à l'article de Pollefeys *et al.* [PKG99].

3.1 Interprétation géométrique

La rectification permet de ramener les caméras dans une configuration canonique dans laquelle les droites épipolaires sont parallèles à l'axe des ordonnées. Rappelons que les droites épipolaires sont définies par l'intersection du plan épipolaire Ω (défini par $\mathrm{C_l}$, $\mathrm{C_r}$ et $\mathbf{M}$, comme sur la figure voir la figure 3.3) avec les plans images I_l et I_r. Lorsque $\mathbf{M}$ parcourt l'espace, Ω décrit un faisceau de plans passant par la droite $(\mathrm{C_l}, \mathrm{C_r})$, et dans chaque image les droites épipolaires décrivent un faisceau de droites passant par l'intersection de $(\mathrm{C_l}, \mathrm{C_r})$ avec le plan image, appelée épipole. Pour que les droites épipolaires soient parallèles, il faut que leur point d'intersection, l'épipole, soit à l'infini (dans le plan image).

D'un point de vue géométrique, cela correspond à aligner les plans images

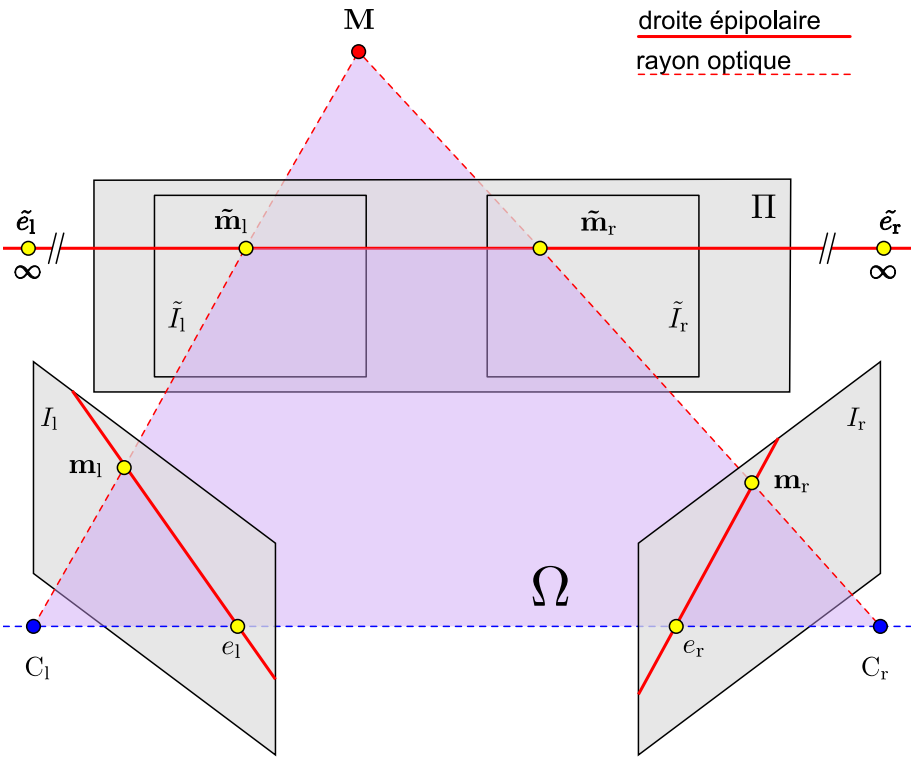


FIG. 3.3: La rectification consiste à reprojeter les images des plans caméras I_l et I_r sur un même plan Π, parallèle à la droite joignant les deux centres optiques C_l et C_r.

des caméras entre eux parallèlement à la ligne de base.

Remarque : Bien que nous expliquions les fondements de la rectification par des transformations 3-D, elle demeure une transformation 2-D.

3.2 Matrices de rectification

Nous allons montrer qu'une manière simple de rectifier une paire d'images consiste à appliquer une transformation homographique à chacune des deux images. Chacune de ces transformations est représentée par une matrice de rectification, notée R, et de taille 3×3. La question est de déterminer la paire de matrices de rectification R_l et R_r permettant d'obtenir des droites épipolaires horizontales. Une formulation algébrique du problème permet d'exprimer ces matrices à partir de la matrice fondamentale F, ou des paramètres de calibrage.

3.2.1 A partir de la matrice fondamentale

La matrice fondamentale F caractérise complètement la géométrie épipolaire d'un système stéréoscopique quelconque. Pour plus de détails, se reporter à la partie 2.2.2 page 27. Un minimum de 7 paires de points suffisent à la calculer. Mais des méthodes robustes au bruit, par exemple celle décrite par Zhang [Zha96], utilisent un grand nombre de points.

Nous savons que lorsque la paire d'image est rectifiée, les droites épipolaires sont horizontales. L'intérêt de cette configuration canonique est que deux points qui se correspondent ont nécessairement la même ordonnée. Reprenons l'équation 2.21 qui définit la matrice fondamentale F et ajoutons la contrainte que nous venons de définir. La matrice nouvelle fondamentale $\tilde{\mathsf{F}}$, caractérisant la relation des images rectifiées, est de la forme suivante :

$$\tilde{\mathsf{F}} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}. \tag{3.1}$$

Soient H_l et H_r les matrices de rectification, et F la matrice fondamentale du système avant rectification. F s'écrit :

$$\mathsf{F} \sim {\mathsf{H}_\mathrm{l}}^\mathsf{T} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \mathsf{H}_\mathrm{r} \tag{3.2}$$

Une décomposition en valeurs singulières permet d'écrire la matrice fondamentale sous la forme :

$$\mathsf{F} = \begin{pmatrix} e' & u_1 & u_2 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 \\ 0 & \sigma_1 & 0 \\ 0 & 0 & \sigma_2 \end{pmatrix} \begin{pmatrix} e^\mathsf{T} \\ {v_1}^\mathsf{T} \\ {v_2}^\mathsf{T} \end{pmatrix} = Q'\Sigma Q^\mathsf{T} \tag{3.3}$$

Comme F est définie à un facteur d'échelle près, on peut prendre pour valeurs propres $\sigma_1 = 1$ et $\sigma_2 > 0$. Une simple manipulation sur les trois matrices permet de se ramener à l'équation 3.2, où σ a été réparti de manière à symétriser l'expression :

$$\mathsf{H}_\mathrm{l} = \begin{pmatrix} e^\mathsf{T} \\ \sqrt{\sigma}{v_2}^\mathsf{T} \\ -{v_1}^\mathsf{T} \end{pmatrix} \text{ et } \mathsf{H}_\mathrm{r} = \begin{pmatrix} {e'}^\mathsf{T} \\ {u_1}^\mathsf{T} \\ \sqrt{\sigma}{u_2}^\mathsf{T} \end{pmatrix}. \tag{3.4}$$

Nous avons donc réussi à obtenir une paire de matrices de rectification compatible avec F à partir d'une simple décomposition en valeurs singulières de F. Notons qu'il n'y a pas un unique couple d'homographies H_l et H_r qui satisfasse l'équation 3.3.

3.2.2 A partir des paramètres de calibrage

Il est possible de déduire les matrices de rectifications à partir des paramètres intrinsèques et extrinsèques des deux caméras qui composent le système. Une manière simple de procéder est de calculer la matrice fondamentale F à partir de ces paramètres, puis d'appliquer la méthode précédemment décrite. Une autre approche [ATV00, Int01] consiste à changer l'orientation des plans images pour que les caméras droite et gauche visent la même direction, ramenant ainsi le

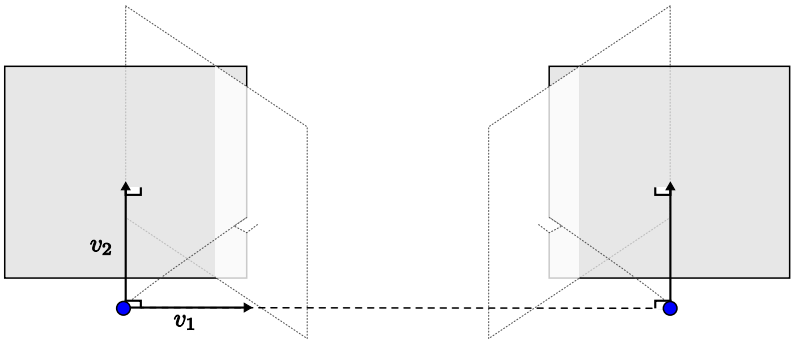


FIG. 3.4: La rectification c'est comme l'amour, ca n'est pas se regarder l'un l'autre, mais regarder ensemble dans la même direction

système dans la configuration canonique où les plans images sont confondus avec le plan Π.

Soient P_l et P_r les deux matrices de projection, K_l et K_r les matrices intrinsèques, R_l et R_r les matrices de rotation, $\mathbf{t}_\mathrm{l}$ et $\mathbf{t}_\mathrm{r}$ les vecteurs de translation, et C_l et C_r les centres optiques associés (pour plus de détail se référer à la description du modèle sténopé dont traite la partie 2.1.1 page 12). Les nouveaux axes v_1, v_2 et v_3 du repère des caméras rectifiées s'écrivent :

$$v_1 = \frac{C_\mathrm{l} - C_\mathrm{r}}{||C_\mathrm{l} - C_\mathrm{r}||}, \qquad v_2 = \frac{\mathsf{R}_\mathrm{l}''' \times v_1}{||\mathsf{R}_\mathrm{l}''' \times v_1||}, \qquad v_3 = \frac{v_1 \times v_2}{||v_1 \times v_2||},$$

avec R_l''' la troisième colonne de la matrice de rotation R_l. Notons que v_1 est aligné sur la ligne de base.

Dans la configuration rectifiée, les vecteurs de translation $\mathbf{t}_\mathrm{l}$ et $\mathbf{t}_\mathrm{r}$ restent inchangés. La nouvelle matrice de rotation R (commune aux deux caméras) est formée par la concaténation des vecteurs colonnes :

$$\mathsf{R} = \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix} \qquad \text{et} \qquad \mathsf{K} = \frac{1}{2}(\mathsf{K}_\mathrm{l} + \mathsf{K}_\mathrm{r}). \tag{3.5}$$

En reprenant les équations 3.5, les matrices de rectification doivent ramener le système dans sa configuration canonique, c'est à dire :

$$\mathsf{H}_\mathrm{l}(\mathsf{K}_\mathrm{l}\mathsf{R}_\mathrm{l} + \mathbf{t}_\mathrm{l}) = \mathsf{K}\mathsf{R} + \mathbf{t}_\mathrm{l} \qquad \text{et} \qquad \mathsf{H}_\mathrm{r}(\mathsf{K}_\mathrm{r}\mathsf{R}_\mathrm{r} + \mathbf{t}_\mathrm{r}) = \mathsf{K}\mathsf{R} + \mathbf{t}_\mathrm{r}. \tag{3.6}$$

En résolvant les équations 3.6, nous obtenons H_l et H_r de la forme :

$$\mathsf{H}_\mathrm{l} = \mathsf{K}\mathsf{R}(\mathsf{K}_\mathrm{l}\mathsf{R}_\mathrm{l})^{-1} \qquad \text{et} \qquad \mathsf{H}_\mathrm{r} = \mathsf{K}\mathsf{R}(\mathsf{K}_\mathrm{r}\mathsf{R}_\mathrm{r})^{-1} \tag{3.7}$$

Cet algorithme de rectification ne supporte que des pixels carrés. Avant de l'appliquer, il est donc important de vérifier que les caméras aient cette caractéristique.

3.3 Variété des rectifications

Les travaux de Zhang et Loop [LZ99] prouvent qu'un système stéréoscopique admet plusieurs rectifications valides. Plus précisément, Devernay [Dev97] démontre que l'ensemble des paires de matrices de rectification autorisées forment une variété de dimension 9. Nous allons commencer par expliquer comment, à partir d'un jeu de matrices H_l et H_r, trouver l'ensemble des matrices de rectification valides.

Après avoir expliqué à quoi correspondent les degrés de liberté, nous en donnerons une interprétation géométrique.

3.3.1 Degrés de liberté

Étant données les matrices de rectification H_l et H_r, il existe un autre couple de matrices H'_l et H'_r correspondant à la même géométrie épipolaire canonique :

$$\mathsf{H}_\mathrm{l}^\mathsf{T} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \mathsf{H}_\mathrm{r} \sim \mathsf{H}_\mathrm{l}'^\mathsf{T} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \mathsf{H}'_\mathrm{r} \tag{3.8}$$

L'équation 3.8 permet de déduire les matrices M_l et M_r tel que :

$$\mathsf{M}_\mathrm{l}\mathsf{H}'_\mathrm{l} = \mathsf{H}_\mathrm{l} \qquad \text{et} \qquad \mathsf{M}_\mathrm{r}\mathsf{H}'_\mathrm{r} = \mathsf{H}_\mathrm{r}. \tag{3.9}$$

Devernay [Dev97] prouve que ce couple de matrices est de la forme :

$$\mathsf{M}_\mathrm{l} = \begin{pmatrix} a & b & c \\ 0 & e & f \\ 0 & h & i \end{pmatrix} \text{ et } \mathsf{M}_\mathrm{r} = \begin{pmatrix} a' & b' & c' \\ 0 & e & f \\ 0 & h & i \end{pmatrix}, \text{ avec } \begin{vmatrix} e & f \\ h & i \end{vmatrix} \neq 0 \tag{3.10}$$

Les matrices M_l et M_r étant définies à un facteur d'échelle près, on peut fixer par exemple $i = 1$. L'ensemble des solutions est une variété de dimension 9, donnés par les 9 variables $a, b, c, e, f, h, a', b', c'$. En conclusion, n'importe quel couple de matrices de rectification valide peut être transformé par les matrices M_l et M_r pour donner de nouvelles matrices de rectification valides.

3.3.2 Interprétation géométrique

On peut donner une interprétation géométrique simple à chacun des 9 paramètres décrivant la variété des rectifications (voir la figure 3.5) :

- a et a' sont les facteurs de dilatation selon l'axe des x dans chacune des images rectifiées,
- b et b' inclinent les images rectifiées,
- c et c' sont les abscisses de l'origine de chacune des images rectifiées.
- e est le facteur d'échelle selon l'axe des y (commun aux deux images à cause de la contrainte épipolaire).
- f est l'ordonnée de l'origine,
- h correspond à une déformation « perspective » sur l'image rectifiée,
- et i est le facteur d'échelle global.

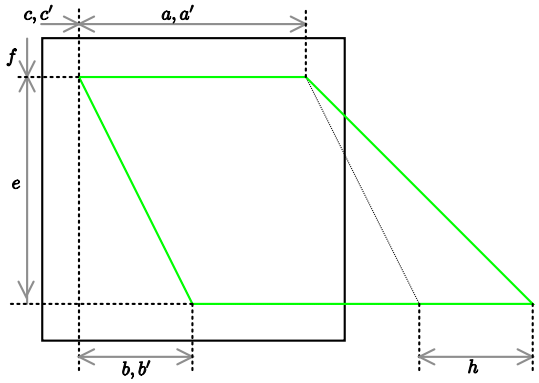


FIG. 3.5: Interprétation de chacun des paramètres et leur effet sur les rectifications.

3.4 Les surfaces d'iso-disparité

Lors du processus de reconstruction, les pixels de l'image gauche sont comparés à ceux de l'image droite afin de retrouver ceux qui se correspondent. La précision de l'appariement est limitée par la résolution des images. Comme la recherche est restreinte à la ligne épipolaire, c'est la résolution le long de cette ligne qui détermine la précision de la mise en correspondance. La surface d'iso disparité correspond à la surface dont l'ensemble des points qui, une fois projetés, ont une disparité donnée (où la disparité est la distance le long de la ligne épipolaire). Ces surfaces caractérisent l'imprécision et la discrétisation de la reconstruction dans l'espace 3-D. Par exemple, une variation d'un pixel de disparité correspond au passage d'une surface d'iso disparité à sa voisine.

Dans une première partie, nous allons reprendre la description de ces surfaces dans le cas général. Puis, dans une deuxième partie, nous allons tout d'abord détailler le cas de la configuration fronto-parallèle (correspondant à la rectification canonique) puis nous examinerons les surfaces en fonction du type de rectification.

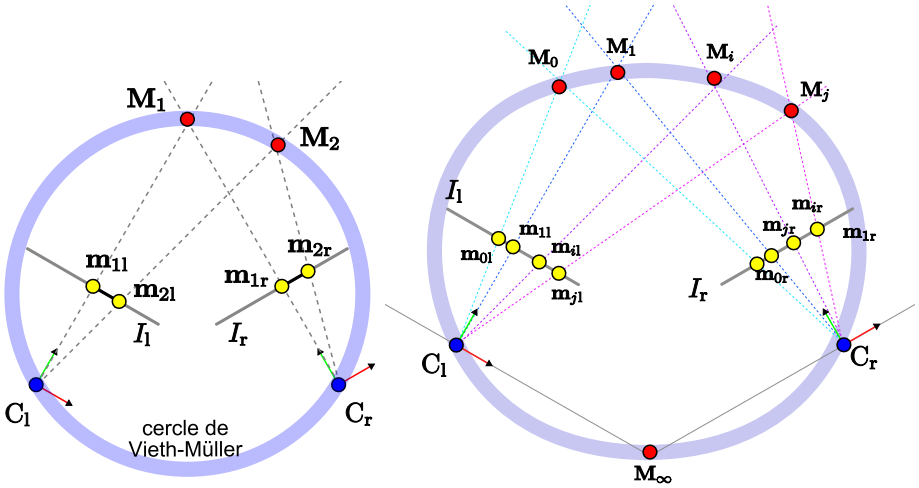
3.4.1 Le cas générique

L'horoptère

Comme l'illustre la figure 3.6(a), lorsque le regard est dirigé vers un point $\mathbf{M}_1$ (en faisant converger les axes optiques vers ce point), les images $\mathbf{m}_{1\mathrm{l}}$ et $\mathbf{m}_{1\mathrm{r}}$ de ce point s'impriment au centre de la fovéa de chaque œil. Par construction, le point visé $\mathbf{M}_1$ se projette au même endroit dans les deux yeux, et donc la disparité associée est nulle. Sans modifier la direction du regard, il y a d'autres points 3-D qui satisfont aussi cette propriété, comme par exemple $\mathbf{M}_2$ de la figure 3.6(a). L'ensemble de ces points 3-D forment une surface, nommée horoptère par François de Aguilon [Agu13] en 1613 (voir la figure 2.14). La section de l'horoptère est un cercle, le cercle de Vieth-Müller (voir la figure 3.6(a)), passant par les deux centres optiques C_l et C_r.

Les surfaces d'iso disparité

En étendant le principe de l'horoptère aux disparités non nulles, nous obtenons des surfaces d'iso disparité. En effet, la surface d'iso disparité correspond à la surface dont l'ensemble des points qui, une fois projetés, ont une disparité donnée. L'horoptère est une surface d'iso disparité, correspondant à la disparité nulle. Les travaux récents de Mandelbaum *et al.* [MMB+98] et de Pollefeys *et al.* [PS04] démontrent que dans une configuration quelconque, la section de cette surface est une conique qui passe par C_l, C_r et $\mathbf{M}_\infty$ (voir la figure 3.6(b)).



(a) (b)

FIG. 3.6: (a) Tous les points 2-D de disparité nulle sont nécessairement issues de points 3-D répartis sur une surface dont la section est le cercle de *Viet-Müller*. De manière plus générale, toutes les paires de points 2-D de même disparité sont issus de points 3-D répartis sur une surface dont la section (b) est une conique [PS04] passant par $\mathbf{M}_\infty$ et les centres optiques $\mathrm{C_l}$ et $\mathrm{C_r}$.

Remarque : La formulation décrite par Pollefeys [PS04] (voir la figure 3.6(b)) suppose que la focale horizontale f_x est identique pour les deux caméras.

3.4.2 Rectification fronto-parallèle

Après rectification, la configuration géométrique est différente, ainsi que l'échantillonnage en pixels des images. Dans cette partie, nous allons chercher à caractériser la surface d'iso disparité après avoir appliqué une rectification fronto-parallèle, c'est à dire qui suppose que les deux matrices de rotation $\mathsf{R}_\mathrm{l} = \mathsf{R}_\mathrm{r} = \mathsf{R}$ et les deux matrices de projection $\mathsf{K}_\mathrm{l} = \mathsf{K}_\mathrm{r} = \mathsf{K}$. Rappelons que K est de la forme suivante :

$$\mathsf{K} = \begin{pmatrix} fk_x & 0 & c_x \\ 0 & fk_y & c_y \\ 0 & 0 & 1 \end{pmatrix}, \tag{3.11}$$

où (c_x, c_y) sont les coordonnées du point principal (à l'intersection de l'image et de l'axe optique), fk_x est la focale en pixel selon l'axe horizontal de l'image, et fk_y la focale verticale (tels que décrits dans la partie 2.1.1 page 12). Les fonctions de projection s'écrivent alors comme suit :

$$\mathbf{m}_\mathrm{l} \sim \mathsf{KR}\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \mathsf{K}\mathbf{t}_\mathrm{l} \qquad \text{et} \qquad \mathbf{m}_\mathrm{r} \sim \mathsf{KR}\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \mathsf{K}\mathbf{t}_\mathrm{r}. \tag{3.12}$$

Pour simplifier, nous alignons arbitrairement le repère monde et celui de la caméra gauche . On a alors $\mathsf{R} = I$, $\mathbf{t}_\mathrm{l} = 0$ et $\mathbf{t}_\mathrm{r} = \begin{bmatrix} -B & 0 & 0 \end{bmatrix}^\mathsf{T}$, avec B la longueur de la ligne de base. Les équations 3.12 se simplifient en :

$$\mathbf{m}_\mathrm{l} = \begin{bmatrix} x_\mathrm{l} \\ y_\mathrm{l} \\ 1 \end{bmatrix} \sim \begin{bmatrix} \eta x_\mathrm{l} \\ \eta y_\mathrm{l} \\ \eta \end{bmatrix} = \mathsf{K}\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}, \tag{3.13}$$

$$\mathbf{m}_\mathrm{r} = \begin{bmatrix} x_\mathrm{r} \\ y_\mathrm{r} \\ 1 \end{bmatrix} \sim \begin{bmatrix} \zeta x_\mathrm{r} \\ \zeta y_\mathrm{r} \\ \zeta \end{bmatrix} = \mathsf{K}\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \mathsf{K}\begin{bmatrix} -B \\ 0 \\ 0 \end{bmatrix}. \tag{3.14}$$

La troisième ligne des équations 3.13 et 3.14 nous donnent $Z = \zeta = \eta$. La deuxième ligne nous permet également d'obtenir la relation simple $y_\mathrm{l} = y_\mathrm{r}$. Cette relation est logique puisque les images sont rectifiées. En reprenant les équations 3.13 et 3.14, nous pouvons développer la formulation de la disparité d_x et obtenir l'équation suivante :

$$d_x = x_\mathrm{l} - x_\mathrm{r} \tag{3.15}$$

$$= \frac{fk_x B}{Z}. \tag{3.16}$$

L'équation 3.16 donne la relation bien connue entre la disparité, la distance, la ligne de base et la focale des caméras (voir la figure 3.7). En remaniant l'équation 3.16, nous obtenons l'expression de la surface d'iso-disparité d_x :

$$Z = \frac{fk_x B}{d_x}. \tag{3.17}$$

Nous pouvons remarquer que si $d_x = 0$ (c'est à dire l'horoptère), alors le plan d'iso-disparité associé est situé à l'infini. Toujours avec la configuration fronto-parallèle , examinons maintenant l'écart Δ_Z entre deux plans d'iso-disparité en fonction d'une variation Δ_d de la disparités :

$$\Delta_Z = \frac{fk_x B}{d_x} - \frac{fk_x b}{d_x + \Delta_d} = -\frac{\Delta_d fk_x B}{d_x(d_x + \Delta_d)}. \tag{3.18}$$

Retenons que l'écart entre deux plans d'iso-disparité augmente d'autant plus que la disparité est faible.

3.4.3 Variété des surfaces d'iso-disparité

Nous avons vu dans la partie 3.3 qu'il existe une infinité de rectifications valides. Dans cette partie, nous allons étudier les surfaces d'iso-disparité pour

l'ensemble des rectifications, en fonction des 9 paramètres qui les décrivent. Rappelons que ces neuf paramètres forment les deux matrices de corrections M_l et M_l définies dans l'équation 3.10 :

$$\mathsf{M}_\mathrm{l} = \begin{pmatrix} a & b & c \\ 0 & e & f \\ 0 & h & i \end{pmatrix} \text{ et } \mathsf{M}_\mathrm{r} = \begin{pmatrix} a' & b' & c' \\ 0 & e & f \\ 0 & h & i \end{pmatrix}.$$

En les appliquant aux fonctions de projection de l'équation 3.12, nous obtenons les nouvelles fonctions de projection suivantes :

$$\begin{bmatrix} \eta x_\mathrm{l} \\ \eta y \\ \eta \end{bmatrix} = \underbrace{\begin{pmatrix} a & b & c \\ 0 & e & f \\ 0 & h & i \end{pmatrix}}_{\mathsf{M}_\mathrm{l}} \underbrace{\begin{pmatrix} fk_x & 0 & c_x \\ 0 & fk_y & c_y \\ 0 & 0 & 1 \end{pmatrix}}_{\mathsf{K}} \underbrace{\begin{pmatrix} X \\ Y \\ Z \end{pmatrix}}_{\mathbf{M}} \tag{3.19}$$

$$\begin{bmatrix} \zeta x_\mathrm{r} \\ \zeta y \\ \zeta \end{bmatrix} = \underbrace{\begin{pmatrix} a' & b' & c' \\ 0 & e & f \\ 0 & h & i \end{pmatrix}}_{\mathsf{M}_\mathrm{r}} \underbrace{\begin{pmatrix} fk_x & 0 & c_x \\ 0 & fk_y & c_y \\ 0 & 0 & 1 \end{pmatrix}}_{\mathsf{K}} \underbrace{\begin{pmatrix} X+B \\ Y \\ Z \end{pmatrix}}_{\mathbf{M}+\mathbf{t}_B}. \tag{3.20}$$

De la troisième ligne des équations 3.19 et 3.20, nous déduisons facilement que $\eta = \zeta = h(fk_yY + c_yZ) + iZ$. La disparité peut s'exprimer en fonction du point 3-D, des fonctions de projection et des 9 paramètres de rectification. Nous avons donc ηx_l qui s'écrit :

$$\eta x_\mathrm{l} = (afk_x)X + (bfk_y)Y + (ac_x + bc_y + c)Z \tag{3.21}$$

avec

$$\eta = (hfk_y)Y + (hc_y + i)Z \tag{3.22}$$

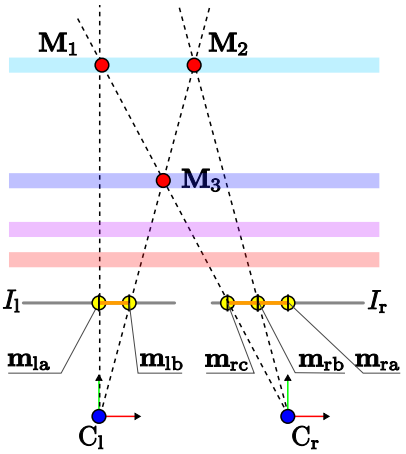


FIG. 3.7: Dans le cas d'une configuration de caméras fronto-parallèle, les points 3-D de même disparité sont répartis sur des plans 3-Dparallèles aux plans images I_l et I_r. Ici, les plans 3-D sont vus de dessus et schématisés par les bandes de différentes couleurs.

En faisant de même pour ηx_{r}, nous obtenons :

$$\eta x_{\mathrm{r}} = (a'k_x)X + (b'k_y)Y + (a'c_x + b'c_y + c')Z - a'k_x B. \tag{3.23}$$

Les équations 3.21, 3.22 et 3.23 permettent de développer $\eta d_x = \eta x_{\mathrm{l}} - \eta x_{\mathrm{r}}$, pour finalement aboutir à l'équation du plan du plan d'iso-dipsarité en fonction de d_x, des neuf paramètres a, a', b, b', c, c', e, f et h, des paramètres de projection fk_x, fk_y, c_x, c_y et de l alongeur de la ligne de base B :

$$0 = \alpha X + (\beta + \beta' d_x)Y + (\gamma + \gamma' d_x)Z + \delta \tag{3.24}$$

avec

$$\begin{aligned}
\alpha &= fk_x(a - a') \\
\beta &= fk_y(b - b') \\
\beta' &= -hfk_y \\
\gamma &= c_x(a - a') + c_y(b - b') + (c - c') \\
\gamma' &= i - hc_y \\
\delta &= fk_x a' B
\end{aligned} \tag{3.25}$$

3.4.4 Intersection des plans d'iso disparité

Dans cette section, nous nous intéressons à la répartition des plans d'iso-disparité, et plus particulièrement à leur intersection. En reprenant l'équation 3.24 pour deux disparités d_1 et d_2 (avec $d_1 \neq d_2$), nous formulons l'intersection par le système suivant :

$$\begin{cases} 0 = \alpha X + (\beta + \beta' d_1)Y + (\gamma + \gamma' d_1)Z + \delta \\ 0 = \alpha X + (\beta + \beta' d_2)Y + (\gamma + \gamma' d_2)Z + \delta \end{cases}. \tag{3.26}$$

La droite d'intersection existe si les deux plans ne sont ni confondus ni parallèles. Dans le cas ou les plans sont parallèles, la droite d'intersection est à l'infini. Dans le cas contraire, cette droite D s'exprime :

$$D = Q + \lambda[n_1 \times n_2], \tag{3.27}$$

avec Q un point de la droite et $[n_1 \times n_2]$ son vecteur directeur. L'opérateur $\times$ représente le produit vectoriel et n_1 et n_2 les vecteurs définis orthogonaux à chacun des plans :

$$n_1 = \begin{bmatrix} \alpha, & \beta + \beta' d_1, & \gamma + \gamma' d_1 \end{bmatrix}^{\mathsf{T}} \tag{3.28}$$

$$n_2 = \begin{bmatrix} \alpha, & \beta + \beta' d_2, & \gamma + \gamma' d_2 \end{bmatrix}^{\mathsf{T}} \tag{3.29}$$

En reprenant les vecteurs n_1 et n_2 non colinéaires, le produit vectoriel s'écrit :

$$[n_1 \times n_2] = \begin{bmatrix} (\beta + \beta' d_1)(\gamma + \gamma' d_2) - (\gamma + \gamma' d_1)(\beta + \beta' d_2) \\ (\gamma + \gamma' d_1)(\alpha) - (\alpha)(\gamma + \gamma' d_2) \\ (\alpha)(\beta + \beta' d_2) - (\beta + \beta' d_1)(\alpha) \end{bmatrix} \tag{3.30}$$

$$= \begin{bmatrix} (\beta'\gamma - \gamma'\beta)(d_1 - d_2) \\ (\alpha\gamma')(d_1 - d_2) \\ (\alpha\beta')(d_2 - d_1) \end{bmatrix} \tag{3.31}$$

$$= (d_1 - d_2) \begin{bmatrix} (\beta'\gamma - \beta\gamma') \\ (\alpha\gamma') \\ -(\alpha\beta') \end{bmatrix} \tag{3.32}$$

Comme le vecteur directeur $[n_1 \times n_2]$ de la droite D est défini à un facteur d'échelle près, nous pouvons simplifier par :

$$[n_1 \times n_2] = \begin{bmatrix} (\beta'\gamma - \beta\gamma') \\ (\alpha\gamma') \\ -(\alpha\beta') \end{bmatrix} \tag{3.33}$$

En remplaçant α, β, β', γ et γ' par leur expression (éq. 3.25), nous pouvons simplifier comme suit :

$$[n_1 \times n_2] = \begin{bmatrix} -fk_y(hc_x(a-a') + i(b-b') + h(c-c')) \\ fk_x(a-a')(i - hc_y) \\ fk_x fk_y h(a-a') \end{bmatrix} \tag{3.34}$$

Nous choisissons arbitrairement un point Q qui appartient au plan situé entre les deux caméras, d'équation $X = B/2$ (Q). En reprenant le système d'équation 3.26, alors Q est de la forme :

$$\begin{cases} X = B/2 \\ Y = \gamma' A \\ Z = -\beta' A \end{cases} \quad \text{avec } A = \frac{(\delta + \alpha B/2)}{\beta'\gamma - \beta\gamma'} \tag{3.35}$$

En remplaçant α, β, β', γ et γ' par leur expression (éq. 3.25), on obtient :

$$A = -1/2 \frac{k_x B(a' + a)}{k_y(i(b-b') + hc_x(a-a') + h(c-c'))} \tag{3.36}$$

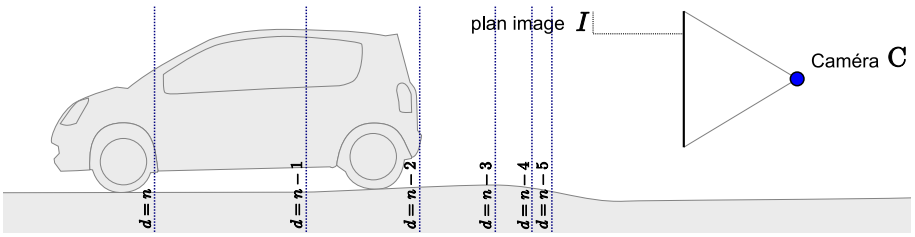
En remplaçant Q par sa définition ((3.35)) dans l'expression de la droite D d'intersection des plans d'iso-disparité, nous obtenons :

$$D = \begin{bmatrix} B/2 \\ \gamma' A \\ -\beta' A \end{bmatrix} + \lambda \begin{bmatrix} \beta'\gamma - \beta\gamma' \\ \gamma'\alpha \\ -\beta'\alpha \end{bmatrix} \tag{3.37}$$

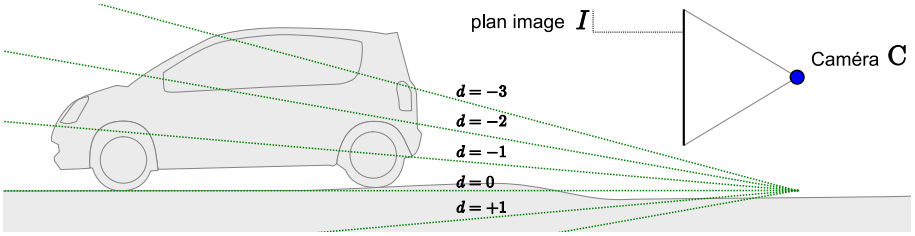
Nous constatons que, quelque soit λ, la droite D obtenue ne dépend ni de d_1, ni de d_2. Par conséquent, tous les plans d'iso-disparité s'intersectent en une unique droite. L'ensemble des plans forment ainsi un faisceaux de plans passant par cette droite. Comme nous allons le détailler plus tard, dans la configuration fronto-parallèle, le faisceau est formé de plans parallèles faisant face aux caméras (comme sur la figure 3.8(a)). Dans ce cas, la droite d'intersection D est à l'infini. Nous verront également que lorsque le système est rectifié par le plan du sol (voir la définition dans la partie 3.5.2), la droite D est à la verticale des centres optiques (figure 3.8(b)).

3.5 Choix d'une rectification

Comme nous venons de l'expliquer, plusieurs jeux de rectifications peuvent convenir pour un même système. Dans cette partie, nous allons développer les différents choix de rectifications que propose la littérature, en argumentant les avantages et inconvénients de chacun.



(a) Rectification fronto-parallèle



(b) Rectification par le plan de la route

FIG. 3.8: Avec une rectification fronto-parallèle (a), les plans d'iso disparité, ici vue en coupe, sont faces au caméras. Avec une rectification par le plan de la route (b), les plans d'iso disparité sont répartis en faisceau au dessus et au dessous de la surface de la chaussée.

3.5.1 Déformation minimale

Il a été montré [HZ04, LZ99] qu'il est préférable de choisir les rectifications déformant le moins possible les images. En effet la déformation est une opération qui dégrade l'information présente dans les images. Par exemple, comme l'illustre la figure 3.9, l'image de la route lointaine est étalée dans l'image rectifiée, créant un sur-échantillonnage. A l'inverse, l'image de la route proche est compressée, impliquant une perte de l'information. Par conséquent, pour minimiser la perte d'information, il faut minimiser la déformation.

3.5.2 Suivant un plan 3-D

Malgré le fait que la meilleure rectification, du point de vue de l'échantillonnage, soit celle qui déforme le moins possible l'image, il existe d'autres rectifications présentant également des propriétés intéressantes. En effet, comme nous l'avons montré, le choix choix d'une rectification conditionne la répartition des plans d'iso-disparité. Et le fait d'aligner ces plans d'iso-disparité, et plus particulièrement celui de disparité nulle, avec des éléments physiques de la scène peut apporter des propriétés intéressantes. Ces dernières étant liées au processus de mise en correspondance, nous les détaillons dans le chapitre concerné (chapitre 4). Rappelons seulement, que le plan de disparité nulle (horoptère) se projette à l'identique dans les deux images (voir la figure 3.11). Par la suite, nous utiliserons le terme « rectification par rapport à un plan Ω » pour désigner le fait d'aligner le plan de disparité nulle avec le plan 3-D Ω choisi.

Rectifier par rapport à un plan 3-D Ω consiste donc à rectifier la paire

d'images de manière à ce que tous les points de Ω se projettent à l'identique dans les deux images. Étant donné qu'un plan 3-D est défini par 3 paramètres, fixer le plan support c'est fixer 3 des degrés de libertés de la rectification. Rappelons que la rectification comporte 9 paramètres. Par conséquent, lorsque l'on fixe le plan Ω de disparité nulle, les rectifications possibles conservent une liberté de jauge de $9 - 3 = 6$ degrés de libertés.

Vue de dessus

Parmi toutes celles-ci, il en existe une qui fait coïncider (à un facteur d'échelle près) les coordonnées du plan 3-D avec les coordonnées de l'image. Ce type de rectification, lorsqu'elle est appliquée au plan du sol, donne l'impression d'une vue aérienne (voir la figure 3.9). Cette pseudo « vue de dessus », également applicable au cas monoculaire, peut permettre au conducteur de visualiser comment le véhicule est placé sur la chaussée, comme c'est le cas du système *Around View Monitor* de *Nissan* présenté dans la figure 3.10.

C'est aussi ce type de rectification qu'utilisait le premier système de détection d'obstacles proposé par Bertozzi *et al.* [BB97]. En prenant pour support le plan de la route, et sous l'hypothèse d'une route plane, cette dernière apparaît exactement identique dans les deux vues. Sous cette condition, une simple différence d'image permet de détecter tout ce qui dépasse de la chaussé. Mais cette approche, en plus d'être limitée au cas de routes planes, engendre une déformation telle que la perte d'information est préjudiciable.

Optimisée

Nous préférons utiliser une rectification qui minimise la déformation, ou à défaut, les calculs. Comme nous l'avons déjà détaillé, choisir un plan 3-D pour lequel la disparité est nulle revient à fixer 3 des 9 degrés de liberté, en laissant 6 degrés de liberté. Nous pouvons donc fixer arbitrairement 6 des degrés de liberté de la variété. En choisissant $a = e = i = 1$ et $b = c = f = h = 0$, nous simplifions l'équation 3.10 comme suit :

$$\mathsf{M}_\mathrm{l} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \text{ et } \mathsf{M}_\mathrm{r} = \begin{pmatrix} a' & b' & c' \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \text{ avec } \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix} \neq 0 \tag{3.38}$$

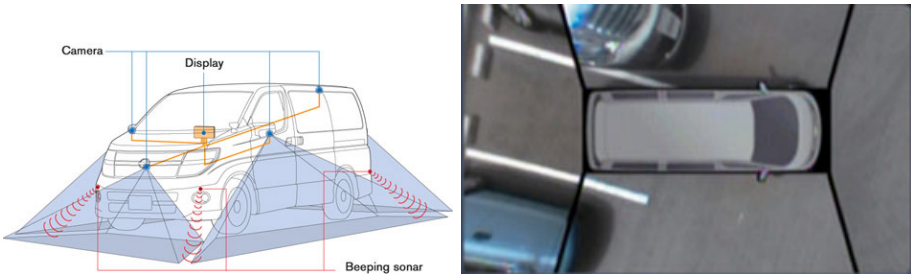
Une interprétation géométrique des trois paramètres a', b' et c' est donnée par la figure 3.12. Le fait que M_l soit égale à la matrice identité I dispense de l'appliquer. Ainsi, si le système fourni des images fronto-parallèles, la rectification d'une seule des deux images, par exemple à l'image droite, suffira pour obtenir une rectification par un plan 3-D. A noter que M_r conserve l'ordonnée des points. Les paramètres du plan 3-D (équation 3.24) se simplifient également comme suit :

$$\begin{aligned} \alpha &= k_x(1 - a'), & \beta &= -k_y b', & \beta' &= 0 \\ \gamma &= c_x(1 - a') - c_y b' - c', & \gamma' &= 1, & \delta &= k_x a' B \end{aligned} \tag{3.39}$$

En découle la simplification de l'équation 3.24 du plan 3-D :

$$\alpha X + \beta Y + (\gamma + d_x)Z + \delta = 0 \tag{3.40}$$

FIG. 3.9: Avec une rectification en pseudo vue aérienne, l'image est très déformée.



(a) (b)

FIG. 3.10: Le système *Around View Monitor* de *Nissan* utilise un réseau de caméras (a), toute rectifiées par le plan de la route, pour afficher une pseudo vue de dessus au conducteur (b).

FIG. 3.11: Avec une rectification par le plan de la route, la route apparaît à l'identique dans les images gauche et droite, ici entrelacées.

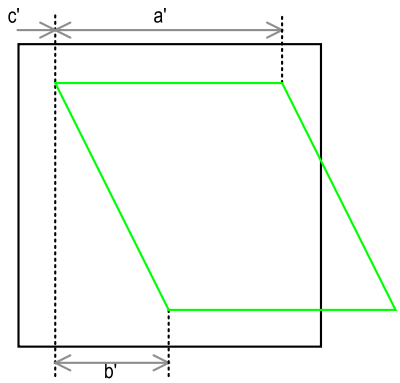


FIG. 3.12: Interprétation de chacun des 3 paramètres et leur effet sur la rectification.

3.5.3 Suivant le plan de la route

En supposant que le plan de la route est horizontal, l'équation 3.40 doit se simplifier en $\beta Y + \delta = 0$ lorsque d_x est nul. On en déduit facilement que $\alpha = \gamma = 0$. On peut donc poser le système :

$$\begin{cases} 0 = \alpha \\ 0 = \gamma \\ 0 = \beta Y + \delta \\ H = Y \end{cases} \tag{3.41}$$

En remplaçant α, γ, β et δ par leur expression (éq. 3.39), nous obtenons le système suivant :

$$\begin{cases} 0 = fk_x(1 - a') \\ 0 = c_x(1 - a') - c_y b' - c' \\ 0 = -fk_y b' H + fk_x a' B \end{cases} \tag{3.42}$$

En résolvant ce système, nous trouvons la valeur des paramètres de M_r en fonction des paramètres de calibrage :

$$a' = 1, \qquad b' = \frac{k_x}{k_y} \times \frac{B}{H}, \qquad c' = -C_y b',$$

où fk_x (resp. fk_y) est la distance focale suivant l'axe horizontal (resp. vertical) en pixel, C_y l'ordonnée du point principal, B la distance séparant les deux caméras (la ligne de base) et H la hauteur de la route par rapport aux caméras (comme les caméras sont au dessus de la route, alors $Y < 0$). Si les pixels sont carrés, alors le rapport des focales k_x/k_y est égal à 1.

Examinons maintenant le faisceaux de plan d'iso-disparité. Rappelons que $\alpha = \gamma = 0$ et que En reprenant l'équation du plan $\beta Y + d_x Z + \delta = 0$, nous développons :

$$\begin{cases} 0 = \beta Y + \delta \\ 0 = \beta Y + d_x Z + \delta \end{cases} \tag{3.43}$$

3.6 Conclusion

Après avoir expliqué en quoi consiste une rectification, nous avons détaillé ses paramètres. Dans une deuxième partie, nous avons présenté les surfaces d'iso disparité et leur particularités dans le cas d'un système stéréoscopique. En effet, dans cette configuration, ces surfaces sont des plans. Nous avons également décrit la position et la répartition de ces plans en fonction des paramètres de la rectification choisie. Le choix de ces surfaces peut avoir une influence sur le processus de mise en correspondance stéréoscopique.

Chapitre 4

Mise en correspondance stéréoscopique

Ce chapitre traite du processus de mise en correspondance stéréoscopique. Ce processus est l'une des étapes clef de la détection d'obstacles (voir la figure 4.1). En effet l'appariement est une étape préalable nécessaire à la triangulation/reconstruction. Tout au long de ce chapitre, nous supposons que les images sont rectifiées, pour pouvoir exploiter la géométrie épipolaire simplifiée.

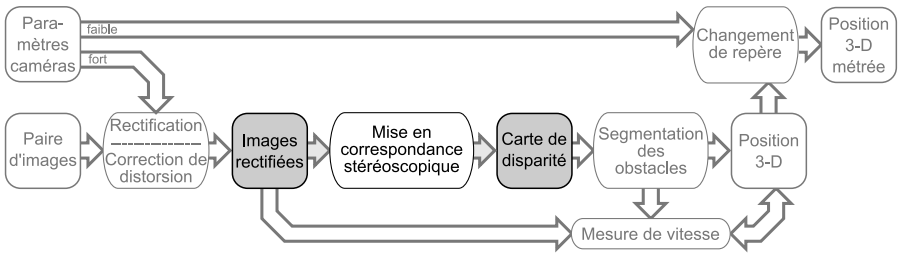


FIG. 4.1: La mise en correspondance stéréoscopique effectuée sur une paire d'images rectifiées permet de reconstruire l'information de profondeur sous la forme d'une carte de disparité.

Dans une première partie, nous détaillons les méthodes de l'état de l'Art. Après avoir décrit les principes de base de la méthode, nous détaillerons une implantation reprenant le même découpage que Scharstein et Szeliski [SS02]. Nous détaillerons également quelles sont les extensions intéressantes pour les applications automobiles.

Dans une deuxième partie, nous présentons une nouvelle approche permettant à la fois d'améliorer la qualité de l'appariement, et d'identifier certains plans dans la scène reconstruite (par exemple la route).

4.1 Algorithmes de mise en correspondance

4.1.1 Le principe

L'étape de mise en correspondance consiste à trouver les points qui se correspondent entre les images gauche et droite. Pour ce faire, les approches traditionnelles évaluent un à un tous les candidats $\mathbf{m}_\mathrm{r}$ qui pourraient correspondre à $\mathbf{m}_\mathrm{l}$ et retiennent le meilleur appariement (voir la figure 4.2).

La qualité de l'appariement est exprimée par un critère de corrélation. La fonction de corrélation quantifie la similarité d'apparence entre deux points. Comme la seule mesure de corrélation entre deux pixels n'est pas assez discriminante, elle est généralement calculée sur une fenêtre de pixels, entourant les dits points. Dans la figure 4.2, le point $\mathbf{m}_\mathrm{l}$ est caractérisé par la fenêtre de pixels A et $\mathbf{m}_\mathrm{r}$ par B. Il existe plusieurs méthodes pour mesurer la similarité entre deux fenêtres de pixels, et nous en détaillerons deux dans la partie suivante.

disparité	-11	-10	**-9**	-8	-7
corrélation	0.050	0.035	**0.027**	0.035	0.044

FIG. 4.2: La fenêtre de pixels A (caractérisant le point $\mathbf{m}_\mathrm{l}$) est comparée à des fenêtres B correspondant à plusieurs disparités (caractérisant le point $\mathbf{m}_\mathrm{r}$ pour plusieurs disparités). La disparité retenue est celle qui minimise la fonction de corrélation (ici en gras).

Lorsque les images sont rectifiées, et nous considérons que c'est toujours le cas, les candidats possibles pour $\mathbf{m}_\mathrm{r}$ se trouvent nécessairement sur la même ordonnée que $\mathbf{m}_\mathrm{l}$. Une paire de points est donc entièrement décrite par l'association des coordonnées du point $\mathbf{m}_\mathrm{l}$ et de l'écart d'abscisse entre $\mathbf{m}_\mathrm{l}$ et $\mathbf{m}_\mathrm{r}$. Cet écart est appelé disparité, et sera noté d dans la suite de ce manuscrit.

L'algorithme 1 présente une manière de calculer la disparité pour tous les points (pixels) de l'image I_l. En résulte une carte de disparité, qui associe chaque pixel de l'image I_l à une disparité. Cette carte de disparité est souvent représentée par une image d'intensités. Par convention, nous associons les disparités faibles (environ -100 dans la figure 4.3) à des intensités faibles. Inversement, les disparités hautes (0 dans la figure 4.3) ont de fortes intensités.

Algorithme 1 : L'algorithme de mise en correspondance stéréoscopique cherche, pour chaque pixel de I_l, le meilleur point dans I_r qui lui correspond.

```
pour (x,y) := Liste des pixels faire
    s̄ := le pire score possible ;
    pour d := Plage des disparités faire
        s := 0 ;
        pour (w_x,w_y) := Liste des pixels de la fenetre faire
            s := s + C ( I_l (x,y), I_r (x +d,y) ) ;
        fin
        si s meilleur que s̄ alors
            d̄ := d ;
            s̄ := s ;
        fin
    fin
fin
```

Fig. 4.3: La carte de disparité issue de la paire d'image de la figure 4.2. Les disparités basses (environ -100), qui correspondent à des points proches, sont représentées en sombre. Au contraire, les disparités hautes (proches de 0), affichées en clair, représentent des points éloignés.

4.1.2 Une implantation optimisée

Trop de méthodes de mise en correspondance stéréoscopique par corrélation sont publiées chaque année pour pouvoir en faire un état de l'art complet [SS02] (http ://vision.middlebury.edu/stereo/) . Donc, dans cette partie, nous n'en détaillerons qu'une seule, la plus simple, avec quelques variantes. Dans les grandes lignes, nous découpons le processus à la manière de *D. Scharstein* et *R. Szeliski* [SS02]. Dans cette implantation, le parcours des pixels et de la plage des disparités est inversé. C'est à dire qu'au lieu d'évaluer toutes les disparités pour un pixel (cf. algorithme. 1), nous évaluons tous les pixels de l'image pour chaque disparité. Cette inversion dans l'ordre des parcours permet d'optimiser certains calculs. Pour chaque disparité traitée, les étapes à appliquer successivement à la totalité de l'image sont :

1. le calcul de corrélation,
2. l'agrégation de la corrélation,
3. la mise à jour de la disparité,
4. et éventuellement un raffinement de la disparité.

Dans la suite de cette partie, nous décrivons quelques implantations pour chacune de ces étapes.

4.1.3 Le calcul de corrélation

Le critère de corrélation est une fonction d'erreur entre deux pixels. Plus il est bas, meilleur il est. Une différence absolue (AD) des intensités est le critère le plus simple. Évalué sur l'ensemble des pixels, le critère AD donne une image de différence absolue (voir la figure 4.4). La différence élevée au carrée (SD) est elle aussi très utilisée.

Certains critères plus complexes se basent sur une région d'intérêt normalisée, combinant les étapes de corrélation et d'agrégation. Sans rentrer dans le détail, ce type de critère permet de réduire l'influence d'une différence d'intensité globale entre les deux images (à cause de réglages différents des caméras). Une convolution préalable des images par le filtre passe haut $\sum_{i,j} I(x+i, y+j) - \bar{I}(x+i, y+j)$, aboutit à un résultat similaire. Pour plus de détails, se reporter à [Dev97].

4.1.4 La fonction d'agrégation de la corrélation

Les méthodes utilisant des fenêtres de corrélation agrègent le coût de corrélation en le sommant ou le moyennant sur une région support. Car la différence d'intensités entre deux pixels n'est pas un critère suffisamment discriminant. Nous avons implanté et testé deux types de région support en utilisant une fenêtre rectangulaire (voir la figure 4.5) ou la convolution gaussienne. Ces deux implantations donnent des résultats comparables pour des temps de calcul différents. Car il est connu que la convolution par un masque rectangulaire peut être calculée en un temps indépendant de la taille du masque, en utilisant l'optimisation dans les deux directions (expliquée graphiquement par la figure 4.6).

La taille de la région support est un paramètre important et doit être choisie en fonction de l'application. Une fenêtre trop grande va recouvrir plusieurs objets en même temps et donc ne pas saisir les détails de la scène. Au contraire,

Fig. 4.4: Le critère AD (différence absolue des intensités) est calculé entre chaque pixel de l'image gauche (voir la figure 4.2) et son homologue dans l'image droite.

Fig. 4.5: Chaque pixel de cette image affiche la moyenne du score de corrélation (ici AD) sur une fenêtre l'entourant de 7x7 pixels.

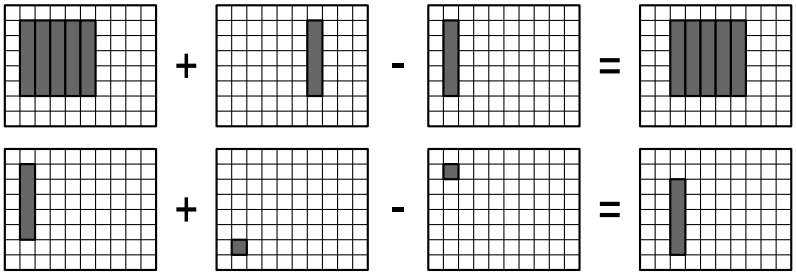


FIG. 4.6: Optimisation de l'agrégation du score dans les deux directions [FHZF93].

une fenêtre trop petite ne sera pas assez discriminante, et augmentera le ratio de faux appariements. Il existe des parades pour adapter la taille de la fenêtre. Mais celles utilisables en temps-réel [YPL04] n'apportent pas d'information supplémentaire, mais ne font que propager l'information déjà présente dans le voisinage du point.

4.1.5 La mise à jour

La disparité à retenir est celle ayant le score de corrélation le plus bas, stratégie habituellement appelée « gagnant l'emporte ». En procédant par mise à jour itérative, la recherche de ce minimum s'effectue en même temps que le calcul de corrélation. Soit $\bar{d}$ la meilleure disparité retenue parmi toutes celles déjà évaluées. Soit $\bar{s}$ le score de corrélation associé à $\bar{d}$. Si le critère de corrélation pour la disparité courante est meilleur que celui retenu dans $\bar{s}$, alors il faut mettre $\bar{s}$ et $\bar{d}$ à jour. Comme tous les pixels de l'image sont traités en même temps, $\bar{d}$ et $\bar{s}$ sont deux images. La première est l'image des disparités retenues et la deuxième, l'image des scores de corrélation retenus. Après avoir examiné toutes les disparités (et leurs scores), celles retenus au fil des mises à jour sont celles qui minimisent l'erreur de corrélation sur toute la plage des disparités testées.

4.1.6 Affinage de la disparité

La méthode que nous avons décrite se limite à l'évaluation de disparités entières, en pixel. Il est cependant possible d'obtenir une précision supérieure au pixel (dite sous-pixelique). Le principe d'une telle méthode consiste à prendre comme valeur de la disparité, non pas sa valeur entière d_0, mais celle minimisant la fonction passant également par ses voisines $d_0 - 1$ et $d_0 + 1$. Le nouveau minimum d_0^* est donc défini par la fonction de corrélation C en ces trois points :

$$d_0^* = \frac{C(d_0+1) - C(d_0-1)}{4C(d_0) - 2C(d_0+1) - 2C(d_0-1)}$$

Les détails du calcul sont disponibles dans l'annexe 8.1 (page 159). Le résultat de la fonction de corrélation C peut également être affiné pour la disparité non

entière d_0^* :

$$C(d_0^*) = -\frac{((C(d_0+1)-2)C(d_0+1)+C(d_0-1))C(d_0-1)}{8\times S(d_0)} + C(d_0).$$

avec S la fonction définie dans l'équation 4.1, également utilisée pour quantifier la qualité de l'appariement.

4.1.7 Les critères de qualité

Comme le laisse paraître la carte de disparité 4.3, le calcul de la disparité est parfois imprécis, voire défaillant. En effet, les zones peu texturées, ou les zones occultées dans l'une des vues constituent un réel défit pour ce type d'approche. Dans ces conditions, la valeur de disparité renvoyée par l'algorithme n'est pas pertinente, et il est préférable de l'ignorer, tout particulièrement dans le cas d'applications critiques comme c'est le cas des systèmes d'aide à la conduite. Pour identifier d'éventuels échecs, la littérature propose quelque critères, essentiellement basés sur l'analyse de la fonction de corrélation. Dans cette partie, nous présentons les trois critères que nous avons utilisés : la valeur de la corrélation, la forme de la fonction au minimum et l'unicité de la solution.

Seuillage sur le score de corrélation

Associée à la carte de disparité (voir la figure 4.3), la carte du score de corrélation donne une bonne indication de la qualité de la mise en correspondance. Un score élevé est le signe d'une mauvaise similarité entre les deux projections $\mathbf{m}_\mathrm{l}$ et $\mathbf{m}_\mathrm{r}$. Un seuillage sur les valeurs de corrélation peut donc permettre d'éliminer certains échecs du processus de mise en correspondance. Le choix du seuil se fait en fonction de la scène. Nous avons choisi un seuil élevé de manière à ne pas éliminer les bons appariements, quelque soit la scène.

Suppression des extrêmes

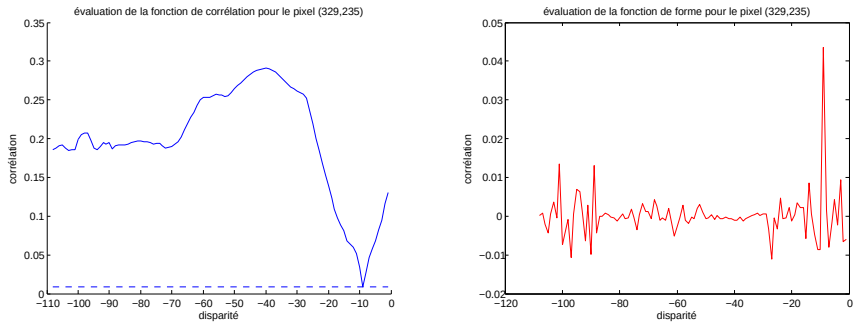
Lorsque la disparité retenue est la première ou la dernière testée, il y a fort à parier que le minimum ne soit pas pertinent. En effet, si la disparité réelle est en dehors de la plage des disparités testées, alors celle retenue est souvent un des deux extrêmes de la plage. Nous éliminons purement les pixels pour lesquels la disparité retenue correspond à un des deux extrêmes.

Forme de la fonction de corrélation au minimum

Si le point examiné est peu texturé, alors la mise en correspondance a de fortes chances d'échouer. Dans ce cas, une faible variation de disparité fait peu varier le résultat de la fonction de corrélation. On dit qu'elle est peu « marquée ». A l'inverse, un point fortement texturé engendre un minimum très marqué de la fonction (voir la figure 4.7). Pour détecter et ne garder que ce cas de figure, et ainsi minimiser le taux d'échec, nous sélectionnons à posteriori les disparités qui sont suffisamment marquées. Un minimum très marqué se quantifie par une forte réponse de la fonction suivante :

$$S(x,y,d) = \big(C(x,y,d-1)+C(x,y,d+1)\big) - 2\times C(x,y,d), \qquad (4.1)$$

où $C(x, y, d)$ est la fonction de corrélation pour le point de coordonnées (x, y) et de disparité d.



(a) $C_{\mathrm{SAD}}(\mathbf{m}_1, d)$ (b) $S(\mathbf{m}_1, d)$

FIG. 4.7: Évaluation du critère AD (a) et de la forme de cette fonction (b) sur une fenêtre de 7×7 pixel autour du point $\mathbf{m}_1$ de la figure 4.2, en fonction de la disparité. Le meilleur candidat, ici disparité -9, est celui qui minimise ce critère.

Ce critère de qualité est l'un des plus pertinents. Malheureusement, comme pour le score, il requiert le choix d'un seuil. Pour s'affranchir d'un seuil arbitraire, nous avons implanté un seuillage automatique, qui sélectionne 80% des pixels de la carte de disparité. Une façon de sélectionner ces points consisterait à partitionner l'histogramme des scores de corrélation, mais la répartition très inégale des scores rend une approche par histogramme inadaptée. Nous préférons une recherche par dichotomie de ce seuil. En pratique, une demi-douzaine d'itérations suffisent à trouver un seuil filtrant 20% (à $\pm 5\%$) de la carte.

Unicité du minima du score de corrélation

Les motifs réplétifs constituent un défi connu pour les méthodes de mise en correspondance stéréoscopique. Car, comme la montre la figure 4.8, si la taille de la fenêtre de corrélation est trop petite, les deux lignes blanches ont la même apparence, et peuvent être confondues. Pour éviter ce genre de confusion, Nedevschi *et al.* [NDF+04] proposent d'ignorer les points qui présentent plusieurs minima marqués.

4.1.8 Volume de reconstruction tronqué

Dans l'implantation que nous venons de décrire, tout les pixels d'une image sont « testés » pour toutes les disparités possibles. De cette manière, tout point 3-D visible dans les deux images peut être reconstruit. Comme l'illustre la figure 4.9(a), le volume formé par l'ensemble de ces points (la zone de couverture du système) est délimitée par l'intersection des deux cônes de vues issus des deux caméras. En fonction des applications, ce volume peut s'avérer plus grand que nécessaire, et donc engendrer des calculs inutiles. C'est dalleur l'hypothèse que font Nedeveschi *et al.* [NDM+05], en éliminant les points 3-D situés dans certaines parties du volume.

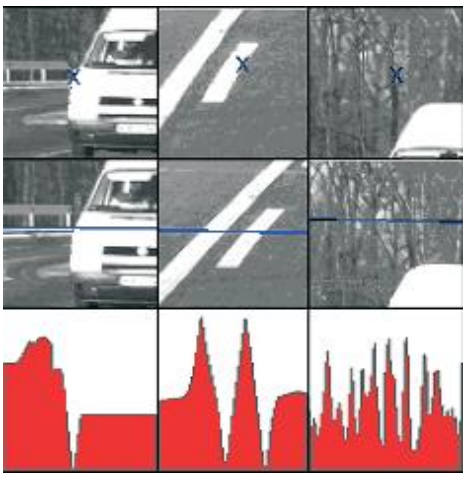


FIG. 4.8: De haut en bas : image gauche, image droite, et fonction de corrélation associée au point central de l'image gauche. Seule la fonction de corrélation du premier point (colonne de gauche) a un minimum qui n'est pas ambigü [NDF+04].

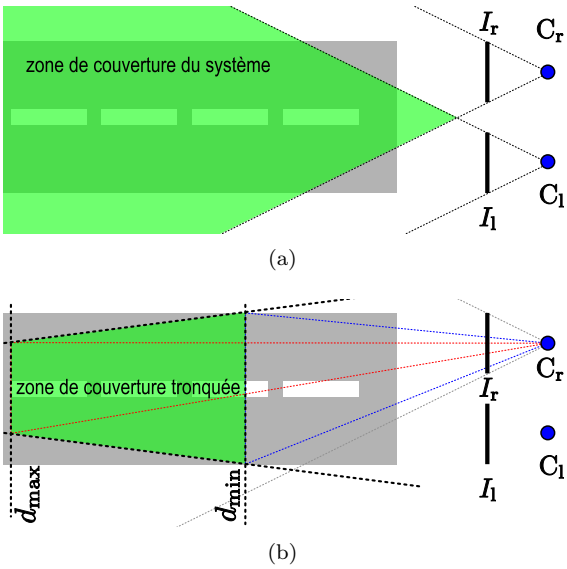


FIG. 4.9: La zone de couverture du système (a) est délimitée par l'intersection des deux cônes de vue. Pour changer cette zone, il faut adapter la taille des images en fonction de la disparité (b).

Par exemple, les points très en dessous ou au dessus de la surface de la route ne présentent pas d'intérêt pour les applications de détection d'obstacles. Dans cette partie, nous expliquons comment nous économisons du temps de calcul en tronquant ce volume (voir la figure 4.9(b)). Le principe consiste à ignorer certains appariements. En pratique, cela revient à rogner la paire d'images mises

en corrélation. Dans cette partie nous limitons le volume à l'aide de 6 plans de coupes. Deux de ces plans de coupes sont définis par la disparité minimum et maximum et correspondent aux distances minimum et maximum de reconstruction. Les quatre autres plans de coupes définissent un « couloir » dans le volume. Comme l'illustre la figure 4.9(b), le rognage dépend de la disparité. Pour des raisons pratiques évidentes, nous utilisons un recadrage de l'image par une région rectangulaire, et chacune des quatre arêtes définit un des 4 plans de coupe. La position et la taille de cette région sont linéairement dépendantes de la disparité. Ainsi, les étapes de corrélation, agrégation et sélection sont évaluées sur une portion de la paire d'images (voir la figure 4.10), économisant le calcul sur le reste de l'image.

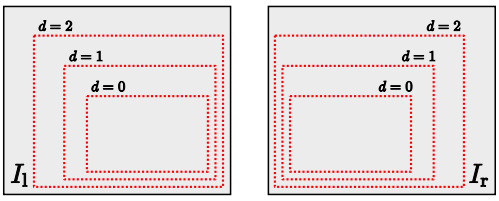


FIG. 4.10: La taille et la position de la sous partie de l'image (en pointillés rouge) dépend de la disparité d.

Attention, l'utilisation d'une telle optimisation induit un effet de bord. Comme nous l'avons précisé, dans la partie traitant des critères de qualité, il est préférable d'ignorer les disparités extrêmes. Mais cette sélection devient inefficace lorsque le volume de reconstruction est tronqué. En effet, dans ce cas, la plage des disparités réellement calculées ne correspond pas forcement à celle définie, et varie en fonction du pixel considéré. Pour éviter le calcul de cette plage complexe, nous choisissons d'ignorer cette étape de sélection. Mais ceci a pour effet d'engendrer des artefacts aux frontières du volume, qu'il faudra alors prendre en compte dans les processus qui suivent. Pour cette raison, nous n'utilisons pas cette optimisation dans les résultats ci-après.

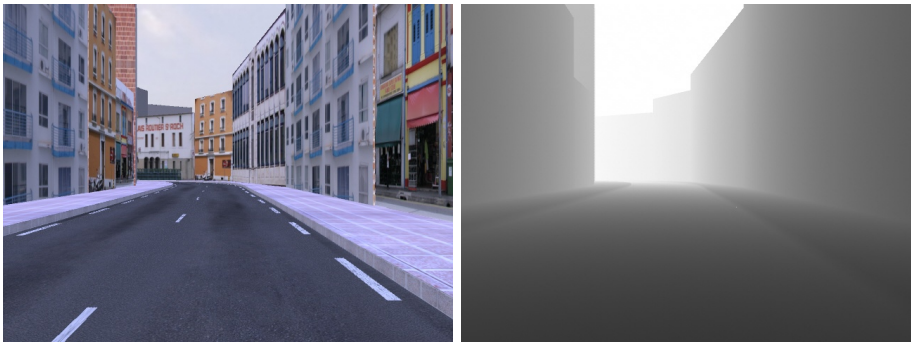
4.1.9 Résultats

Pour évaluer quantitativement la qualité de l'algorithme de mise en correspondance, nous reprenons l'évaluation utilisée par *Scharstein et Szeliski* [SS02]. Cette méthode consiste à dénombrer les pixels pour lesquels la disparité a été correctement estimée, c'est à dire qui satisfont :

$$|d - \bar{d}| \leq t, \tag{4.2}$$

où d est la disparité estimée, $\bar{d}$ la vraie disparité et t un seuil de tolérance en pixels. Par la suite, nous choisissons $t = 1$. Le nombre d'estimations correctes est alors normalisé par le nombre total de pixels. Rappelons que dans notre implantation, seules les meilleures estimations sont conservées (voir la section 4.1.7 traitant du seuillage adaptatif). Par conséquent, le taux des estimations correctes est exprimé par rapport aux nombre d'estimations et non par rapport aux nombres de pixels de l'image.

La nécessité de connaître la disparité de référence $\bar{d}$ nous a poussé à évaluer l'algorithme sur une séquence synthétique (voir la figure 4.11). Cette séquence de 150 paires d'images simule une paire de caméras équipées d'un capteur 1/3.2 pouces (5.680×4.536 mm) d'une résolution de 640×480 pixels avec un objectif de 6 mm de distance focale. Les caméras sont placées à 130 cm au dessus du sol, espacées de 40 cm l'une de l'autre et regardent droit devant. Cette configuration correspondant à la configuration canonique fronto-parallèle, les images n'ont pas besoin d'être rectifiées.



(a) image gauche (b) image de profondeur

Fig. 4.11: Image gauche (a) de la première paire issue de la séquence de paires d'images synthétiques et (b) la carte de profondeur associée : le blanc correspond à une distance de 100 m.

Les résultats que nous exposons ci-après correspondent à l'implantation suivante :

1. le calcul de corrélation est réalisé par une différence absolue
2. l'agrégation de la corrélation est faite sur une fenêtre carrée de 11 pixels de côté
3. la sélection de la disparité retient le meilleur score de corrélation (le plus bas)
4. et le raffinement de la disparité n'est pas appliqué.

La figure 4.12 correspond à carte de disparité calculée pour la première paire d'images de la séquence synthétique, et la figure 4.13 affiche en blanc les pixels correctement estimés.

Le ratio de disparités correctes est finalement calculé sur l'ensemble des 150 paires d'images. Les résultats sont repris par le graphique 4.14, sur lequel nous pouvons remarquer que le ratio des estimations correctes évolue entre 82% et 95%, pour une moyenne de 89,4%. Les dents de scie sont dues, non pas à une variation de la qualité du calcul de la carte de disparité, mais à des variations du nombre de pixels sélectionné par le seuillage adaptatif.

En examinant, sur la figure 4.13, la répartition des disparités correctes, nous remarquons que certaines surfaces sont mieux appariées que d'autres. Alors que la proportion d'appariements corrects semble maximum pour les immeubles, elle ne l'est pas pour les pixels correspondant à la route. Cette différence provient de l'hypothèse implicite qui est faite sur l'orientation locale de la surface. En effet,

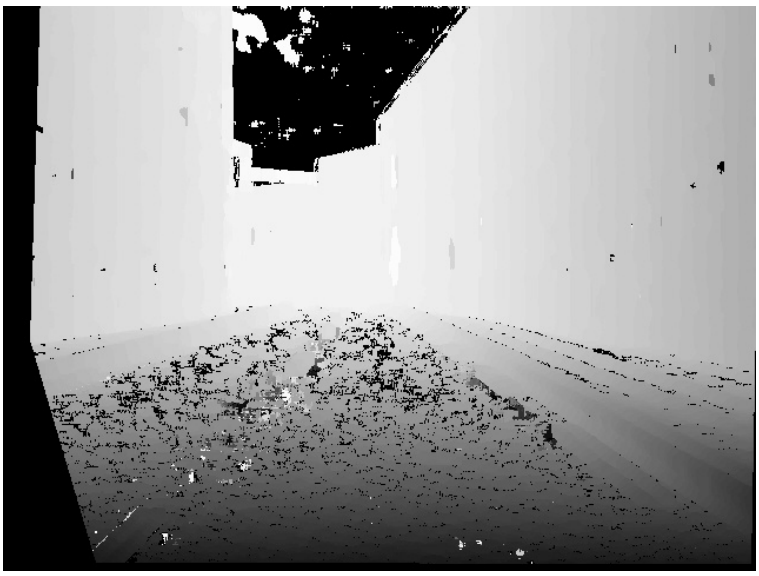


FIG. 4.12: La carte de disparité affiche les disparités de 0 (en blanc) à -80 (en noir). Les 20% de disparités éliminées par le seuillage adaptatif (voir le texte pour plus de détails) sont masquées par du noir.

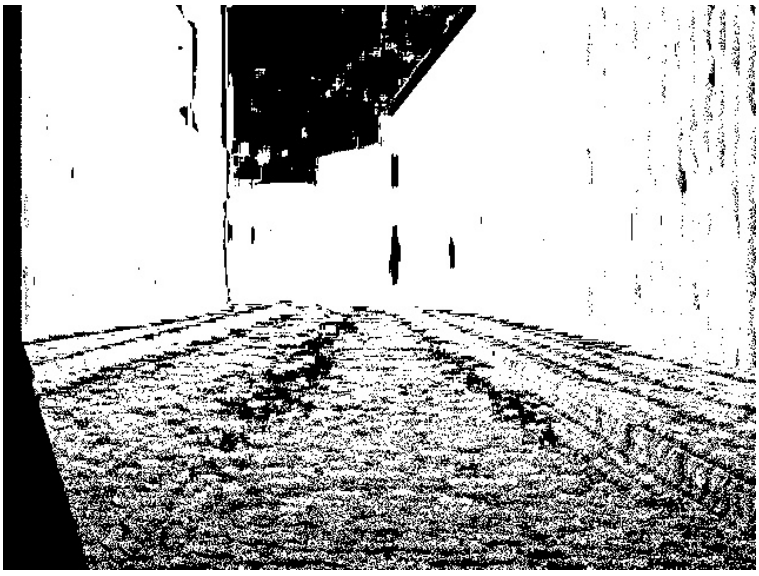


FIG. 4.13: Carte des disparités correctement estimées : les pixels blancs correspondent aux disparités estimées avec une précision ≤ 1 pixel.

l'étape d'agrégation telle qu'elle est implantée, suppose que toutes les surfaces de la scène font face aux caméras. Il est possible de prendre en compte d'autres orientations locales des surfaces, notamment pour améliorer le processus de mise en correspondance. Pour ce faire, nous proposons une nouvelle approche, que

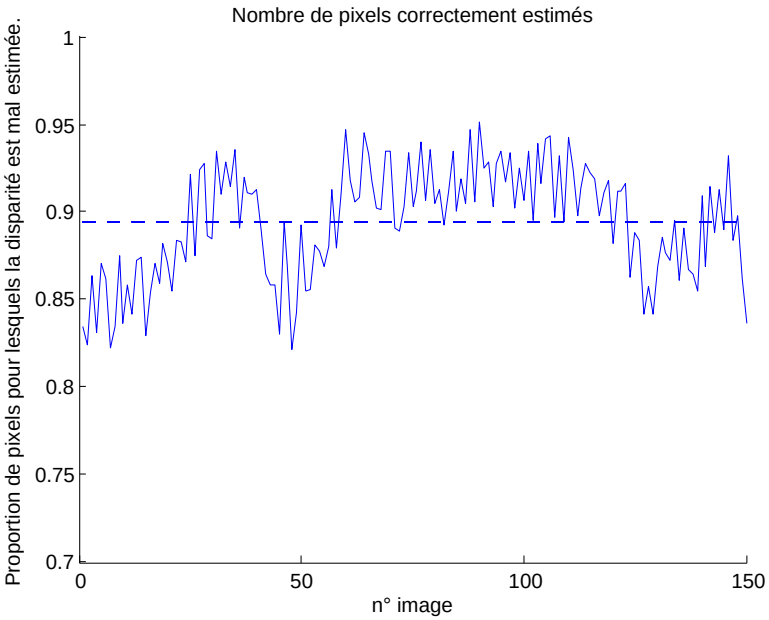


FIG. 4.14: Proportion des disparités correctement estimées en fonction du temps (trait plein) et la moyenne sur l'ensemble de la séquence (trait pointillé).

nous présentons dans la partie qui suit.

4.2 Calcul de l'orientation locale de la surface

Dans cette partie nous proposons une méthode permettant d'estimer l'orientation locale de la surface reconstruite en même temps que sa disparité. Nous utilisons alternativement le terme de dérivée de la disparité et d'orientation locale de la surface. Prendre en compte cette dernière présente deux propriétés intéressantes. Premièrement, les méthodes de corrélation basées sur des fenêtres (voir la partie 4.1) sont plus précises lorsque la dérivée de la disparité est prise en compte [DF94]. Deuxièmement, l'information d'orientation de la surface est une information pertinente pour les applications automobiles (ou de robotique mobile). En effet, on supposera qu'une surface horizontale est praticable par un véhicule, alors qu'une surface verticale est un obstacle.

Pour estimer l'orientation de la surface, un premier type d'approche [YAW03, JHG+03, HLPA04, Lem05] consiste à effectuer une première étape de reconstruction de la scène (par stéréoscopie), puis d'en extraire les surfaces. Ce type d'approche, que nous qualifions de géométrique (cf. 5.1.2), souffre de sérieuses limitations. Premièrement, le calcul de la dérivée à partir de la reconstruction est coûteux en temps de calcul. Deuxièmement, les techniques de corrélation stéréoscopique traditionnelles (présentées dans la partie précédente) supposent que la surface observée fait face aux caméras (voir la figure 4.16(a)), et non inclinée

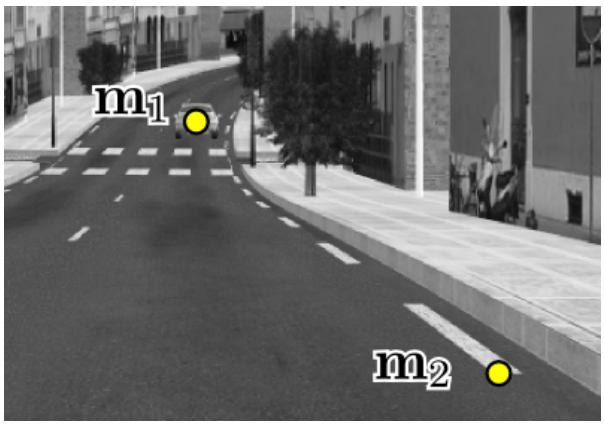


FIG. 4.15: Une scène routière laisse apparaître des surface d'orientations très différentes. Les obstacles, comme celui désigné par le point $\mathbf{m}_1$, sont plutôt verticaux. Au contraire, le point du sol $\mathbf{m}_2$ correspond à une surface horizontale.

(voir la figure 4.16(b)) comme c'est le cas de la route (voir la figure 4.16).

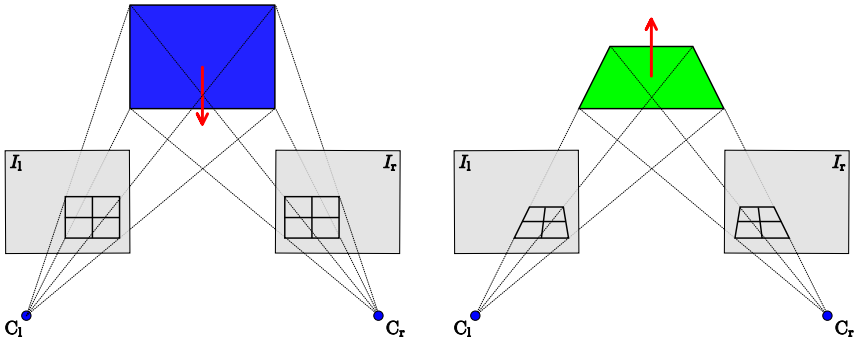


FIG. 4.16: Suivant son orientation, la surface peut apparaître différemment dans les deux images : une surface frontale (ici en bleu) apparaîtra à l'identique dans les deux images, alors qu'une surface horizontale (ici en vert) apparaîtra différemment dans les deux vues.

Un deuxième type d'approche permet de calculer conjointement la disparité et sa dérivée. Le principe consiste à utiliser l'effet de perspective qui, en fonction de l'orientation de la surface, engendre une différence d'apparence dans les deux images (voir la figure 4.16). Pour estimer la dérivée de la disparité, Robert et Hebert [RH94] raffinent la mise en correspondance avec une étape ultérieure de recherche de l'orientation. Cette recherche s'effectue en évaluant la corrélation avec des fenêtres déformées suivant 40 orientations possibles. La meilleure corrélation retenue donne les paramètres de l'orientation. Comme pour le premier type d'approche, la principale limitation de cette approche est qu'elle dépend de la réussite de la mise en correspondance. Si cette dernière échoue, ce qui est

probable lorsque la surface est inclinée (sol), alors tout le reste échoue également. De surcroît, une recherche exhaustive rend impossible une implantation temps-réel. Pour s'affranchir des limitations du temps de calcul, Hattori *et al.* [HM98] proposent un algorithme inspiré de celui de Lucas et Kanade [LK81]. Ce dernier déforme itérativement la fenêtre de corrélation, en se basant sur les gradients d'intensité, pour converger vers la déformation qui convient le mieux. Même si le procédé accélère grandement le temps de traitement, mais nécessite une initialisation correcte. De surcroît, il reste irréaliste de vouloir évaluer l'orientation de tous les pixels en un temps raisonnable.

Enfin, un troisième type d'approche simplifie le processus en ne traitant que deux orientations de surface : les surfaces horizontales et les surfaces verticales. Cette simplification convient parfaitement aux scènes routières, car ces deux orientations y sont prépondérantes. De plus, les calculs à effectuer deviennent raisonnables. Alors que Nakai *et al.* [NTH$^+$07] limitent le calcul aux obstacles potentiels le long de la trajectoire du véhicule, Williamson *et al.* [Wil98] effectue le traitement pour toute la scène. L'hypothèse de Nakai *et al.* étant trop réductrice, notre travail se base sur celui de Williamson *et al.* [Wil98]. Après avoir présenté le principe de notre approche, et les innovations par rapport au travail de *Williamson* , nous exposons notre implantation optimisée (à la manière de celle déjà présentée dans la partie précédente), en détaillant toutes les étapes.

4.2.1 Le principe

Différence d'apparence

Une surface 3-D peut avoir une apparence différente dans les deux images. Ce biais, dû à l'effet de perspective, est d'autant plus marqué que la surface est inclinée (voir la figure 4.16). La raison de cette déformation s'explique aisément à l'aide des surfaces d'iso-disparité. Tout d'abord, rappelons que l'algorithme de mise en correspondance standard utilise une région support (souvent une fenêtre carrée) pour mesurer la similitude entre deux points. Dans cette région support, les pixels des deux images sont tous comparés un à un, et ce, pour une disparité donnée. Cela signifie que c'est en réalité un ensemble de points 3-D de même disparité d qui sont considérés, comme l'illustre la figure 4.17. Nous avons démontré dans la partie 3.4 qu'un tel ensemble de points se répartit sur la surface d'iso-disparité associée, c'est à dire un plan 3-D dans le cas d'images rectifiées. Par conséquent, tester la similarité pixel à pixel des deux régions revient à tester la présence d'un élément de la surface d'iso-disparité. Si la surface réelle se confond effectivement avec la surface d'iso-disparité, alors les pixels des deux régions seront identiques (voir la figure 4.18). Au contraire, lorsque la surface réelle en est trop différente, alors les deux régions support seront d'aspects différents (voir la figure 4.22).

Rappelons que le processus de mise en correspondance, et plus particulièrement la fonction de corrélation C se base sur l'hypothèse qu'un point a la même apparence dans les deux images. Sous cette condition, il est évident qu'une différence d'apparence entre les deux régions détériore les performances du processus. Pour exemple, reprenons les deux points $\mathbf{m}_1$ et $\mathbf{m}_2$ de la figure 4.15. Le premier appartient à une surface verticale et le second à une surface horizontale. Avec une rectification fronto-parallèle, les surfaces d'iso-disparité sont des plans qui font face aux caméras (verticaux). La fonction de corrélation (voir

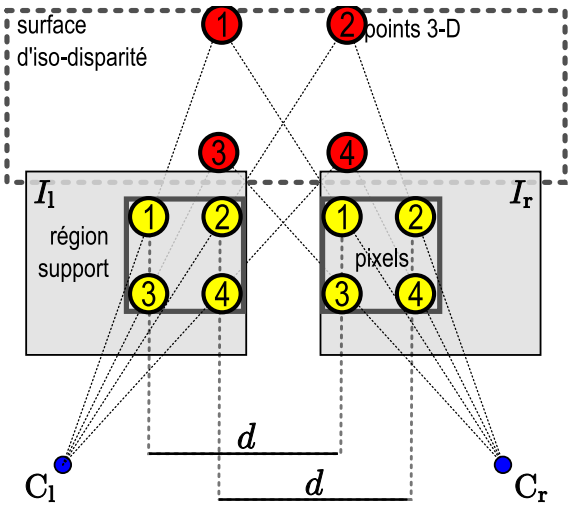


FIG. 4.17: Si l'ensemble des pixels d'une région ont la même disparité d, alors les points 3-D correspondant sont répartis sur la surface d'iso-disparité d.

les figures 4.18 et 4.19) retourne un minimum plus haut (c'est à dire moins bon) lorsque la surface est horizontale. C'est la raison pour laquelle la mise en correspondance échoue parfois pour les points de la route (horizontaux, voir la figure 4.12).

Utiliser une région qui prenne en compte l'orientation locale de la surface permettrait d'améliorer la mesure de corrélation, particulièrement dans le cas de surfaces inclinées. Malheureusement, ce paramètre n'est pas connu a priori, et constitue donc une ou plusieurs variables supplémentaires à estimer. La mise en correspondance consiste alors à effectuer une recherche dans un espace dont les dimensions sont la disparité d et les deux dimensions de l'orientation (o_1, o_2).

Estimer la variable d'orientation

La gestion de l'orientation locale de la surface nécessite l'introduction de nouvelles fonctions W_x et W_y, qui permettent de mettre en correspondance les pixels des deux régions supports. Ces fonctions prennent pour paramètres l'orientation (o_1, o_2) et la position (i, j) du pixel dans la fenêtre. En incorporant W_x et W_y, le critère C_{SAD} (somme des différences absolues sur une fenêtre carrée) s'écrit :

$$C_{\text{SAD}}(x, y, d, o_1, o_2) = \sum_{i,j} \big| I_{\text{l}}\big(x + i, y + j\big) - I_{\text{r}}\big(x + d + W_x(i, j, o_1, o_2), y + W_y(i, j, o_1, o_2)\big)\big|$$

où (x, y) sont les coordonnées du point dans l'image gauche, (i, j) les coordonnées à l'intérieur de la région support, d la disparité, et (o_1, o_2) les paramètres de l'orientation. L'algorithme que nous présentons ici est une adaptation de l'algorithme 1 présenté dans la partie précédente. Cette extension permet d'évaluer la fonction de corrélation C pour chaque quintuplet (x, y, d, o_1, o_2).

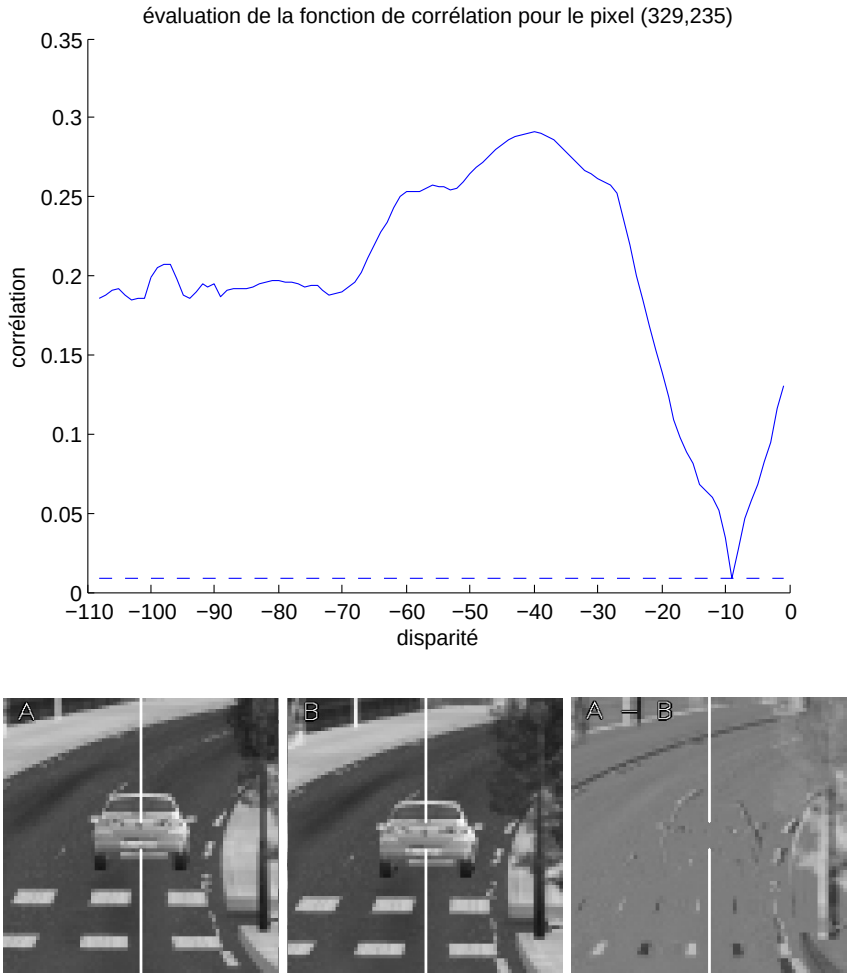


FIG. 4.18: La faible différence d'apparence entre les régions support A et B du point $\mathbf{m}_1$ entraîne un minimum de 0.01 de la fonction de corrélation C_{SAD} sur une fenêtre carrée de 11×11 pixels.

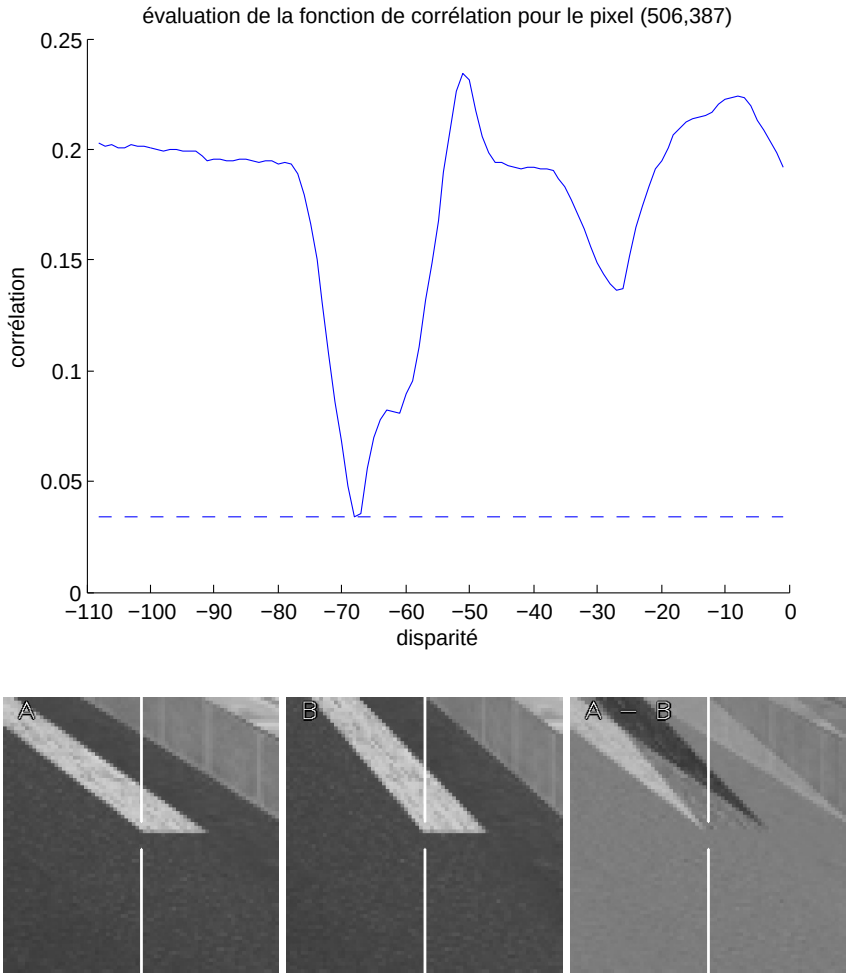


FIG. 4.19: Une différence d'apparence notable entre les régions support A et B du point $\mathbf{m}_2$ entraîne un minimum de 0.035 de la fonction de corrélation C_{SAD} sur une fenêtre carrée de 11×11 pixels.

Algorithme 2 : L'algorithme de mise en correspondance stéréoscopique adapté pour estimer aussi l'orientation locale de la surface.

```
pour (x,y) := liste des pixels faire
    s̄ := le pire score possible ;
    pour d := plage des disparités faire
        pour o := {horizontale, verticale} faire
            s := 0 ;
            pour (w_x,w_y) := liste des pixels de la fenêtre faire
                s := s + C ( I_l (x,y), I_r (x +d,y), o) ;
            fin
            si s meilleur que s̄ alors
                d̄ := d ;
                ō := o ;
                s̄ := s ;
            fin
        fin
    fin
fin
```

L'espace de recherche est donc multiplié par $(o_1 \times o_2)$. Même si nous n'utilisions que $o_1 \times o_2 = 40$ orientations différentes, comme Robert et Hebert [RH94], l'espace de recherche resterait très important. En effet, sous ces conditions, le calcul d'une carte de disparité nécessiterait 40 fois plus d'évaluations de la fonction de corrélation. Pour que le calcul soit réalisable en un temps raisonnable, **nous limitons l'évaluation à seulement deux orientations** : horizontale et verticale (voir la figure 4.16). Une telle simplification est possible car :

1. nous n'avons pas besoin d'une estimation précise de l'orientation de la surface pour discriminer les obstacles de la route,
2. les surfaces horizontales et verticales suffisent à modéliser la majorité des surfaces d'une scène routière (voir la figure 4.15).

4.2.2 Implantation optimisée

Optimisation du parcours des paramètres (x, y, d, o)

A l'instar de l'implantation standard (décrite dans la partie 4.1.2 page 78), cette implantation diffère quelque peu de l'algorithme. Le parcours de l'espace de variables (x, y, d, o_1, o_2) est réordonné dans le but d'optimiser certaines opérations. En effet, pour des raisons similaires à celles expliquées dans la partie précédente, le fait de traiter tous les pixels en même temps permet d'optimiser l'étape d'agrégation. Ainsi, pour chaque disparité, nous procédons aux étapes de corrélation, agrégation, sélection et affinage sur toute l'image. Dans cette implantation, le processus doit également être répété afin de tester chaque orientation.

Optimisation de la déformation des régions supports

Pour évaluer le critère de corrélation suivant plusieurs orientations de surface, il est nécessaire de définir les fonctions W_x et W_y. Rappelons que ces deux fonctions permettent de mettre en correspondance les pixels de la région A (comme sur la figure 4.18) avec ceux de la région B. En pratique, la mise en correspondance passe par une étape intermédiaire, qui, en déformant A ou B (voire les deux), permet de faire correspondre un à un les pixels de l'une et l'autre région.

Comme nous l'avons démontré dans le chapitre précédent, lorsque l'on considère des surfaces planes, les rectifications peuvent jouer le rôle des fonctions de transformations W_x et W_y (en effet, rappelons que l'ensemble des pixels de même disparité correspondent effectivement à un plan). Nous avons également expliqué que les plans d'iso-disparité peuvent être choisis, dans une certaine mesure, en définissant la fonction de rectification adéquate. Dans notre cas, on peut choisir deux rectifications qui correspondent à des surfaces verticales et horizontales.

La hiérarchie entre le parcours des disparités d et des orientations o n'a, quant à elle, que peu d'importance. Nous avons choisi de traiter toutes les disparités pour chaque orientation. Ainsi, le parcours des orientations se retrouve au sommet de la hiérarchie. Nous obtenons deux cartes de disparité (une pour chaque orientation) qu'il faut ensuite fusionner. Tout le calcul de la carte de disparité a déjà été détaillé dans la partie 4.1.2. La nouveauté de cette implantation réside dans l'utilisation de deux rectifications différentes. Reste à choisir les rectifications, et à fusionner les cartes de disparités. Nous expliquons ces deux aspects dans les deux parties qui suivent.

Algorithme 3 : Hiérarchie du parcours dans l'implantation optimisée de l'algorithme 2.

```
pour o := {horizontale, verticale} faire
    pour d := plage des disparités faire
        corrélation ;
        agrégation ;
        sélection ;
    fin
fin
```

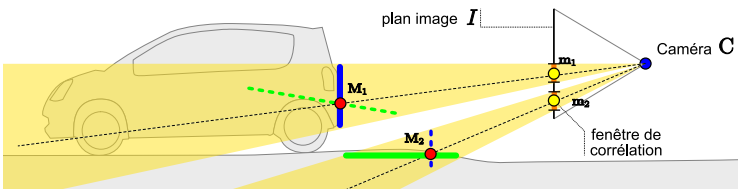
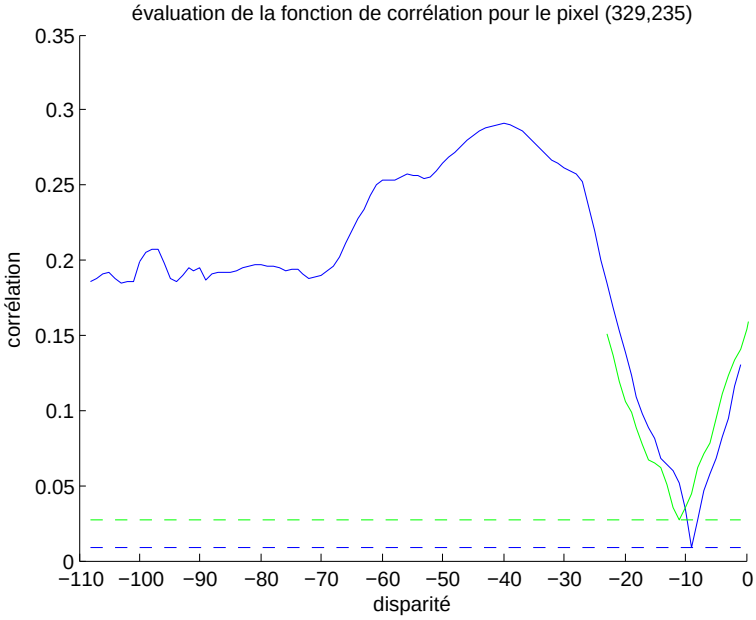


FIG. 4.20: Utiliser les deux rectifications de la figure 3.8 (p. 69) permet de tester deux orientations différentes pour chaque point 3-D (ici $\mathbf{M}_1$ et $\mathbf{M}_2$, appartenant respectivement à une surface verticale et horizontale).



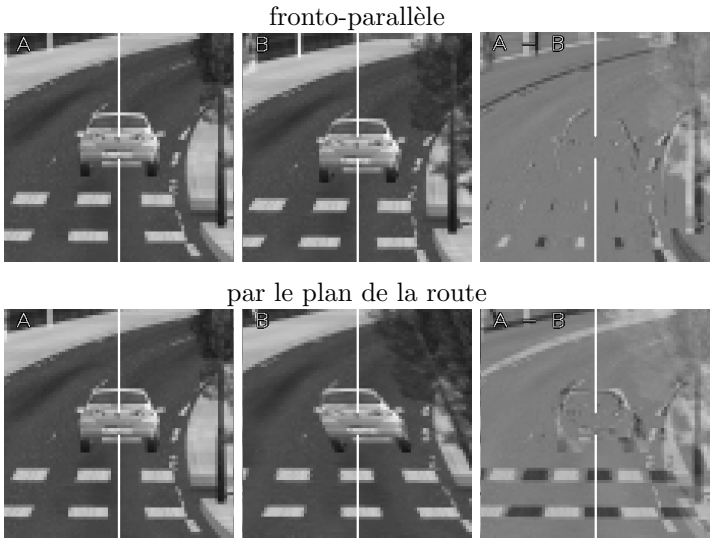


FIG. 4.21: Les régions supports A et B du point $\mathbf{m}_1$ se ressemblent plus lorsque les images sont rectifiées de manière fronto-parallèle. Cela signifie que la surface sousjacente est à tendance verticale.

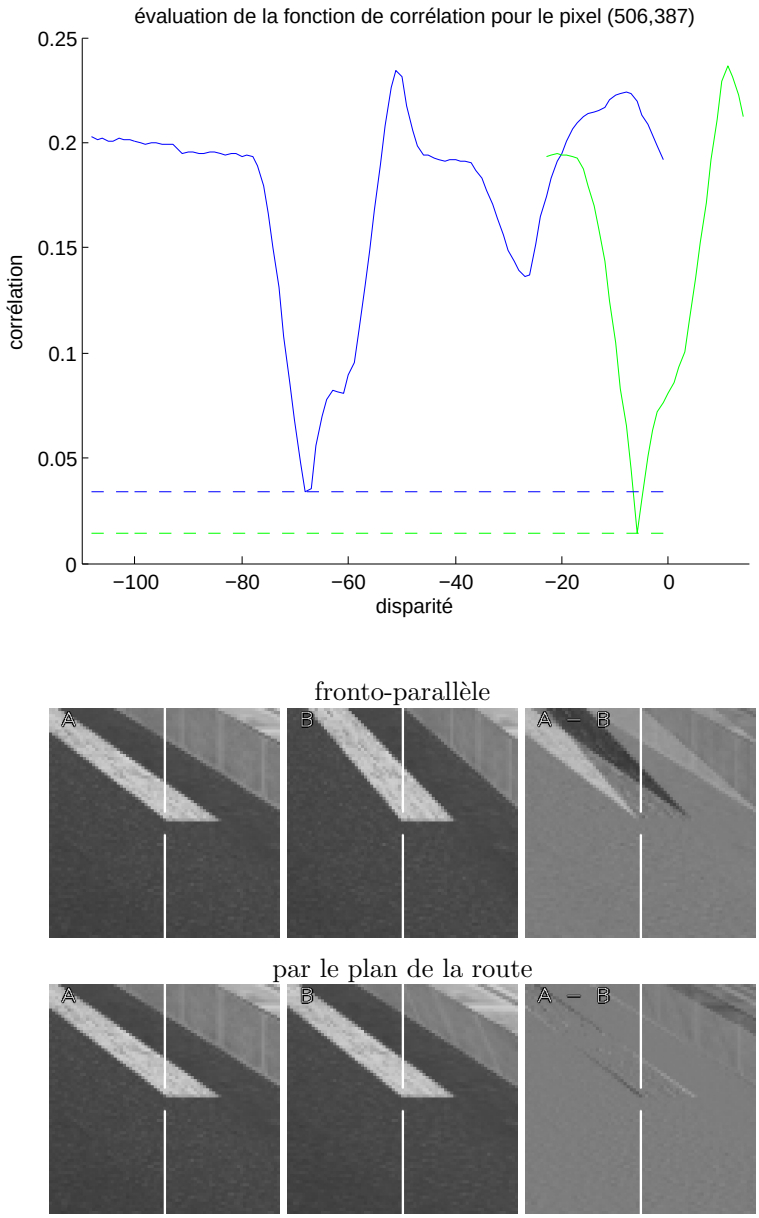


FIG. 4.22: Les régions supports A et B du point $\mathbf{m}_1$ se ressemblent plus lorsque les images sont rectifiées par le plan de la route. Cela signifie que la surface sousjacente est à tendance horizontale.

Créer les multiples rectifications

Dans cette partie nous expliquons comment calculer les multiples rectifications permettant de gérer les deux orientations. En effet, pour obtenir les surfaces d'iso-disparité qui nous conviennent, nous devons utiliser deux rectifications différentes. La première, fronto-parallèle, correspond à la configuration canonique. La deuxième fait correspondre le plan de la route à la disparité nulle.

La première paire de matrices R_l et R_r permet d'évaluer l'orientation verticale. Pour calculer R_l et R_r (pour plus de détails, se référer à la partie 3.4 traitant des surfaces d'iso-disparité) nous utilisons l'approche décrite dans la partie 3.2.2 (p. 60).

Pour calculer la deuxième paire de matrices, correspondant à l'orientation horizontale, une des possibilités serait de calculer directement deux nouvelles matrices R_l et R_r à partir des paramètres de calibrage. Nous préférons une autre approche qui consiste à modifier R_l et R_r, en utilisant la formulation de la variété des rectifications.

Pour ce faire, reprenons l'équation 3.10. Les matrices M_l et M_r qui y sont définies permettent de modifier la paire de matrices de rectification R_l et R_r de manière à obtenir une nouvelle paire de matrices de rectification valide.

Examinons maintenant le faisceau de plans d'iso-disparité obtenu avec de tels paramètres. En reprenant le plan d'équation 3.40 et en annulant les variables α et γ (éq. 3.41), nous obtenons l'équation du plan en fonction de la disparité $d_x \neq 0$:

$$\begin{cases} \beta Y + \delta = 0 \\ \beta Y + d_x Z + \delta = 0 \end{cases},$$

qui se simplifie en :

$$\begin{cases} Y = -\delta/\beta \\ Z = 0 \end{cases}.$$

Ce qui signifie que la droite d'intersection entre un plan d'iso-disparité quelconque et l'horoptère est la droite horizontale d'ordonnée $-\delta/\beta$ qui appartient au plan image Π (voir la figure 3.3).

Fusionner les résultats issus des multiples rectifications

Les scores de corrélation obtenus via l'une et l'autre des rectifications permettent de sélectionner celle qui correspond le mieux aux points reconstruits. Les cartes de disparité obtenues via l'une et l'autre des rectifications sont ainsi fusionnées en accord avec les orientations de prédilections. Mais cette fusion n'est pas directe, car les disparités obtenues par différentes rectifications ne sont pas exprimées dans le même référentiel. Par exemple, la disparité $d = 0$ correspond à une distance infinie dans le cas d'une rectification fronto-parallèle, alors qu'elle correspond à un point de la route avec le second type de rectification. Pour pouvoir fusionner les deux cartes de disparité, il est donc nécessaire d'exprimer toutes les disparités dans le même référentiel. Par souci de simplicité, nous choisissons de toutes les exprimer dans le référentiel correspondant à la configuration fronto-parallèle. Les disparités résultant d'une mise en correspondance sur des images rectifiées par le plan de la route sont converties. Le

changement de repère est caractérisé par la fonction ${}^{f}_{r}f({}^{r}d) = {}^{f}d$, et s'écrit :

$$ {}^{f}_{r}f({}^{r}d) = \left({}^{f}x_{\mathrm{r}} - a' \, {}^{f}x_{\mathrm{r}} - b' \, {}^{f}y_{\mathrm{r}} - c' \right) + {}^{r}d $$

où ${}^{r}d$ est la disparité obtenue après une rectification par le plan de la route, et d_f est la disparité obtenue après une rectification fronto-parallèle, $({}^{f}x_{\mathrm{r}}, {}^{f}y_{\mathrm{r}})$ sont les coordonnées du point dans l'image droite après rectification fronto-parallèle, x_{l} est l'abscisse du même point dans l'image gauche, et a', b' et c' sont les paramètres de la matrice R tels que définis dans l'équation 3.38. Nous pouvons remarquer que la fonction ${}^{f}_{r}f$ consiste à ajouter à la disparité ${}^{r}d$ un terme dépendant de $({}^{f}x_{\mathrm{r}}, {}^{f}y_{\mathrm{r}})$. Pour plus d'efficacité, ce terme peut être pré-calculé et mémorisé dans une carte de correction. La carte de disparité finale (voir la figure 4.23) contient des disparités issues des multiples rectifications.

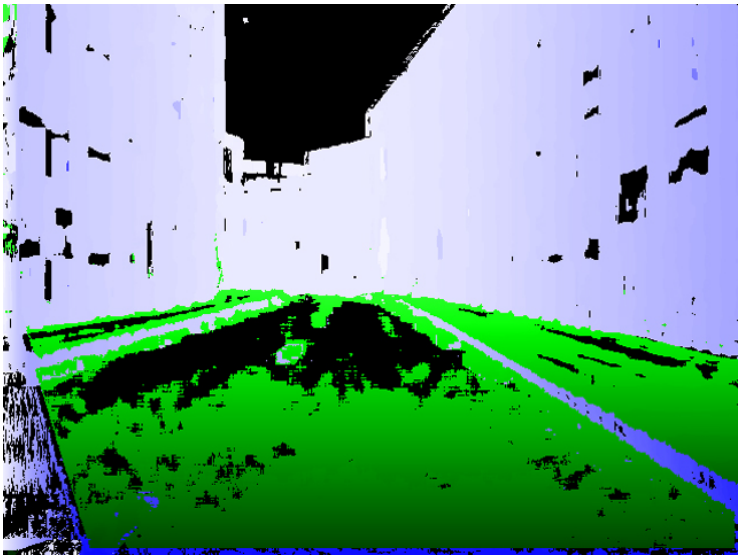


FIG. 4.23: La fusion des cartes de disparité. Pour distinguer les deux types d'orientations, les disparités issues de la rectification fronto-parallèle vont du bleu au blanc ($d = [-80 \ldots 0]$), et celles qui correspondent aux disparités issues de la rectification par le plan de la route vont du noir au vert ($d = [-80 \ldots 0]$). Les tâches noires correspondent à des estimations invalides ne satisfaisant pas les critères de qualité.

Quant aux cartes des autres critères (score de corrélation, forme, unicité, etc), elles sont fusionnées sans qu'il y ait besoin de conversion.

4.2.3 Validations

Dans cette partie, nous présentons une étude quantitative des performances de la méthode de mise en correspondance utilisant la double rectification. Le fait de prendre en compte l'orientation, même de façon simplifiée, permet non seulement de discriminer deux types de surfaces, mais aussi d'améliorer les performances du processus de mise en correspondance stéréoscopique. La performance est évaluée sur deux points :

1. le gain de précision sur l'évaluation de la disparité,
2. et la qualité de la segmentation sol/obstacle.

Les performances suivant ces deux aspects sont détaillées dans les deux parties qui suivent.

Gain de qualité

Lorsque les régions supports sont déformées pour s'adapter à l'orientation de la surface, la disparité correspondant au minimum de la fonction de corrélation est améliorée (comparer la fonction C_{SAD} pour les deux types de rectification dans la figure 4.22). Pour quantifier ce gain de performance, nous avons comparé les cartes de disparités obtenues avec et sans cette extension. Pour évaluer quantitativement la qualité d'un algorithme, nous reprenons la même séquence synthétique et nous appliquons la même méthode d'évaluation que celle décrite dans la partie 4.1.9 (p. 84).

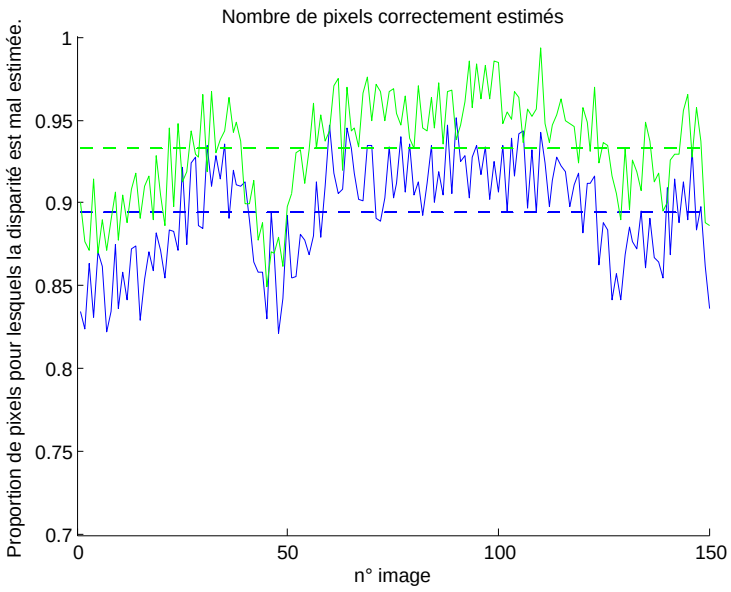


FIG. 4.24: Comparaison des performances de l'algorithme de mise en correspondance stéréoscopique sur une séquence synthétique avec une seule rectification (en bleu) et avec la double rectification que nous proposons (en vert). Les disparités étant estimées avec une précision inférieure au pixel sont considérées comme valides. En moyenne, avec une seule rectification, 89.4% des disparités retenues sont valides contre 93.2% avec la double rectification.

Le graphique 4.24 permet de comparer la qualité de la mise en correspondance classique (en bleu) avec celle utilisant la double rectification (en vert). Rappelons que seules les 80% des meilleures disparités sont conservées (voir la section 4.1.7 traitant du seuillage adaptatif), et que parmi ces dernières,

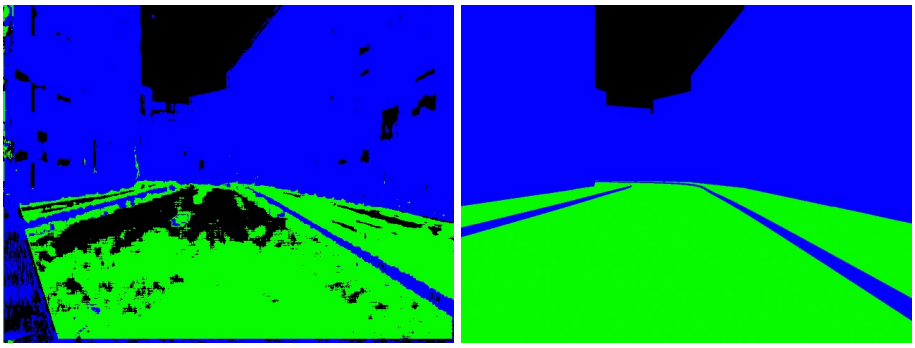
seules celles étant estimées avec une précision inférieure au pixel sont considérées comme valides. En moyenne, avec une seule rectification, 89.4% des disparités retenues sont valides contre 93.2% avec la double rectification. Ce gain de précision dépend évidement du nombre de pixels de l'image qui correspondent à des points du sol. En effet, l'ajout d'une rectification par le plan du sol améliore la précision essentiellement pour ces derniers.

Segmentation sol/obstacle

La segmentation permet de classer chacun des pixels dans une des trois catégories suivantes :

1. le sol, que nous représentons toujours en vert
2. les obstacles, que nous représentons toujours en bleu,
3. ou indéterminé, que nous représentons en noir.

La segmentation estimée, comme celle de la figure 4.25(a), est comparée à la réalité (voir la figure 4.25(b)) pour donner la proportion des classifications inexactes. Pour ce faire, nous dénombrons les pixels mal segmentés, c'est à dire



(a) estimation (b) vérité terrain

FIG. 4.25: Pour évaluer la qualité de la segmentation, nous comparons son estimation (à gauche) à la réalité terrain (à droite).

ceux estimés :

- sol alors que se sont des obstacles : r-nok (sur la figure 4.26),
- obstacles alors qu'ils appartiennent au sol : o-nok.

Les proportions de pixels mal segmentés se calculent comme suit :

$$\frac{\text{r-nok}}{t} \qquad \text{et} \qquad \frac{\text{o-nok}}{t},$$

avec

$$t = \text{r-nok} + \text{r-ind} + \text{r-ok} + \text{o-nok} + \text{o-ind} + \text{o-ok}.$$

Pour les notations, se référer à la figure 4.26. Le graphique de la figure 4.27 fait apparaître le taux de mauvaises classifications en rapport au nombre d'estimations. En vert nous avons le taux de fausses détections du sol (r-nok) et en bleu le taux de fausses détections des obstacles (o-nok). La courbe rouge est l'addition des courbes verte et bleu.

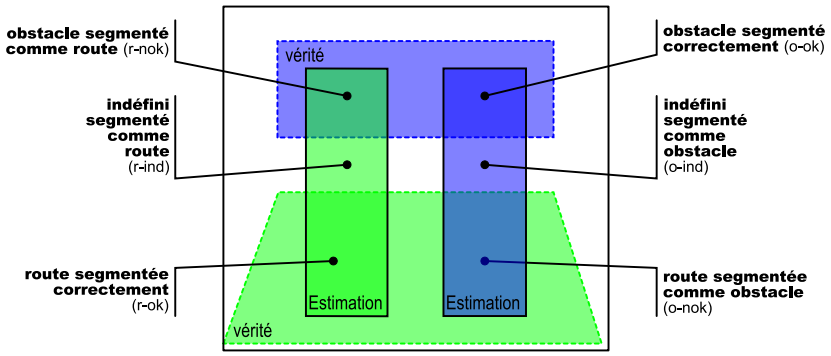


FIG. 4.26: Notations des comparaisons entre les classifications estimées et la réalité.

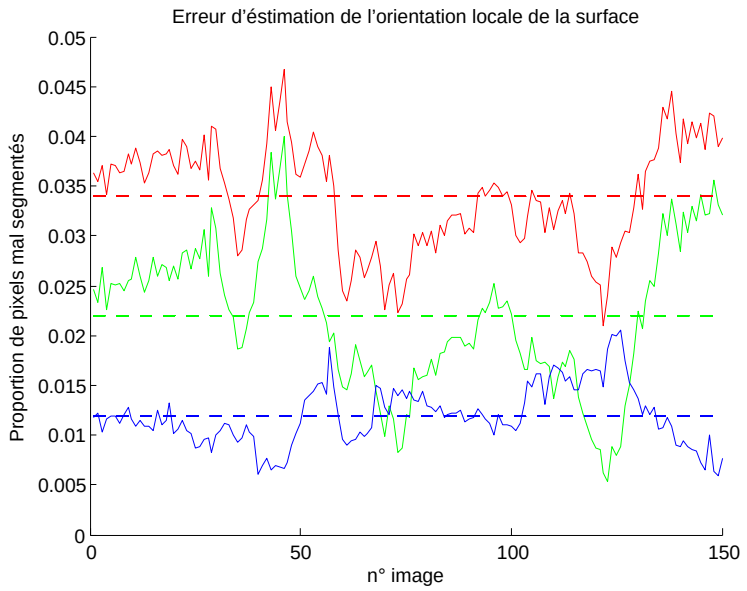


FIG. 4.27: Proportion des pixels mal segmentés : r-nok en vert, o-nok en bleu et la somme des deux en rouge.

Nous avons également effectué des essais sur des séquences réelles. Les résultats obtenus sont de qualité très inférieures. Après un examen, nous nous sommes aperçu que le calibrage nous fournit des rectifications erronées de plusieurs pixels. Ces imperfections dans les rectifications provient essentiellement à la présence de distorsions non radiales dans les images. La violation de la contrainte qui en résulte entraîne un décalage vertical entre deux région qui se correspondent. C'est ce biais qui met en défaut la méthode de calcul de l'orientation de surface.

4.3 Conclusion

Dans la première partie de ce chapitre, nous avons présenté un algorithme, ainsi qu'une implantation optimisée, d'un processus de mise en correspondance stéréoscopique de l'état de l'art. Dans une deuxième partie, nous avons présenté une extension du processus de mise en correspondance stéréoscopique, utilisant de multiples rectifications. Cette approche présente l'avantage de fournir de meilleures performances, mais surtout de fournir également une segmentation des points de la scène en sol/obstacles. L'intérêt d'une telle classification est évidente dans les applications automobiles. Les résultats obtenus sur des séquences synthétiques confirment les apports de cette extension. Notons seulement que les points n'ayant pas la même apparence dans les deux images (par exemple à cause d'un reflet) constituent encore un défit pour ce type d'approche.

Chapitre 5

Segmentation des obstacles

L'application automobile qui nous intéresse, à savoir le suivi de véhicule, nécessite la mise en place d'un processus spécifique à la mesure de vitesse (processus de mesure de vitesse sur la figure 5.1). Ce processus requiert des informations synthétiques sur les véhicules cibles, telles que leur taille apparente et leur position 3-D. En amont de la chaîne de processus, la mise en correspondance stéréoscopique (décrite dans le chapitre 4) fournit une reconstruction 3-D de l'environnement. Cette information, souvent calculée sous la forme d'une carte de disparité, nécessite d'être traitée pour fournir les données nécessaires au suivi de véhicules. Ce traitement intermédiaire s'appelle l'« extraction d'obstacles » ou « segmentation des obstacles ». Il permet d'extraire de la carte de disparité les informations de taille et position des cibles qui seront ensuite traitées par le processus de mesure de vitesse. Dans ce chapitre, nous commençons par décrire l'état de l'art, pour ensuite donner une solution simple, qui permettra de relier les processus de mise en correspondance stéréoscopique et de mesure de vitesse. Notre motivation n'est pas d'ajouter de nouvelles contributions sur cet aspect, mais plus modestement d'être en mesure de fournir des candidats au processus de mesure de mouvement afin de construire un système complet.

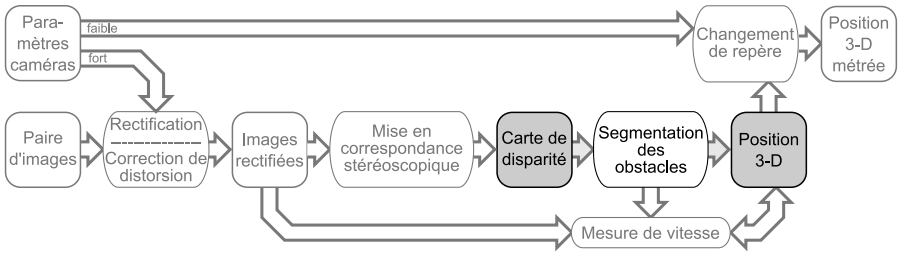


Fig. 5.1: La segmentation d'obstacles est le processus qui vise à extraire des cibles de la carte de disparité. Les cibles obtenues peuvent ensuite être suivies à l'aide des algorithmes de mesure de mouvement.

En première partie, nous commençons par exposer l'état de l'art en ce qui concerne ce processus de segmentation. En deuxième partie, nous présentons une technique de segmentation spécialement pensée pour répondre à la problé-

matique de suivi de véhicules. Pour finir, les résultats sur des séquences synthétiques et réelles permettent d'illustrer les forces et limites de la technique choisie.

5.1 État de l'art

Avant de demander à une machine de détecter les obstacles, il faut être capable d'en donner une définition. D'après le dictionnaire, « un obstacle est ce qui est susceptible d'arrêter ou ralentir le mouvement, la progression, le passage ». Cette définition de l'obstacle par ses conséquences ne nous donne pas les moyens de l'identifier, et c'est là tout le problème. Car la notion d'obstacle n'est pas clairement établie, et peut varier selon les auteurs. Pour certains, seule la chaussée ne constitue pas un obstacle. Pour d'autres, ce sont les objets en mouvement qui représentent une gêne pour la progression. Évidemment, le choix de la définition conditionne fortement l'approche envisagée. Parmi toutes celles que nous connaissons, nous distinguons trois choix possibles. Le premier choix consiste à définir les obstacles par leur apparence, le deuxième par leur géométrie, et le dernier par leur déplacement. Nous donnons une revue de l'état de l'art suivant ces trois définitions dans les trois parties qui suivent.

5.1.1 Apparence

Définir un obstacle par des caractéristiques de son apparence est une solution explorée par de nombreux auteurs [BHD97, SMBD02, HKT+98, YC06]. Les types d'obstacles les plus couramment rencontrés dans les scénarios routiers sont soit des véhicules, soit des piétons. L'un comme l'autre ont une signature visuelle qu'il est possible d'utiliser pour les détecter. Par exemple, les propriétés d'apparence telles que l'ombre portée, la symétrie, la couleur, les contours sont autant d'indices qui sont exploités pour détecter des véhicules. Quant aux piétons, ils ne sont pas en reste, et font l'objet de nombreuses attentions [BBFL02, BBB05]. Se référer à la revue de Zhenjiang *et al.* [LWLW06] pour plus de détails.

L'avantage de ces approches est qu'elles permettent de différentier les différents types d'obstacles et donc d'appliquer une politique adaptée pour chacun d'entre eux. Par exemple, on souhaitera protéger les vulnérables, comme les piétons ou les cyclistes. D'autre part, les méthodes basées seulement sur l'apparence sont applicables aussi bien à un système monoculaire que stéréoscopique. Cependant, dans le cas monoculaire, l'estimation des distances reste hasardeuse, puisqu'elle se base sur l'hypothèse qu'un obstacle est toujours posé sur une route plane.Cette hypothèse est mise en défaut dans de nombreux cas de figures (ombres portées, objet en porte-à-faux, ...). L'utilisation d'une carte de disparité permet de contourner ce problème. Restent deux autres défauts majeurs. Premièrement, le fait d'utiliser les caractéristiques visuelles limite la généricité de la détection. Par exemple, une méthode efficace pour les voitures, n'est pas applicable pour les piétons ou les cycles. Et la grande diversité des scènes routières rend impossible l'apprentissage de tous les types d'obstacles. Il n'y a donc aucune garantie que tous les obstacles puissent être détectés. Deuxièmement, si l'objet est occulté, même partiellement, la détection par l'apparence devient plus difficile, voire impossible.

5.1.2 Géométrie

L'approche géométrique consiste à définir l'obstacle par sa forme. De façon générale, les méthodes se concentrent sur la détection de l'ensemble des éléments constituant la route. Tout le reste représente les obstacles. Ces méthodes s'articulent en deux étapes. La première consiste à différencier le sol des obstacles. La deuxième, que nous appelons agrégation, consiste à synthétiser les caractéristiques des obstacles, afin d'obtenir leur taille, position, vitesse, etc. Les deux étapes sont généralement indépendantes. Nous décrivons un état de l'art pour ces deux aspects dans les parties qui suivent.

Segmentation de la route

La méthode de segmentation de la route dépend fortement du modèle choisi pour la représenter. Nous distinguons quatre types de modèles :

1. un plan calibré,
2. un plan auto-estimé,
3. une surface non-plane estimée,
4. une orientation locale de surface.

Dans la suite, nous abordons successivement les quatre types de modèles, et les approches associées.

Le modèle le plus simple est évidement le modèle utilisant un plan calibré. Ce modèle se base sur le fait que la chaussée se résume à un plan dont la position et l'orientation sont connues (par calibrage) et ne varient pas. La segmentation consiste alors à trouver les points qui correspondent au plan de la route. Maraninchi *et al.* [MR01] effectuent cette fonction directement à partir des images, à l'aide d'une rectification par le plan du sol (se référer au chapitre 3 traitant des rectifications pour plus de détails). Après avoir effectué cette opération, la route appairait à l'identique dans les deux images de la paire stéréoscopique. Ainsi, les pixels qui n'appartiennent pas à ce plan sont identifiés (comme sur la figure 5.2) par simple différence des intensités des pixels.

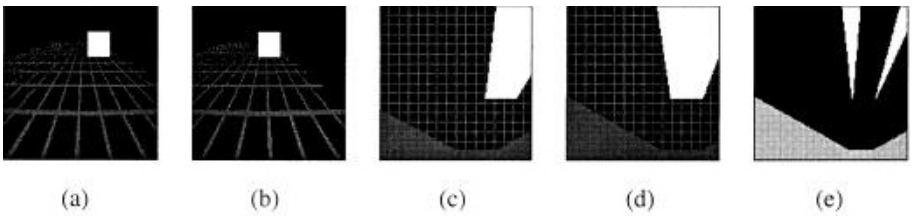


Fig. 5.2: Classification route/obstacle à l'aide d'une rectification par Broggi *et al.* [MR01] : les images gauche (a) et droite (b) sont rectifiées par le plan de la route (c,d). La route appairait de manière identique dans les deux images rectifiées (c,d). Une zone blanche dans l'image de différence (e) note alors la présence d'un obstacle.

Koller *et al.* [KLM94] utilisent les mêmes images rectifiées, mais calculent en plus la disparité en chaque pixel. Les pixels de disparité nulle (ou quasi nulle) sont des points du sol, alors que les autres représentent des obstacles. En plus

d'identifier les pixels affichant des obstacles, la disparité permet de retrouver leur position dans l'espace 3-D. Quant à Braillon *et al.* [BPCL06], ils examinent le flux optique de manière à discerner les mouvements qui correspondent au plan calibré. L'inconvénient majeur dont souffrent ces approches est qu'elles supposent que le sol est un plan 3-D figé et connu. Mais lorsque le véhicule se déplace, les variations importantes de roulis et de tangage qu'il subit empêchent d'utiliser un tel modèle.

Pour ces raisons, il est préférable d'estimer la position du plan de la route à la volée. Dès lors, le problème ne se limite plus à segmenter la scène, mais aussi à estimer les paramètres du plan du sol. Ce processus passe généralement par la reconstruction complète de la scène (par exemple par la construction d'une carte de disparité, comme celle de la figure 4.12), désormais possible à cadence vidéo [YPL04]. Le problème consiste alors à identifier, dans la reconstruction, les parties de la scène qui forment le plan de la route. La diversité des scènes routières et les erreurs de reconstruction rendent la tache complexe. Pour résoudre ce problème, Labayrade *et al.* [LAT02] présentent une approche robuste passant par la construction d'une troisième représentation de la scène, appelé *v-disparité*. La v-disparité, à l'instar de la carte de disparité, peut être représentée par une image (voir la figure 5.3). En fait, c'est une matrice à deux dimensions, dont chaque ligne est l'histogramme de la carte de disparité à la ligne correspondante. Avec cette représentation, il est facile d'associer une droite au

FIG. 5.3: L'approche de Labayrade *et al.* [LAT02] consiste à créer une pseudo vue de coté de la carte de disparité, appelée v-disparité (v pour l'axe v de l'image) dans laquelle se dessine le profile de la route et des obstacles.

profil de la route (avec une transformée de *Hough*). Une fois que les points 3-D appartenant à la route sont étiquetés, ceux restant sont étiquetés comme des obstacles. De la même manière, Nakai *et al.* [NTH+07] estiment le tangage et la hauteur relative des caméras également par une transformée de Hough, tout en se passant de la représentation intermédiaire sous forme de v-disparité. Le grand avantage de ces approches, qui utilisent la redondance d'information, est qu'elles sont robustes au bruit. Mais toutes ces méthodes se basent sur l'hypothèse que le monde est plan. Cette hypothèse est acceptable lorsque l'environnement est maîtrisé, mais rarement en extérieur, où la surface de la route subit toujours quelques légères variations. Et lorsque cela arrive, certains points de la route sont détectés comme des obstacles, ce qui pose un réel problème.

Pour tolérer des variations de la surface du sol, la littérature propose des approches basées sur des modèles de surfaces. La méthode proposée par Lemonde *et al.* [Lem05] reprend les histogrammes de la v-disparité, mais réalise la

segmentation indépendamment sur chaque ligne de la carte de v-disparité. Le fait d'ignorer le profil longitudinal de la route permet d'étendre la méthode au cas des routes non-planes. Mais cette extension se fait au détriment de la robustesse. En effet, si le long d'une ligne l'image, la route n'est pas visible, alors son identification, et donc la segmentation n'est pas possible. Quant à Labayrade *et al.* [LAT02] et Hautière *et al.* [HLPA04], ils améliorent la méthode de v-disparité avec un modèle de route composé par une succession de plans. Nedevschi *et al.* [NDM+07] utilisent un profil courbe [NSG+04] pour modéliser la route. Afin de faciliter la mise en correspondance de leur modèle avec les données, ils exécutent une étape préliminaire d'extraction du marquage routier, afin d'identifier par leur apparence (et non par la géométrie) les points appartenant à la route.

Le dernier type d'approche s'attache seulement à modéliser l'orientation locale de la surface de la route. Parmi elles, celles introduites par Robert *et al.* [RH94] et Qian Yu *et al.* [YAW03] calculent, à partir de la carte de disparité, la normale en chaque point reconstruit. De cet ensemble des normales, est dégagée une direction privilégiée, choisie pour représenter l'orientation du sol. Tous les points ayant une orientation trop différente de celle du sol caractérisent la présence d'obstacles. Mais l'estimation de l'orientation en chaque point à partir de la carte de disparité est une opération délicate. En effet, le bruit et les erreurs d'estimation de la disparité rendent le calcul de sa dérivée très imprécis. Plutôt que d'estimer la normale à partir de la carte de disparité, Williamson [Wil98] préfère utiliser directement les images acquises. Grâce à de multiples rectifications, l'approche proposée par Williamson dégage une orientation dominante. En effet, la puissance de calcul disponible limite à deux le nombre d'orientations évaluées. L'estimation de l'orientation à l'aide de cette technique donne donc un résultat très imprécis, mais suffisant pour identifier les surfaces praticables par un véhicule.

Agrégation

Une fois que la surface de la route est identifiée et segmentée, il reste à traiter l'information concernant les obstacles. En effet, la nature des données permettant de représenter ces derniers est à choisir en fonction de l'application visée. Pour du guidage de véhicule, une carte de l'environnement sera alors la représentation la plus appropriée [BBF98, Wil98, TMB04, SAH07, JHG+03, YAW03]. Ce type de carte fait apparaître les zones praticables par le véhicule porteur.

Mais plus généralement, les applications automobiles nécessitent des données plus synthétiques sur les obstacles, comme leur position, vitesse et éventuellement taille. Par la suite, nous utiliserons indifféremment les termes « obstacle » et « cible ». Dans ce cas, il est nécessaire de traiter la carte de disparité afin d'en extraire les cibles (voir la figure 5.4). Ce processus, que nous appelons agrégation, permet d'établir une boite englobant les points d'un même obstacle afin de constituer une cible. La plupart du temps, la segmentation consiste à regrouper les points voisins entre eux.

La notion de proximité des points diffère suivant les approches. La représentation euclidienne n'est pas très adaptée, et il lui est souvent préférée la carte de disparité, voire la u-disparité. De nombreux auteurs [LAT02, Lem05, HLPA04, NDM+07] utilisent les composantes connexes de la u-disparité pour agréger les points d'un même obstacle.

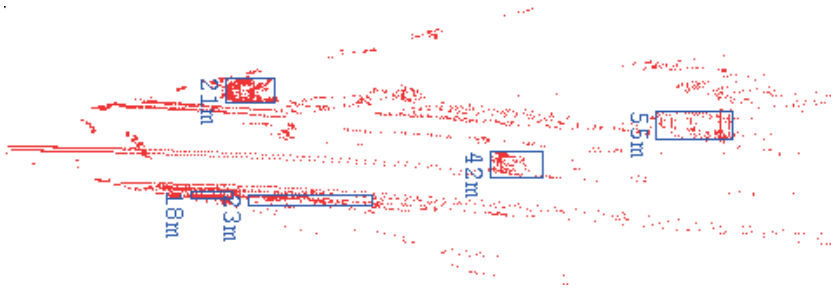


FIG. 5.4: L'opération d'agrégation des points 3-D (ici en vue de dessus) permet d'obtenir des cibles, ici représentées par des boites englobantes [NDF+04].

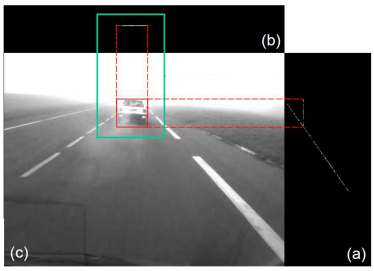


FIG. 5.5: En combinant *v-disparité* et *u-disparité*, Hautière *et al.* [HLPA04] reconstruisent la boite englobante de l'obstacle.

Nakai *et al.* prennent le problème dans l'autre sens. Au lieu de reconstruire l'environnement pour ensuite y chercher des obstacles, ils génèrent des hypothèses d'obstacles pour ensuite les vérifier. Ce type d'approche se dispense d'étape d'agrégation, puisque les cibles sont initialement générées.

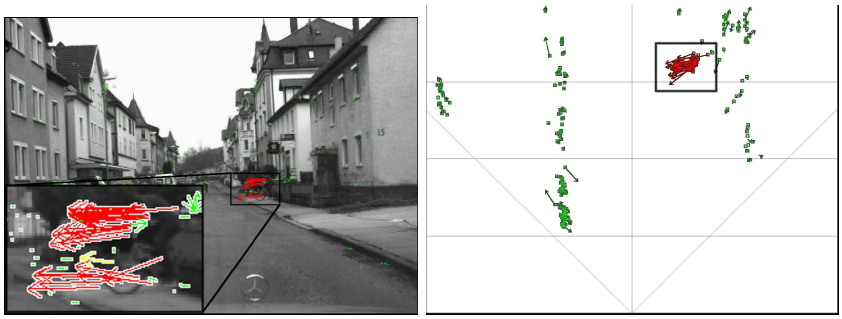
5.1.3 Mouvement

Le troisième type d'approche définit les objets en mouvement comme des obstacles (voir la figure 5.6). Rares sont les approches qui se basent sur cette définition, car beaucoup d'applications automobiles ne peuvent se contenter de considérer uniquement les obstacles en mouvement. Mais le mouvement constitue une caractéristique très discriminante, et dans certaines applications comme le suivi de véhicules, il peut constituer un critère de détection très pertinent.

Toute les approches décrites dans cette partie [Hei02, FH02, RFG07, HUW06, TM04, AKI05] basent la segmentation sur la contrainte qui lie le flux optique et la profondeur [WD86] (voir la figure 5.6).

Heinrich *et al.* [Hei02] segmentent les objets en mouvements en comparant le flux optique mesuré à celui prédit par le couplage du mouvement propre et de la carte de disparité. C'est un première manière de faire. Franke *et al.* [FH02] et Rabe *et al.* [RFG07] améliorent la précision et le taux de faux positif en filtrant les détections sur plusieurs dates à l'aide du filtrage de *kalman*. L'odométrie du véhicule fournie par les capteurs embarqués semble d'une précision insuffisante [FH02] et semble de meilleur qualité lorsqu'il est estimé par des technique de vision [RFG07]. Hu *et al.* [HUW06] calculent les déplacements directement dans la carte de u-disparité.

Une deuxième manière consiste vérifier le respect de la contrainte sur les cartes de disparité plutôt que sur les flux optiques. Pour se faire, Agrawal *et al.*



(a) (b)

FIG. 5.6: Franke *et al.* [FH02].

[AKI05] proposent de mettre à jour la carte de disparité observée à la date $t-1$ par le mouvement propre. Puis la comparaison de la carte prédite et de celle estimée permet de dégager des objets qui suivent un mouvement différent.

Finalement, la méthode proposée par Talukder *et al.* [TM04] reprend simultanément les deux manières précédentes d'utiliser la contrainte.

5.2 Le tamis à obstacles

Dans cette partie, nous présentons une nouvelle approche, volontairement simple, permettant d'identifier et de localiser les obstacles potentiels afin de fournir des cibles au processus de mesure de vitesse. Notre but est de fournir de nombreux « candidats », sachant que les faux-positifs seront facilement éliminés par le processus de suivi. Notre approche se base sur une définition géométrique des obstacles. Comme la majorité des approches de ce type, elle s'articule en trois étapes :

1. reconstruction 3-D,
2. segmentation de la route et
3. agrégation des cibles.

Les étapes 1 et 2 sont effectuées conjointement au moyen de la mise en correspondance stéréoscopique « améliorée » (cf chapitre 4.2). En effet, ce processus fournit une reconstruction 3-D de la scène sous la forme d'une carte de disparité, ainsi que l'orientation locale de la surface en chaque pixel. Comme ce processus est largement étudié dans le chapitre précédent, nous ne le détaillerons pas à nouveau dans cette partie. Rappelons seulement que nous utilisons cette information d'orientation pour discriminer les zones praticables de celles qui constituent des obstacles. Cette information à elle seule peut suffire à l'élaboration d'une carte des zones praticables, suffisante dans le cadre d'applications comme le guidage de véhicules autonomes [MR01]. Mais les applications de type « suivi de véhicule » (parfois appelées « attache immatérielle » ou ACC[1]) nécessitent l'identification, la localisation et le suivi des véhicules cibles.

[1] ACC pour Adaptative Cruise Control, ou Automatic Cruise Control

Le traitement que nous détaillons dans cette partie consiste à agglomérer les points 3-D reconstruits de manière à regrouper tout ceux qui appartiennent à un même obstacle. Pour effectuer cette étape d'agglomération (ou d'agrégation), nous appliquons une méthode que nous avons appelée le « tamis à obstacles ». L'idée fondatrice est de retenir les agglomérats de points de taille suffisante, et de rejeter les plus petits. En première partie, nous expliquons le principe du tamis à obstacles. En deuxième partie, nous détaillons une implantation optimisée de cette approche. Pour conclure, nous illustrons l'efficacité de la méthode sur quelques séquences réelles.

5.2.1 Principe du tamis à obstacles

Dans le chapitre précédent (4.2, page 87) nous avons expliqué comment obtenir une carte de disparité qui donne également la classification de chaque pixel comme « obstacle » ou « espace libre » (voir la figure 4.23). En retirant l'espace libre de cette dernière, il reste une carte ne mentionnant que les obstacles, que nous appellerons carte d'obstacles. Pour obtenir les cibles nécessaires au processus de mesure de vitesse, il reste à agréger ensemble les pixels qui appartiennent à un même obstacles. La nature éparse et bruitée des données rend ce processus délicat. En effet, comme le montre la figure 5.7, les obstacles ne sont souvent que partiellement reconstruits. Dans ce cas, il est difficile de relier entre elles les différentes parties d'un même obstacle. Ce problème est généralement résolu par l'utilisation de la cohérence spatiale. Par exemple, la méthode très populaire de *u-disparité* [LAT02] utilise la répétition de certaines valeurs de disparité le long de l'axe vertical puis horizontal de l'image pour identifier la présence d'un obstacle. De la même manière, nous souhaitons utiliser cette cohérence spatiale, mais simultanément suivant les trois dimensions. Cela se résume par la recherche d'un volume contenant une densité de points assez conséquente pour être considérée comme un obstacle.

Cette recherche est effectuée en deux étapes. Le première consiste à éclater la carte d'obstacles en une série de calques (voir la figure 5.7). Chaque calque peut être représenté par une image binaire, faisant apparaître tous les pixels d'une disparité donnée. On peut imaginer un calque comme une tranche du monde. En pratique, ça n'est pas une disparité unique, mais une plage de valeurs de disparité qui sont considérées.

Pour chaque calque (voir la figure 5.8), l'exercice consiste à trouver un, ou plusieurs volumes, contenant une densité suffisante d'obstacles. Tous les volumes possibles sont passés en revue, et ceux étant suffisamment « remplis » par des obstacles sont retenus.

Cette approche repose sur deux hypothèses :

1. un obstacle doit être mesuré à disparité constante, c'est à dire qu'il fait face aux caméras
2. et il doit recouvrir un nombre suffisant de pixels pour se démarquer du bruit.

Nous reprenons ces mêmes hypothèses.

Implantation optimisée

Dans cette partie, nous présentons une implantation optimisée de la méthode du tamis à obstacles. Comme le processus de mise en correspondance stéréosco-

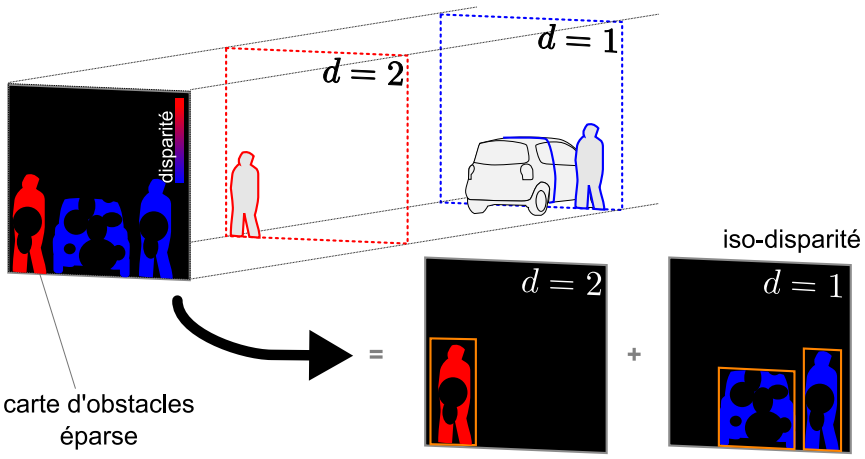


FIG. 5.7: Décomposition de la carte d'obstacles en une série de calques d'iso-disparité.

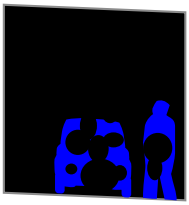


FIG. 5.8: Calque d'iso-disparité.

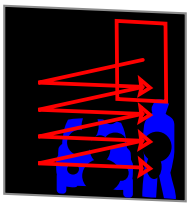


FIG. 5.9: Recherche d'un obstacle de taille suffisante.

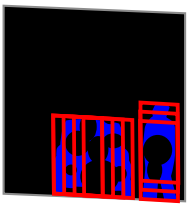
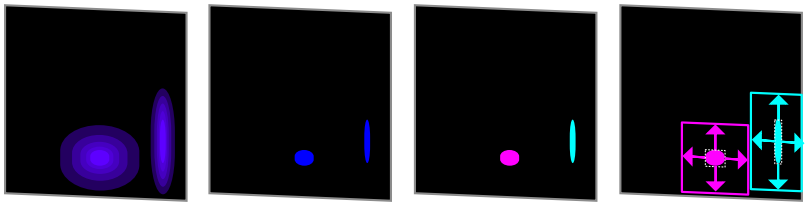


FIG. 5.10: Obstacles retenus.

pique, celui ci peut également être implanté de manière optimisée. L'optimisation réside essentiellement dans le fait de traiter tous les pixels en une seule fois. Dans cette partie, nous présentons une implémentation basée sur des fonctions de la bibliothèque *OpenCV* [Int01].



(a) (b) (c) (d)

FIG. 5.11: Les quatre étapes de l'implantation sont (a) le calcul du ratio d'obstacles, (b) le seuillage, (c) la recherche des composantes connexes et (d) la dilatation.

L'implantation s'effectue alors en 4 étapes.

1. Calculer la proportion de pixels étiquetés comme « obstacle » dans une fenêtre glissante constitue la première étape. Ce calcul peut être réalisé par un calcul de moyenne sur une fenêtre glissante (illustré par la figure 5.11(a)). Ce calcul ne dépend pas de la taille de la fenêtre, mais uniquement de la taille de l'image (se référer à l'optimisation des fenêtres glissantes, décrite par la figure 4.6 p. 80). A ce stade, la proportion de pixels blancs est mémorisée dans le pixel au centre de chaque fenêtre (voir la figure 5.11(a)). Par la suite, il faudra sélectionner les fenêtres dont le nombre de pixels blancs (étiquetés obstacles) dépasse la proportion fixée.
2. Un seuillage de ces rapports obstacle/non-obstacle donne une nouvelle image binaire (comme celle illustrée par la figure 5.11(b)). Les fenêtres (représentées par leur pixel central) sont ainsi classées en deux catégories, celles dont la proportion de pixels obstacles est supérieure au seuil fixé, et les autres.
3. Si l'objet détecté dépasse en taille celle de la fenêtre de détection, alors l'algorithme retourne plusieurs fenêtres qui se jouxtent (voir la figure 5.10). Dans ce cas, nous ne souhaitons retenir qu'une seule fenêtre qui les englobe toutes. Pour cela, nous relions toutes les fenêtres dont les centres sont connectés. L'ensemble des composantes connexes donne une région (sur la figure 5.11(c), tous les pixels d'une même couleur représentent une région caractérisant une même cible).
4. Une opération de dilatation simplifiée (voir la figure 5.11(d)) permet de recouvrer la fenêtre associée à chaque région. L'opération de dilatation n'est pas une opération de dilatation au sens morphologique du terme. Car, le passage de la fenêtre englobant la région de pixels (boites en pointillés blancs sur la figure 5.11(d)) à la vraie taille de la fenêtre (en trait plein) ne nécessite que de l'agrandir par une constante.

Pour limiter le nombre de parcours de l'image, les opérations de recherche de composantes connexes et de dilatation sont effectuées en une seule passe. Dans le cas d'une application de suivi de véhicules, la segmentation des obstacles se restreint à la détection de véhicules (ou autres obstacles de taille suffisante) se trouvant sur l'axe du véhicule porteur.

L'algorithme 4 correspond à notre implantation.

5.3 Résultats

Le but de cette partie est de présenter les résultats du tamis à obstacle. Malheureusement, le temps nous a manqué pour effectuer une évaluation quantitative, même sur des séquences de synthèse. Nous ne présenterons ici que quelques résultats préliminaires, obtenus sur des séquences réelles (voir les figures 5.12, 5.13 et 5.14).

Les résultats présentés ici ont été obtenus sur des séquences réalisées à l'aide de deux caméras *Dragonfly* de *Point Grey Research Research*. Ces appareils de prise de vue ont une distance focale de 6 mm (objectif *Pentax*), avec une capteur CCD de 640×480 pixels de résolution répartis sur une surface de $1/3''$. La distorsion est négligeable. Les caméras sont placées à 40 cm l'une de l'autre, regardant toutes les deux vers l'avant du véhicule, et strictement d'aplomb.

Algorithme 4 : Algorithme du « tamis » à obstacles. Les fonctions commençant par *cv* sont fournies par la bibliothèque *OpenCV* [Int01].

```
liste d'obstacles ← aucun;
identifiant obstacle ← 0 ;
identifiant unique ← identifiant obstacle + 1 ;
pour d ← d_min to d_max faire
    [d_1, d_2] ← intervale de disparité( d);
    tampon ← cvInRange( carte de disparité, d_1, d_2);
    taille apparente minimale ← TailleMinimale( d);
    tampon ← cvSmooth( tampon, CV_GAUSSIAN, taille apparente
    minimale);
    tampon ← cvThreshold( tampon, ratio de remplissage de la
    fenêtre, identifiant obstacle, CV_THRESH_BINARY);
    pour tous les pixel ← tampon faire
        si valeur( pixel) = identifiant obstacle alors
            fenêtre_englobante ← ( coordonnées( pixel), coordonnées(
            pixel) ) ;
            pour tous les p_i ← ComposantesConnexes(pixel) faire
                valeur( p_i) ← identifiant unique ;
                fenêtre_englobante ←
                NouvelleBoiteEnglobante(fenêtre_englobante,
                coordonnées(p_i)) ;
            liste d'obstacles ← (liste d'obstacles, dilateBoiteEnglobante(
            fenêtre_englobante, taille apparente minimale) ) ;
            identifiant unique ← identifiant unique + 1 ;
```

Sur ces trois séquences, le processus de segmentation d'obstacles est limité à l'axe du véhicule, sur la largeur d'une voie de circulation. De cette manière, seul le véhicule cible et ses abords immédiats sont détectés. Les zones détectées par le processus sont affichées en rouge sombre sur les figures 5.12, 5.13 et 5.14. Puis le processus de mesure de vitesse valide les détections en utilisant la cohérence temporelle. Les cibles effectivement suivies sont affichées en vert.

Nous pouvons remarquer que les véhicules sont souvent détectés à plusieurs profondeurs différentes (figure 5.12 et 5.14). Cet artefact est la conséquence d'une superposition des calques. En effet, rappelons que lors de l'éclatement de la carte de disparité en une série de calques, ces derniers représentent des tranches de disparités qui se superposent. Tout comme il l'a été fait dans les deux autres direction, il est possible de fusionner les multiples détections le long de l'axe longitudinal.

5.4 Conclusion

Nous avons présenté une méthode de segmentation d'obstacles, qui, en se basant sur une carte de disparité déjà segmentée, fournit les cibles nécessaires au processus de mesure de vitesse. Cette approche utilise des critères géométriques pour détecter et segmenter les obstacles. Comme pour beaucoup d'approches de ce type, nous avons voulu tirer profit de la cohérence spatiale pour résister au

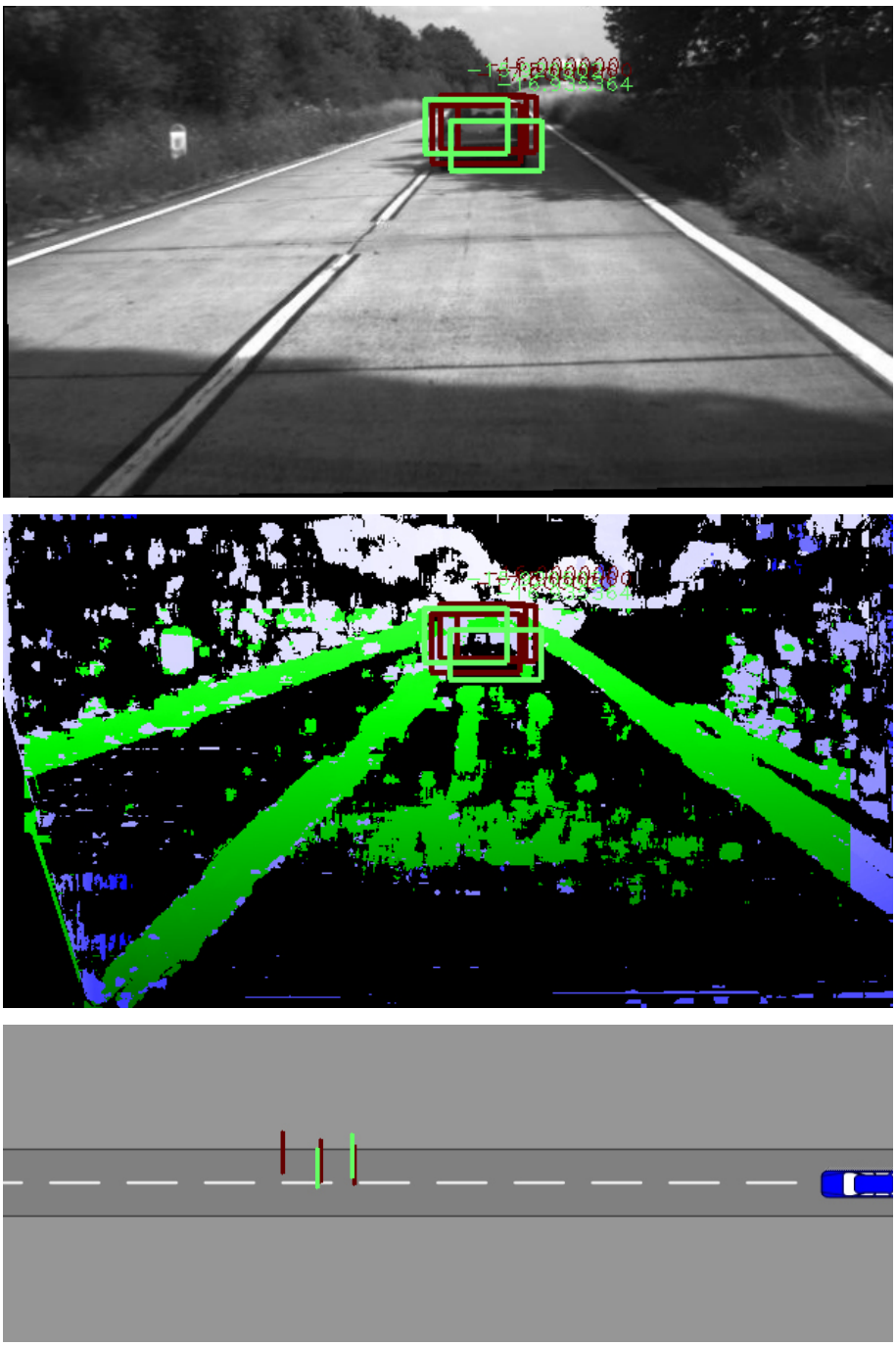


FIG. 5.12: Les rectangles rouges sombres sont le résultat du tamis à obstacles, affichés sur l'image (en haut), sur la carte de disparité segmentée (au milieu) et sur une vue de dessus (en bas). Les boîtes vertes représentent les obstacles qui n'ont pas été éliminés par l'étape ultérieure de mesure de vitesse (voir le chapitre suivant). Pour une meilleure lisibilité de cet exemple, la détection a été restreinte à la voie de circulation.

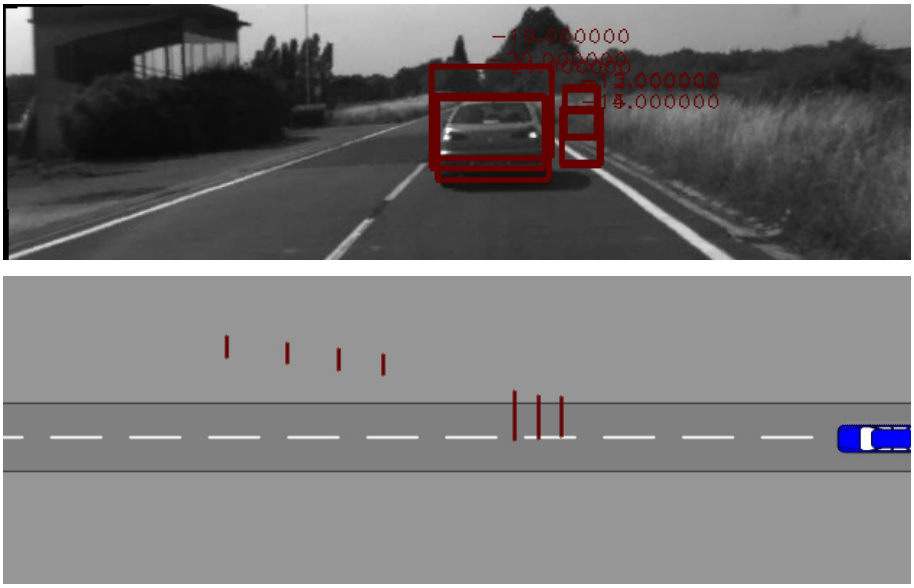


Fig. 5.13: Sont détectés le véhicule précédent le porteur, ainsi qu'une partie des herbes hautes bordant la chaussée.

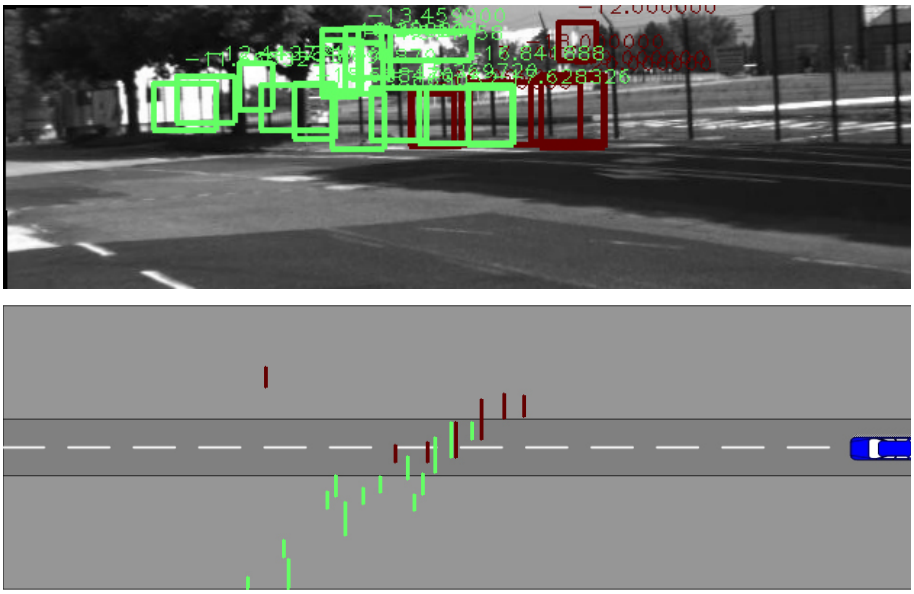


Fig. 5.14: La clôture est détectée (en rouge) sur l'axe du véhicule. Le véhicule tournant vers la droite, les obstacles détectés sont suivis (en vert) lors de leur déplacement (relatif) sur la gauche du champ de vision.

mieux au bruit de mesures.

Cette brique fonctionnelle peut être améliorée. Notamment en fusionnant les multiples détections d'un même obstacle. Néanmoins, l'implantation actuelle rempli l'objectif désiré, à savoir : fournir des candidats au processus de mesure de mouvement. Car les faux positifs ne sont pas une gêne dans la mesure où ils seront filtrés par le processus de mesure de vitesse.

Chapitre 6

Mesure du déplacement 3-D des cibles pour une application automobile

La prise de décision par le véhicule intelligent passe nécessairement par la perception de l'environnement. L'estimation des position et vitesse des obstacles environnants est l'étape clef de la compréhension de l'entourage du véhicule. Dans une scène routière, les obstacles sont en perpétuelle évolution, puisque des véhicules s'y déplacent en même temps que l'observateur. L'information de mouvement est donc aussi importante que l'information de position. Dans le cas de certaines applications, comme l'*ACC*, le système utilise essentiellement l'information de vitesse relative à la cible, qui constitue donc un paramètre primordial.

Grâce à l'effet *Doppler*, le RADAR et le LIDAR fournissent une mesure directe de vitesse relative atteignant la précision nécessaire aux applications tel que l'*ACC*. Les technologies basées sur la vision (sans lumière structurée) ne fournissent pas cette mesure directe de vitesse relative. Une alternative consiste à détecter la cible à des instants successifs et, par différence finie, d'en déduire la vitesse. La mesure du déplacement nécessite la mise en œuvre d'algorithmes spécifiques, différents des algorithmes fournissant l'information de position (voir la figure 6.1).

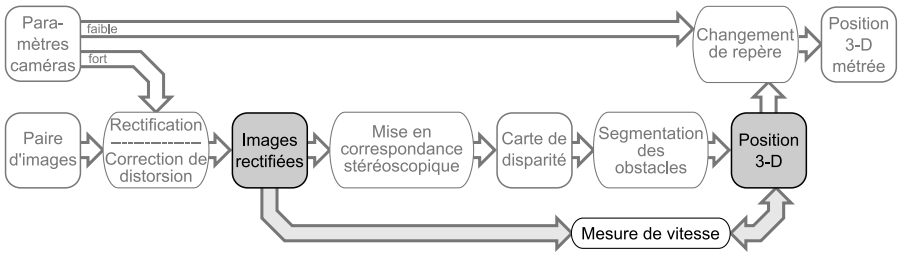


Fig. 6.1: Le flux d'images stéréoscopiques permet de mesurer le déplacement 3-D des obstacles.

Pour preuve, la nature a également fait ce choix. Car, aussi bien chez l'homme que chez l'animal, la capacité de percevoir le mouvement fait intervenir des processus spécifiques. Dès les années 1970, l'étude du singe-hibou et du macaque [AK71] a montré que le cortex visuel (préstrié) est constitué d'aires cérébrales séparées. A l'aide d'électrodes, des chercheurs ont pu enregistrer l'activité électrique de cellules du cortex préstrié de singes, lesquels étaient soumis à différents stimuli (couleurs, points lumineux en mouvement, barres d'orientation variable). Les cellules de l'aire MT (V5 chez l'homme) sont sensibles aux mouvements, et pour la plupart, spécifiquement à certaines directions, et aucune n'est sensible à la couleur des cibles en mouvement.

Dans ce chapitre nous présentons une nouvelle approche de suivi, basée sur l'algorithme standard de Lucas & Kanade [LK81] (abrégé LK). Le but de notre approche est d'étendre la mesure de déplacement au cas stéréoscopique, afin d'obtenir une mesure 3-D du mouvement.

En première partie, nous présentons un état de l'art reprenant les méthodes de mesure de mouvements 3-D. En deuxième partie, nous commençons par décrire en profondeur l'algorithme LK, qui propose une formulation permettant de suivre un point T désigné entre les dates t et $t+1$. A ce propos, nous détaillons la formulation standard, ses nombreuses extensions ainsi que l'implantation fournie dans *OpenCV* [Int01]. Puis, nous présentons notre nouvelle méthode de mesure de mouvement 3-D. Cette nouvelle approche est une extension de l'algorithme Lucas & Kanade, permettant de traiter une séquence de paires d'images rectifiées, afin de mesurer les déplacements 3-D. L'adaptation s'articule en quatre points. Une évaluation quantitative et qualitative de ces adaptations est présentée.

6.1 État de l'art de la mesure de déplacement 3-D par vision

Dans cette partie, nous présentons les principales méthodes de mesure de déplacement 3-D basées sur la vision. Le domaine de l'estimation du mouvement 3-D a été très largement étudié, et considéré de différentes façons. Nous catégorisons les différentes approches selon les hypothèses faites sur la scène et la nature du capteur employé, et distinguons ainsi les quatre familles suivantes :

6.1.1 un système monoculaire et mouvements 3-D rigides (*Structure-from-motion*),

6.1.2 un système monoculaire et mouvements 3-D non rigides,

6.1.3 un système multi-caméras et mouvements 3-D rigides (*Stereo-Motion*),

6.1.4 un système multi-caméras et mouvements 3-D non rigides (flux de scène).

Nous aborderons également une cinquième famille de méthodes dont la problématique, légèrement différente, consiste à reconstruire l'information 3-D à partir du mouvement, mais sans notion de déplacement 3-D. Nous appelons cette cinquième famille :

6.1.5 carte de profondeur dynamique.

Remarque : Le mouvement rigide est caractérisé par la rotation R et la translation $\mathbf{t}$ que relient deux ensembles de points (ou surfaces) 3-D. Dans la pratique, le mouvement est généralement effectué par la caméra, et l'ensemble

des points reste immobile. Mais le problème reste le même. Par opposition, les mouvements non rigides caractérisent un ensemble de points ayant des mouvements distincts.

6.1.1 Systèmes monoculaires et mouvements 3-D rigides (*Structure-from-motion*)

La première famille, appelée *Structure-from-motion* (structure à partir du mouvement), regroupe les méthodes s'attachant à reconstruire le mouvement relatif en même temps que la structure de la scène. Ullman [Ull79] suppose la scène statique (rigide). Costeira et Kanade [CK98] et Zhang *et al.* [ZF92a] supposent une scène rigide par morceaux. Seuls Avidan et Shashua [AS99] étendent cette technique à un mouvement non-rigide particulier : un mouvement linéaire (voir la figure 6.2(a)).

(a) (b)

Fig. 6.2: A gauche : Avidan et Shashua [AS99] montrent que sous l'hypothèse d'un mouvement linéaire, le déplacement d'une cible peut être reconstruit avec un système monoculaire. A droite : Lou *et al.* [LTH+05] proposent un modèle en « fil de fer » 3-D spécifiquement adapté aux véhicules à 4 roues.

6.1.2 Systèmes monoculaires et mouvements 3-D non rigides

En tirant avantage de connaissances *a priori*, ou en modélisant directement des hypothèses sur la scène, un système monoculaire peut fournir l'estimation de mouvements non rigides. Stein *et al.* [SMS03] caractérisent chaque cible par une fenêtre englobante, et se servent des variations de taille apparente pour estimer les variations de profondeur. Metaxas *et al.* [MT93] et Pentland *et al.* [PH91] utilisent un modèle déformable prédéfini et en estiment les paramètres en vue de reconstruire le mouvement 3-D. Koller *et al.* [KDNT92] et Lou *et al.* [LTH+05] (voir la figure 6.2(b)) proposent un modèle en « fil de fer » 3-D spécifiquement adapté aux véhicules à 4 roues. Les 12 paramètres permettant au modèle de s'ajuster au véhicule sont estimés en même temps que la pose. Ferryman *et al.* [FWM98] étendent ce modèle pour prendre compte la texture de la cible. En tirant ainsi partie de l'information photométrique, la localisation de la cible est ainsi plus précise. Le travail antérieur de Ullman [Ull84] suppose que le meilleur

modèle est celui qui minimise l'écart par rapport à modèle rigide. Récemment, des chercheurs ont découvert que, dans la pratique, beaucoup d'objets non rigides subissent des déformations structurées. Le modèle de l'objet peut alors être considéré comme la combinaison pondérée de plusieurs formes de bases rigides. Pour le suivi de visage, Gokturk *et al.* [GBG01], dans une étape préliminaire, construisent l'espace modélisant toutes les déformations faciales en appliquant une analyse en composantes principales (*PCA*) sur des données stéréoscopiques. Le résultat est une base permettant de modéliser approximativement tout visage par une combinaison linéaire des formes de base. La complexité réduite du modèle obtenu par PCA permet d'estimer simultanément la pose et les paramètres de forme, dans une seule image. Torresani *et al.* [THB03] utilisent l'algorithme *EM* (*Expectation-Maximization*) pour estimer simultanément la forme 3-D et le mouvement. Sous la condition d'un modèle de projection orthographique, cet algorithme apprend les paramètres de la gaussienne qui modélise les déformations du modèle. Cette gaussienne permet également de compléter efficacement les données manquantes. Xiao *et al.* [XCK04], en définissant un ensemble de contraintes, proposent d'éliminer toute ambiguïté quant à la reconstruction du mouvement et de la forme de l'objet suivi. Cela permet, sous l'hypothèse d'un modèle de projection perspective faible, d'exprimer la solution directe. Nécessitant plusieurs images, les méthodes de Torresani *et al.* [THB03] et Xiao *et al.* [XCK04] fonctionnent hors-ligne. D'autre part, l'hypothèse d'une combinaison linéaire de formes de base ne suffit pas à modéliser des déformations fines et/ou complexes.

Pour plus de détails sur le suivi monoculaire basé modèle, se référer à l'ouvrage de Lepetit et Fua [LF05].

6.1.3 Systèmes multi-caméra pour la mesure de mouvements 3-D rigides (*Stereo-Motion*)

Cette famille d'approches, appelée *stereo-motion*, regroupe les travaux qui fusionnent l'information stéréoscopique et celle du mouvement 2-D pour mesurer le mouvement 3-D (voir la figure 6.3). La plupart des approches, comme celles proposées par Waxman *et al.* [WD86], Young *et al.* [YC90], Zhang *et al.* [ZF92b] et Shi *et al.* [SSP94], supposent la rigidité de la scène et/ou du mouvement. Liao *et al.* [LAA97] et de Malassiotis *et al.* [MS97] font exception et expliquent comment suivre des déplacements non-rigide, Liao *et al.* [LAA97] utilisent algorithme de relaxation, alors que Malassiotis *et al.* [MS97] étendent le modèle déformable de Metaxas *et al.* [MT93]. La première approche ne peut pas fournir de reconstruction dense du champ de déplacement. La seconde nécessite un connaissance *a priori* sur la scène, formalisée par le modèle déformable de la cible.

Dans un contexte automobile, l'estimation du mouvement rigide sert essentiellement à déterminer le mouvement propre du véhicule (*ego-motion*). Cette information peut être fournie par les capteurs odométriques, mais la précision trop faible des capteurs de série favorise l'utilisation de techniques de visions. L'estimation du mouvement ne peut se faire que sous l'hypothèse d'une scène statique. En pratique, une étape préliminaire permet d'éliminer les objets en mouvement, Pour éliminer de tels objets (considérés comme des *outliers* dans ce problème), Talukder *et al.* [TM04] utilisent la technique d'estimation itérative de l'erreur aux moindres carrés (*least mean-square error*). Agrawal *et al.* [AKI05]

utilisent l'algorithme de RANSAC pour éliminer les objets en mouvement. Rabe *et al.* [RFG07] identifient les points statiques à l'aide de la prédiction faite par un filtre de Kalman. Ce filtrage permet également d'améliorer la précision de la mesure du mouvement propre.

6.1.4 Systèmes multi-caméra et mouvements 3-D non rigides (flux de scène)

Vedula *et al.* [VBR+99] introduisent le concept de flux de scène (voir la figure 6.3), l'équivalent 3-D du flux optique. Pour reconstruire le flux de scène (*scene-flow* en anglais), nous distinguons deux catégories d'approches, suivant que l'estimation du mouvement et de la 3-D soient effectués séparément ou simultanément. Celles qui effectuent les deux processus séparément constituent la première catégorie. Les méthodes regroupant le mouvement et la stéréoscopie dans le même formalisme constituent la deuxième catégorie.

Calcul séparé de la reconstructions 3-D et du mouvement

Pour reconstruire le flux de scène (voir la figure 6.3), Vedula *et al.* [VBR+99] présentent un algorithme linéaire utilisant le flux optique estimé sur les flux vidéos de l'ensemble des caméras du système. Étant donné le flux de scène et la structure initiale de la scène, les variations de la scène peuvent être reconstruite.

Dans un système complet de détection et de suivi d'obstacle routier, la reconstruction 3-D est généralement calculée à chaque instant [BBFN00, SZB99, NDF+04]. Comme l'estimation de la 3-D est effectué, il ne reste plus qu'a mettre en correspondance les multiples détections au cours du temps. Bertozzi *et al.* [BBFN00] associent à l'aide du flux optique provenant de l'une des deux vues. Sobottka *et al.* [SZB99] utilise la reconstruction 3-D de la scène (carte de profondeur) pour en extraire une silhouette 3-D de la cible (un véhicule), puis de rechercher cette silhouette dans les reconstructions suivantes. Miyahara *et al.* [MSKI06] fusionnent l'ensemble des détections par une regression linéaire. Nedevschi *et al.* [NDF+04], Dang *et al.* citedang :02, Franke *et al.* [FH02, RFG07], Leibe *et al.* [LCCG07] et Agrawal *et al.* [AKI05] effectuent la mise correspondance au cours du temps à l'aide du très populaire filtrage de *Kalman*. Le filtre de Kalman permet de modéliser la dynamique des cibles. Leibe *et al.* [LCCG07] tient également compte d'un critère de cohérence photométrique.

Unification de la stéréoscopie et du flux optiques

Zhang et Kambhamettu [ZK00] reformule le problème de l'estimation du flux de scène sous la forme d'une énergie à minimiser. Le flux de scène est obtenu par la recherche du mouvement affine qui correspond à la déformation de l'image, sous la contrainte globale de surface lisse. Dans un deuxième article [ZK01], ils améliorent cette dernière approche affin de préserver les discontinuités de surface et gérer les parties masquées. Lorsqu'une partie de la surface est masquée, l'information de profondeur lui étant associée est ignorée et le flux de scène ne dépend plus que du flux optique. Pons *et al.* [PKF07] contournent l'estimation du flux optique. La surface et le flux de scène sont modélisés par une fonction d'énergie à minimiser. La fonction se base des critères globaux et locaux à propos de la reprojection de la surface de la scène dans les images. Le

problème de visibilité n'est pas géré. Comme pour les autres approches fondées sur une fonction d'énergie, Zhang et Kambhamettu [ZK00, ZK01] pondèrent les termes de la régularisation à l'avance. Devernay *et al.* [DMG06] reconstruisent le flux de scène directement à partir des flux vidéos. Après une étape préliminaire de reconstruction de la surface 3-D de la scène, ils effectuent une suivi des points suffisamment texturés. La mise en correspondance entre l'ensemble des prises de vues consécutives est effectuée à l'aide d'une extension de l'algorithme de Lucas & Kanade, dans laquelle les points sont représentés par un élément de surface localement plan.

6.1.5 Carte de profondeur dynamique

Quelques approches s'attachent à reconstruire une carte de profondeur variant au cours du temps en utilisant des contraintes de déplacement. Le résultats n'est pas le flux de scène 3-D défini dans la partie précédente. Tao *et al.* [TSK01] supposent que la scène se découpe en un ensemble d'éléments localement plan. Les contraintes de déplacement sont intégrées pour prédire la profondeur de ces éléments dans l'observation suivante. Tous les éléments sont ensuite rassemblés sous l'hypothèse de cohérence photométrique. Davis *et al.* [DRR03] et Zhang *et al.* [ZCS03] proposent deux autres approches de même catégorie. Le but est de reconstruire une forme dans une scène dynamique en effectuant la mise en correspondance dans l'espace-temps, plutôt que dans l'espace uniquement. Le fait que plusieurs images sont nécessaires pour évaluer la cohérence spatio-temporelle en fait des méthodes hors-ligne. De plus, avec ces méthodes, des variations d'illumination permettent d'améliorer la précision de la mise en correspondance. Ces trois approches fournissent une carte de profondeur dynamique, mais ne déterminent en aucun cas une correspondance temporelle sur les points 3-D de la scène : ce n'est pas un calcul de déplacement 3-D.

6.2 Méthode proposée pour la mesure de mouvement par stéreo-vision

Comme l'illustre la figure 6.3, il est établi [WD86, SSP94, KK95, Mil97] que le flux optique et l'information stéréoscopique sont connectés. Pourtant, parmi les approches décrites dans l'état de l'art, toutes celles [ISK92, BBFN00, DHS02, NDF$^+$04, TM04, RFG07] utilisant ces deux principes le font de manière séparée. Aucune n'unifie ces deux aspect pour effectuer une mesure de vitesse 3-D directe. Pourtant, nous sommes convaincus que la meilleure approche possible doit unifier les deux formalismes. Une combinaison efficace de la stéréoscopie et de l'analyse du mouvement permettrait de surpasser les limitations de chacune de deux approches. La plupart des auteurs considèrent le flux optique et la 3-D de manière indépendante, pour les fusionner ensuite, ce que nous appelons « suivi non-contraint » (expliqué dans la partie 6.2.5).

Nous avons préféré réunir dans une même formulation l'estimation des flux optiques et la 3-D afin d'estimer directement le flux de scène (cf. 6.3). Nous avons basé nos travaux sur l'algorithme de Lucas & Kanade, très utilisé en vision pour sa rapidité et son efficacité [BFSB92]. Après une brève présentation de l'algorithme de Lucas & Kanade monoculaire (partie 6.2.1), nous présentons les quatre extensions permettant de l'étendre à un système stéréoscopique :

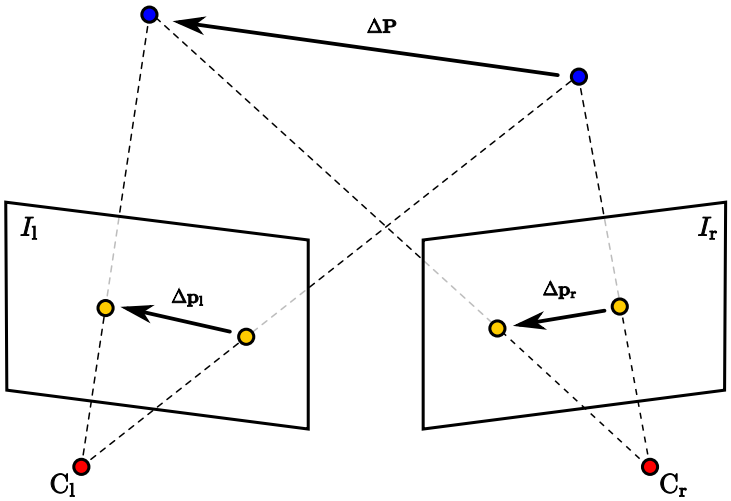


FIG. 6.3: Le flux de scène induit un flux optique dans chacune des deux images de la paire stéréoscopique.

1. gérer les deux vues simultanées (6.2.4)
2. utiliser des paramètres 3-D (6.2.5)
3. gérer les changements d'échelle des cibles (6.2.6)
4. utilisation de régions d'intérêt, plutôt qu'un suivi de points (6.2.7)

Des résultats quantitatifs sur des séquences synthétiques, ainsi que des résultats qualitatifs sur des séquences réelles sont présentés.

6.2.1 Suivi 2-D par l'algorithme de suivi de Lucas & Kanade

L'algorithme de Lucas & Kanade (abrégé *LK*) est une méthode très populaire en vision par ordinateur permettant de localiser un élément d'image T (de l'anglais *template*) défini dans une image I donnée (voir la figure 6.4). Initialement, cette méthode était utilisée pour faire de la reconstruction stéréoscopique en appariant des prises de vue simultanées. Mais le plus souvent, elle est utilisée pour mesurer un flux optique 2-D entre deux prises de vues consécutives. Dans ce dernier cas, l'élément d'image T est une sous-région extraite de l'image au temps t, et I est l'image acquise au temps $t+1$.

Le modèle

Cet algorithme se fonde sur un modèle d'apparence : chaque point est caractérisé par une fenêtre de pixels. Le point recherché est caractérisé par la région T, et le point candidat par la région $I(\mathsf{W}(\mathbf{x};\mathbf{p}))$. La similitude entre T et $I(\mathsf{W}(\mathbf{x};\mathbf{p}))$ se mesure par la fonction de corrélation entre ces deux fenêtres. La région $I(\mathsf{W}(\mathbf{x};\mathbf{p}))$ la plus similaire à T est celle qui minimise la fonction d'erreur

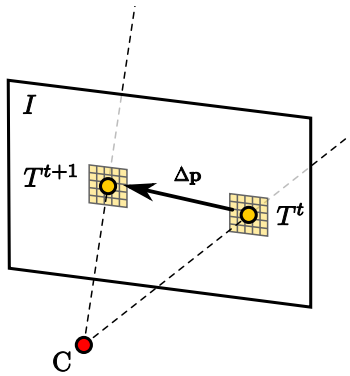


FIG. 6.4: L'algorithme de Lucas & Kanade maximise la similitude entre l'élément d'image T et une sous-région de l'image I. T^t et T^{t+1} caractérisent un point au temps t et $t+1$.

suivante :

$$\sum_{\mathbf{x}} \left[I(\mathsf{W}(\mathbf{x};\mathbf{p})) - T(\mathbf{x}) \right]^2 , \tag{6.1}$$

où $\mathbf{x}$ désigne une pixel de la région et $\mathbf{p}$ les paramètres à estimer, et W est la fonction de déformation (*warp* en anglais) paramétrée par $\mathbf{p}$.

Les paramètres d'états

Dans l'équation 6.1, la fonction W met en correspondance un point de texture $\mathbf{x}$ de l'élément T avec un point point de texture $\mathbf{x}$ de l'image I. Cette fonction est paramétrée par un vecteur $\mathbf{p}$. Dans le cas le plus simple, la fonction W modélise une translation 2-D. Dans ce cas, la fonction s'écrit :

$$\mathsf{W}(\mathbf{x};\mathbf{p}) = +\mathbf{p} = \begin{pmatrix} x + p_1 \\ y + p_2 \end{pmatrix} , \tag{6.2}$$

où $\mathbf{x}$ désigne un point de la texture et $\mathbf{p}$ est le vecteur de translation. Pour les points $\mathsf{W}(\mathbf{x} + vecteurp)$ correspondant à des coordonnées sous-pixeliques, une interpolation est utilisée.

Mise à jour itérative des paramètres d'état

L'algorithme de Lucas & Kanade est une méthode itérative cherchant un $\Delta\mathbf{p}$ dans le but de minimiser :

$$\sum_{\mathbf{x}} \left[I(\mathsf{W}(\mathbf{x};\mathbf{p} + \Delta\mathbf{p})) - T(\mathbf{x}) \right]^2 . \tag{6.3}$$

Cette approche suppose qu'une estimation initiale de $\mathbf{p}$ est connue. Habituellement, $\mathbf{p}$ est initialisé avec les précédentes valeurs observées, excepté pour la première observation où les points sont détectés par une méthode d'extraction de points d'intérêt [ST94]. Cet algorithme fait partie de la famille des approches

différentielles car il se base sur l'approximation au premier ordre d'un développement de Taylor de l'équation (6.1). L'équation (6.3) devient alors :

$$\sum_{\mathbf{x}} \left[I(\mathsf{W}(\mathbf{x};\mathbf{p})) + \nabla I \frac{\partial \mathsf{W}}{\partial \mathbf{p}} \Delta \mathbf{p} - T(\mathbf{x}) \right]^2, \tag{6.4}$$

où ∇I est le gradient de l'image I. La minimisation de l'équation 6.4 est un problème de moindres carrés et a une solution directe qui se dérive comme suit. Le vecteur des dérivées partielles de l'equation (6.4) par rapport aux coordonnées de $\Delta\mathbf{p}$ est donné par :

$$2 \sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathsf{W}}{\partial \mathbf{p}} \right]^{\mathsf{T}} \left[I(\mathsf{W}(\mathbf{x};\mathbf{p})) + \nabla I \frac{\partial \mathsf{W}}{\partial \mathbf{p}} \Delta \mathbf{p} - T(\mathbf{x}) \right]. \tag{6.5}$$

Les dérivées partielles sont nécessairement toutes nulles à l'optimum, ce qui conduit à l'expression suivante pour la mise à jour des paramètres :

$$\Delta \mathbf{p} = H^{-1} b, \tag{6.6}$$

où H est l'approximation par la méthode Gauss-Newton de la matrice Hessienne :

$$H = \sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathsf{W}}{\partial \mathbf{p}} \right]^{\mathsf{T}} \left[\nabla I \frac{\partial \mathsf{W}}{\partial \mathbf{p}} \right], \tag{6.7}$$

et b est :

$$b = \sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathsf{W}}{\partial \mathbf{p}} \right]^{\mathsf{T}} \left[T(\mathbf{x}) - I\left(\mathsf{W}\left(\mathbf{x};\mathbf{p}\right)\right) \right]. \tag{6.8}$$

puisque l'équation 6.3 est une forme quadratique définie positive, cet optimum est le minimum global de cette équation. $\mathbf{p} + \Delta\mathbf{p}$ devient le nouveau vecteur de paramètres $\mathbf{p}$. Cette mise à jour est réitérée jusqu'à ce que l'erreur obtenue soit tolérable (en dessous d'un seuil donné), ou que le nombre d'itérations dépasse un quota pré-défini. Dans le deuxième cas, on considère qu'il n'y a pas eu convergence et le résultat est invalidé.

6.2.2 Les nombreuses extensions de Lucas & Kanade

La simplicité et la rapidité de l'algorithme *LK* le rendent très populaire. Cependant, l'approche classique souffre de nombreuses limitations. Par exemple, elle n'accepte que de faibles déplacements et ne supporte pas les changement d'illumination. De très nombreux auteurs proposent des modifications de l'algorithme afin de l'étendre à des cas non supportés par la version originale. Dans cette partie nous présentons une dizaine de ces extensions. Il est intéressant de soulever que ces extensions ne sont généralement pas exclusives, et sont même souvent combinées.

Filtrage temporel

L'équation 6.6 permet de mettre à jour les paramètres d'état $\mathbf{p}$ uniquement en fonction de l'erreur constatée $[T(\mathbf{x}) - I\left(\mathsf{W}\left(\mathbf{x};\mathbf{p}\right)\right)]$. Si le comportement de la cible est connu, il est possible d'améliorer la fonction de mise à jour de façon à en tenir compte. Par exemple, le filtrage de Kalman [Sze94][HB96] peut améliorer le suivi d'un véhicule à la dynamique connue.

Multi-résolution

Dans l'algorithme standard, l'approximation au premier ordre (équation 6.4) faite pour résoudre la minimisation limite la mesure à des déplacements d'environ un pixel. Pour contourner cette limitation, les approches pyramidales commencent par diminuer la taille des deux images, jusqu'a ce que le déplacement à estimer ne représente plus qu'un seul pixel. Le déplacement d'environ un pixel est alors estimé entre les deux images grossières par la méthode Lucas & Kanade. Mais en diminuant la taille, et le nombre de pixels, la paire d'images perd inévitablement de l'information. Cela implique une forte imprécision sur le mouvement estimé. La mesure de mouvement est alors raffinée sur une image plus grande, en utilisant l'estimation grossière comme nouvelle initialisation (voir la figure 6.9). Bien évidement, l'estimation grossière doit être suffisamment proche de la solution, pour que l'algorithme Lucas & Kanade fonctionne. Ce dernier point limite l'écart de résolution. Dans le cas où l'écart de résolution entre l'image grossière et l'original est trop important, la parade très largement utilisée [BAHH92, SC94, Sze94, BJ96, HB96, Gle97, ETC98, Bou00] consiste à procéder par étapes, en faisant intervenir des images intermédiaires de plus en plus résolues.

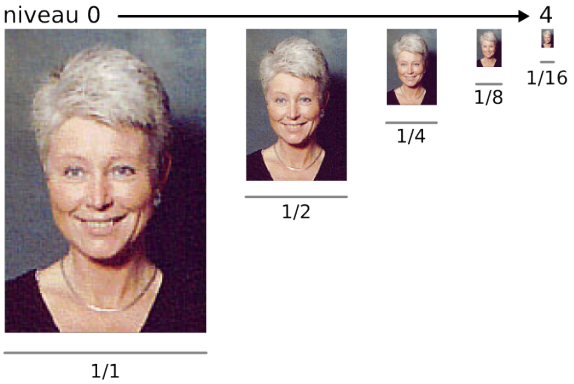


FIG. 6.5: La pyramide d'image est une représentation multi-résolution d'une image. Dans ce cas, *Lena* est représentée en pleine résolution au niveau 0 et le niveau 4 affiche le niveau le plus grossier. Entre chaque niveau de la pyramide, le nombre de pixels de l'image est divisé par 4 et les distances par 2.

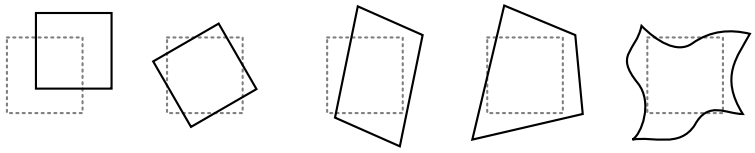
Termes d'ordre supérieur

Dans l'algorithme standard, l'approximation au premier ordre (équation 6.4) faite pour minimiser l'équation 6.1 peut engendrer un échec de la convergence. La proportion d'échecs peut être améliorée en utilisant l'algorithme de minimisation de Newton. Mais cet algorithme nécessite le calcul d'un développement limité au deuxième ordre. Pour résoudre un système du second ordre, la matrice Hessienne doit être calculée en plus de la matrice Jacobienne, engendrant des calculs coûteux. Nagel *et al.* [Nag87] proposent une technique de minimisation au second ordre efficace. D'après des expérimentations, Bainbridge *et al.* [SL97]

démontrent que l'approximation au premier ordre pondérée (typiquement par une fenêtre gaussienne) une donne les meilleurs résultats.

Déformation

Dans l'algorithme original, la fonction d'extraction W qui modélise le changement d'état se limite aux translations pures (voir la figure 6.6 et éq. 6.2). Ainsi, le vecteur d'état $\mathbf{p} = \begin{bmatrix} u & v \end{bmatrix}^\mathsf{T}$ représente le vecteur de déplacement 2-D. Si la translation pure convient très bien pour modéliser le flux optique,



(a) translation (b) rotation (c) affine (d) perspective (e) aurtes

FIG. 6.6: Les différentes fonctions $\mathsf{W}(\mathbf{x};\mathbf{p})$ de transformations.

le suivi d'objets 3-D peut nécessiter la mise en œuvre de modèles plus complexes. Par exemple, le modèle de déformation affine est couramment utilisé [HB96, BAHH92, ST94, BJ96] (voir la figure 6.6). Dans ce cas, la fonction W s'écrit :

$$\mathsf{W}(\mathbf{x};\mathbf{p}) = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \mathbf{x} + \begin{bmatrix} u \\ v \end{bmatrix} \tag{6.9}$$

où $\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ est la matrice de transformation affine et $\begin{bmatrix} u \\ v \end{bmatrix}$ est le vecteur de translation. Dans ce cas, le vecteur de paramètres $\mathbf{p}$ se compose des variables a_{11}, a_{12}, a_{21}, a_{22}, u et v. On peut trouver des modèles encore plus complexes, permettant par exemple de modéliser l'effet de perspective [DMG06], ou des paramètres très spécifiques, comme des déformations de papier [PB07] ou des déformations faciales [ETC98].

Plus le nombre de paramètres du vecteur $\mathbf{p}$ est grand, plus les calculs à effectués sont important. En effet, rappelons que le nombre de paramètres de $\mathbf{p}$ conditionne la taille de la matrice Jacobienne $\frac{\partial \mathsf{W}}{\partial \mathbf{p}}$. Cette matrice, et l'approximation de la matrice hessienne qui en découle, doivent être recalculées pour chaque point suivi (une fois, ou à chaque itération suivant les implantation).

Images couleurs

La mesure de corrélation utilisée dans l'algorithme KLT (éq. 6.1) est basée sur les différences des intensités. Dans le cas d'images couleurs, l'information chromatique est ignorée. En général, cette réduction de l'information n'a pas trop d'importance, puisque la luminance représente l'essentiel de l'information visuelle. Mais dans certains cas, l'utilisation de la couleur peut s'avérer déterminante. Dans ce cas, il faut donc redéfinir la fonction de corrélation pour tenir compte de la chrominance [SI97, HPN99, Gou05]. La principale difficulté réside dans le caractère vectoriel inhérent à la couleur.

Changements d'illumination

L'algorithme *LK* répond au problème de mise en correspondance de deux fenêtres de pixels. Cette approche n'est applicable au suivi que si l'apparence de la cible ne varie par au cours du temps. En effet, pour que la fonction de corrélation donne de bons résultats, il ne faut ni changement d'éclairage, ni reflets, ni spécularités. Les scènes d'extérieurs réunissent rarement les conditions optimales, c'est pourquoi nombre d'approches [Bar06, SC94, Sze94, SI97, ETC98, HB96, Gou05] proposent d'adapter la fonction de corrélation (6.1) afin qu'elle tolère au moins les changements d'illumination. Ce phénomène est le plus souvent modélisé par une fonction linéaire à deux paramètres α et β. La fonction de corrélation devient :

$$\sum_{\mathbf{x}} \left[\alpha \times I(\mathsf{W}(\mathbf{x};\mathbf{p})) + \beta - T(\mathbf{x})\right]^2, \tag{6.10}$$

Les deux paramètres α et β supplémentaires doivent être estimés en même temps que les paramètres $\mathbf{p}$, ce qui complexifie le calcul. Pour accélérer le temps de calcul, Bartoli [Bar06] propose une implantation qui minimise les traitements à effectuer à chaque itération.

Echantillonage de l'espace d'état

Le suivi visuel par corrélation de texture estime les paramètres qui caractérisent la déformation d'une fenêtre par rapport à l'autre. Certaines approches [Gle97, SI97] apprennent les variations possibles des paramètres en dehors du processus de suivi. Bien que ces approches soient efficaces, elles ne sont pas utilisables dans le cas d'un suivi en temps réel, car l'étape d'apprentissage n'est pas réalisable en temps réel.

Texture adaptative

La méthode de Edwards *et al.* [ETC98] considère la texture comme une des inconnues du problème. Le jeu de paramètres qui la caractérise est donc estimé en même temps que les paramètres de pose. Pour des raisons évidentes, ces paramètres sont synthétisés dans un espace réduit, défini préalablement à l'aide d'une analyse en composantes principales. Devernay *et al.* [DMG06] et Dellaert *et al.* [DTT98] associent la variance à l'intensité de chaque pixel. La texture, ainsi que sa variance, sont continuellement mises à jour en utilisant un filtrage de Kalman.

Suivi par détection

S. Avidan [Avi04] a été le premier à proposer une modification d'une méthode de détection (*Support Vector Machine*) afin de la rendre utilisable pour le suivi. La nouvelle méthode *SVT* (pour *Support Vector Tracking*), est à mi-chemin entre un processus de suivi et de détection et elle tire profit des deux types de processus. L'idée est de maximiser la fonction *SVM*, habituellement utilisée pour la classification. La recherche de cet optimum est accélérée par l'utilisation d'un développement de Taylor au premier ordre. C'est une technique connue pour le calcul des flux optiques [LK81]. Même s'il s'en inspire, cet algorithme n'est pas un algorithme de suivi. En effet, les méthodes de suivi se basent sur des observations successives (voir la figure 6.1), alors que le *SVT* n'utilise que

l'observation du temps t. La détection précédente est néanmoins utilisée comme initialisation, permettant ainsi de retrouver la cible, entre deux acquisitions successives. T. Chateau *et al.* [CGBCL06] reprennent l'idée en l'adaptant aussi à *ADABOOST*.

Cette approche est très intéressante, car elle améliore efficacement la partie détection pour le suivi. Mais elle garde certaines limitations propres aux méthodes de détection basées sur un apprentissage. En effet, les performances du suivi dépendent fortement de la base d'apprentissage. Tout d'abord, il est impossible de suivre des objets pour lesquels l'algorithme n'a pas été entraîné. Ensuite, SVM n'est pas conçu pour détecter précisément la cible (voir la figure 6.8(a)), ce qui peut engendrer des instabilités durant le suivi.

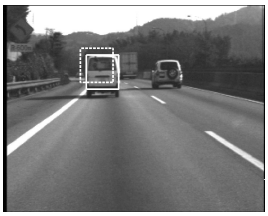
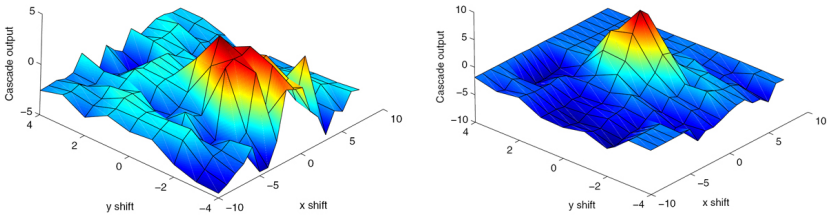


FIG. 6.7: (à droite) L'approche *Support Vector Tracking* [Avi04] part de la position précédente (trait pointillé) pour converger vers une texture optimisant le critère *SVM* (trait plein). La mise à jour itérative du vecteur de déplacement est inspirée des méthodes de calcul de flux optique.



(a) (b)

FIG. 6.8: La réponse de la fonction *SVM* en fonction d'un décalage de position dans l'image ($(0,0)$ étant la position réelle). La fonction n'a pas forcément un maximum marqué (a), ce qui peut rendre le suivi instable. En créant une base d'apprentissage spécifique au suivi [DW06], la fonction *SVM* devient plus marquée (b), rendant le suivi plus stable.

Witthopf *et al.* [DW06] proposent de contourner ce problème en utilisant une base entraînée pour le suivi et non pour la détection (voir la figure 6.8(b)).

6.2.3 Implantation multi-résolution fournie dans *OpenCV*

Dans sa librairie *C++* de traitement d'images, *Intel* fournit deux implantations de l'algorithme de suivi de Lucas & Kanade : *cvCalcOpticalFlowLK* et *cvCalcOpticalFlowPyrLK*. Dans cette section, nous détaillons la deuxième, car son auteur – Jean Yves Bouguet [Bou00] – y a inclus des optimisations et des améliorations significatives.

Nous commençons par présenter l'approche pyramidale qui permet de suivre de larges déplacements. En effet, à l'instar des méthodes d'optimisation locale,

l'algorithme *LK* original peut échouer quand l'initialisation (généralement par un vecteur nul) est trop éloignée de la solution. Puis nous décrivons l'extension qui permet au processus d'atteindre une précision sous-pixelique. Nous présentons ensuite quelques optimisations qui permettent d'accélérer les temps de calcul. Pour finir, ces modifications sont synthétisées sous la forme de l'algorithme effectivement codé dans *OpenCV*.

L'approche pyramidale

Pour obtenir ces différents niveaux de détail, l'implantation de *Bouguet* construit plusieurs versions de l'image et les hiérarchise selon leur résolution (voir la figure 6.5). Cette représentation, très utilisée dans le domaine de la vision par ordinateur, s'appelle une pyramide.

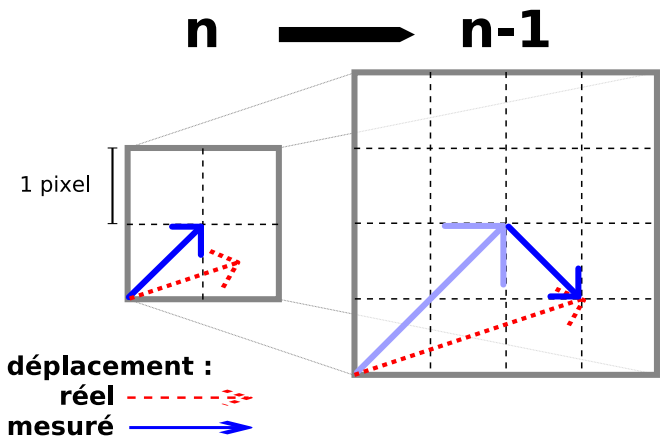


FIG. 6.9: L'estimation au niveau de résolution $n-1$ raffine celle précédemment effectuée au niveau n.

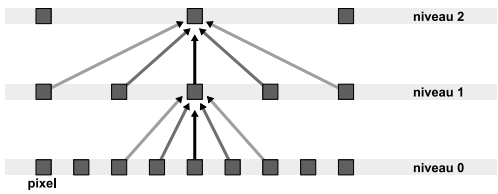


FIG. 6.10: La pyramide gaussienne est construite récursivement : chaque niveau résulte du filtrage et de l'élagage du niveau inférieur – par simplicité, l'image est ici représentée par une ligne de pixels.

Il existe plusieurs types de pyramide, se différentiant essentiellement par le processus de réduction de l'image. Bouguet a choisi une pyramide gaussienne, faisant intervenir un filtrage en plus de la réduction de taille. Une approche récursive permet de calculer un niveau de la pyramide gaussienne à partir de son prédécesseur : l'image du niveau $N-1$ est convoluée par l'approximation

d'un filtre gaussien (voir la figure 6.10), ainsi dépouillée d'une ligne et d'une colonne sur deux. L'image filtrée et réduite obtenue constitue le nouvel étage N de la pyramide.

Par conséquent, chaque étage de la pyramide a une surface 4 fois plus petite que le précédent, et les distances y sont deux fois moindre. L'implantation actuelle tolère au plus 7 niveaux (en plus de l'image originale), ce qui permet de suivre jusqu'à $2^7 = 128$ pixels de déplacement.

La taille totale de la pyramide est définie en nombre de pixels par :

$$s \times \sum_{i=0}^{N} \frac{1}{4}^{i}, \tag{6.11}$$

avec s le nombre de pixels de l'image originale, et N le nombre de niveaux de la pyramide. A noter que l'équation 6.11 a une asymptote en $s \times 4/3$ quand $N \to \infty$. Autrement dit, la taille mémoire nécessaire au codage de la pyramide (hors image originale) n'excédera jamais 1/3 de celle de l'image originale.

Précision sous-pixelique

Rappelons que le but de l'algorithme de Lucas & Kanade est de minimiser la fonction de corrélation 6.1. Cette fonction de corrélation compare un à un tous les pixels $T(\mathbf{x})$ à leur homologue respectif dans I (donné par $I(\mathsf{W}(\mathbf{x};\mathbf{p}))$). Pour qu'elle soit calculable, il faut que chaque pixel $T(\mathbf{x})$ ait un unique pixel homologue dans I. Or, cette propriété n'est pas toujours valide. En effet, lorsque les deux textures sont séparées par des coordonnées non-entières, un pixel $T(\mathbf{x})$ recouvre plusieurs pixels de I. C'est pourquoi la méthode est limitée à l'estimation de déplacements de coordonnées entières.

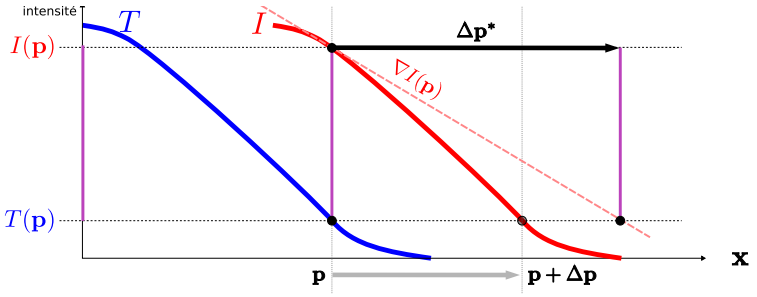
Pour généraliser la formulation à des déplacements sous-pixeliques, Bouguet modifie la fonction W qui apparie un pixel de I à un pixel de T. Le but est de pouvoir comparer $T(\mathbf{x})$ à un pixel aux coordonnées non entières dans I. La solution consiste à approximer un tel pixel par la combinaison linéaire des plus proches pixels réellement présent dans l'image I (interpolation bilinéaire). Cette extension permet d'évaluer la corrélation, et donc le déplacement, pour des coordonnées non-entières.

Résoudre le problème opposé pour diminuer les calculs

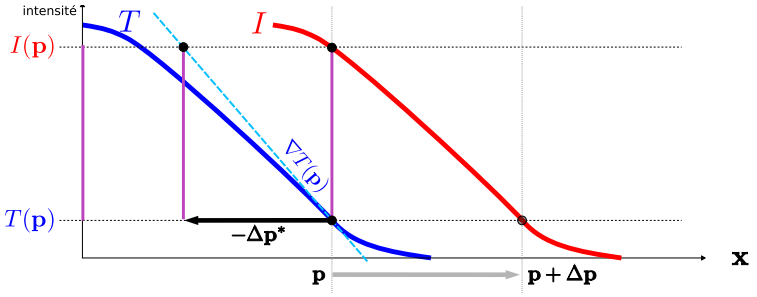
Comme le montre la figure 6.11(a), l'approximation au premier ordre de l'équation 6.4) met en œuvre la dérivée de l'image ∇I. Bouguet aborde le problème de façon inverse en utilisant la dérivée ∇T du *template* T. Comme le démontre la figure 6.11(b), le résultat n'est pas identique, mais l'approche tout à fait similaire. Cette ré-écriture permet deux optimisations très performantes. Premièrement, il n'est plus nécessaire de calculer le gradient sur toute l'image I, mais seulement sur le *template* T. T étant beaucoup plus petit que I, le gain de performance est significatif. Deuxièmement, l'équation 6.7 devient :

$$H = \sum_{\mathbf{x}} \left[\nabla T \frac{\partial \mathsf{W}}{\partial \mathbf{p}}\right]^{\mathsf{T}} \left[\nabla T \frac{\partial \mathsf{W}}{\partial \mathbf{p}}\right], \tag{6.12}$$

L'approximation de la matrice Hessienne ne dépend alors plus que de ∇T et de $\mathbf{x}$ (la géométrie de la fenêtre). Il n'est donc plus nécessaire de calculer et



(a) Lucas & Kanade



(b) Bouguet

FIG. 6.11: Là où Lucas & Kanade utilise la dérivée partielle ∇I de l'image I pour calculer l'approximation $\Delta\mathbf{p}^*$, la méthode de Bouguet utilise la dérivée partielle ∇T du template T pour calculer l'approximation $-\Delta\mathbf{p}^*$.

d'inverser cette matrice à chaque itération. Seul un changement de la taille $\mathbf{x}$ de la fenêtre impose de recalculer H^{-1}, ce qui se produit lorsque la fenêtre déborde l'image. Sachant que l'algorithme converge en une dizaine d'itérations en moyenne, ce sont environ 9 constructions et inversions de matrices 3×3 qui sont économisées.

Parallélisation des tâches

La percée récente des architectures multi-cœurs, aussi bien sur le marché grand public que professionnel, avantage les algorithmes parallélisés. Cet aspect est pris en compte dans la dernière version d'*OpenCV (release 1.0)*, puisque la fonction *cvCalcOpticalFlowPyrLK* se voit remaniée de sorte que les points suivis puissent être suivis indépendamment les uns des autres.

Algorithme 5 : Approche pyramidale de l'algorithme de Lucas & Kanade, appliqué au suivi de points 2-D – implantation de J.Y. Bouguet [Bou00] dans *OpenCV release 1.0* [Int01].

```
pour niveau l ← level to 0 faire
    // Calcul du facteur multiplicateur au niveau l de la pyramide
    s ← FacteurEchelle(l);
    // Mise a l'échelle de l'image à la date t
    I ← Pyramide( Image(t), s);
    // Mise a l'échelle de l'image à la date t + Δ_t
    J ← Pyramide( Image(t + Δ_t), s);
    // Boucle parallélisable
    pour tous les points i faire
        // i est le point à suivre
        si Status(i) = Mauvais alors ne pas traiter ce point ;
        // u et v sont les vecteurs d'état de i aux dates t et t + Δ_t.
        u ← Point(i,t) ×s;
        v ← Point(i,t + Δ_t);
        // Copie ou mise à jour de v.
        si l = level alors v ← v ×s;
        sinon v ← v + v;
        // Extraction de la fenêtre autour de i à la date t.
        patchI ← SousRegion(I,u,taille);
        // et calcul de son gradient dans les deux directions.
        patchIx ← GradientX(patchI);
        patchIy ← GradientY(patchI);
        // Liste des pixels à l'intérieur de la fenêtre et de l'image
        p_i ← ListePixels(PatchI);
        // Calcul itératif de v, se référer à l'algorithme 6.
        v ← ItérationLK( patchI, patchIx, patchIy, p_i, J, v, taille);
        Point(i,t) ← v;
    fin
fin
```

Algorithme 6 : $MiseAJourB$ de l'algorithme 8.

```
Data : b_x, b_y, t, patchIx(p), patchIy(p)
Result : b_x, b_y
begin
    b_x ← b_x + t× patchIx(p) ;
    b_y ← b_y + t× patchIy(p) ;
    return b_x, b_y ;
end
```

Algorithme 7 : $MiseAJourH$ de l'algorithme 8.

Data : H_{xx}, H_{xy}, H_{yy}, t, patchIx(p), patchIy(p)
Result : H_{xx}, H_{xy}, H_{yy}
begin
  $H_{xx} \leftarrow H_{xx}$ + patchIx(p) $\times$ patchIx(p) ;
  $H_{xy} \leftarrow H_{xy}$ + patchIx(p) $\times$ patchIy(p) ;
  $H_{yy} \leftarrow H_{yy}$ + patchIy(p) $\times$ patchIy(p) ;
  return H_{xx}, H_{xy}, H_{yy} ;
end

Algorithme 8 : *ItérationLK* de l'algorithme de l'algorithme 5.

```
Data : patchI, patchIx, patchIy, p_i, J, v, taille
Result : v
begin
    tant que iteration j ≤ maxIteration faire
        // Extraction de la fenêtre autour de i à la date t + Δ_t.
        patchJ ← SousRegion(J,v,taille);
        // et de la liste des pixels correspondants
        p_j ← ListePixels(patchJ);
        // calcul des pixels p communs aux deux fenêtres
        p ← p_i ∩ p_j ;
        // calcul des paramètres de H et de b
        si p inchangé alors
            b_x ← b_y ← 0;
            pour tous les pixel p faire
                t ← patchI(p) - patchJ(p);
                // Se référer à l'algorithme 6 pour plus de détails
                (b_x, b_y) ← MiseAJourB( b_x, b_y, t, patchIx(p), patchIy(p));
            fin
        sinon
            b_x ← b_y ← H_xx ← H_xy ← H_yy ← 0;
            pour tous les pixel p faire
                t ← patchI(p) - patchJ(p);
                // Se référer aux algorithmes 6 et 7 pour plus de
                   détails
                (b_x, b_y) ← MiseAJourB( b_x, b_y, t, patchIx(p), patchIy(p));
                (H_xx, H_xy, H_yy) ← MiseAJourH( H_xx, H_xy, H_yy, t, patchIx(p),
                patchIy(p));
            fin
        fin
        déterminant ← H_xx × H_yy − H_xy × H_xy ;
        si déterminant < ε_a alors
            // abandonner si le déterminant est trop faible
            Status(i) = Mauvais ;
            return ;
        déterminant ← 1/ déterminant ;
        Δ_x ← (H_yy × b_x − H_xy × b_y)× déterminant ;
        Δ_y ← (H_xx × b_x − H_xy × b_y)× déterminant ;
        // mise à jour du vecteur d'état v
        v ← v + (Δ_x, Δ_y) ;
        // arrêter si la mise a jour n'es pas significative
        si  Δ_x^2 + Δ_y^2 < ε_b alors return ;
    fin
    return v;
end
```

6.2.4 Utiliser les deux images

Deux vues d'un même objet

Dans un système stéréoscopique, à chaque instant t, les deux caméras délivrent les images I_l (gauche) et I_r (droite). Ainsi, le point 3-D $\mathbf{M}$ se projette dans les images gauche et droite en deux points 2-D : $\mathbf{m}_l$ et $\mathbf{m}_r$. Rappelons que l'algorithme de Lucas & Kanade caractérise un point 2-D par une fenêtre de pixels. Dans le cas stéréoscopique, nous choisissons donc d'utiliser deux fenêtres de pixels. Une pour chaque point 2-D. En effet, bien que $\mathbf{m}_\mathrm{l}$ et $\mathbf{m}_\mathrm{r}$ soient issus

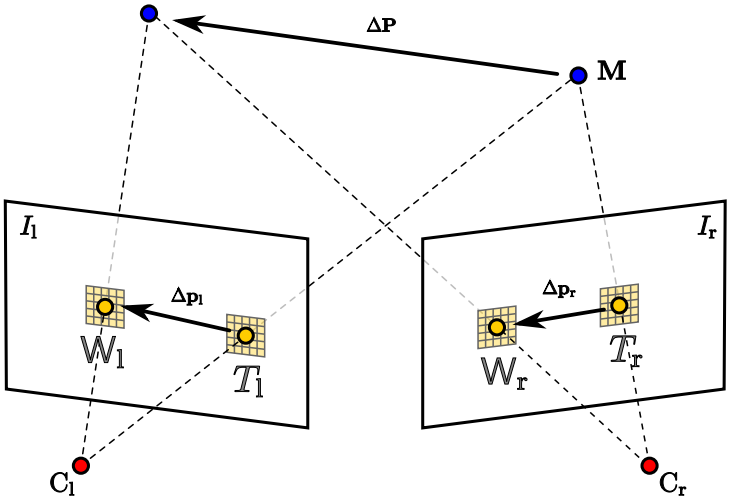


Fig. 6.12: L'algorithme de Lucas & Kanade permet de suivre un point à l'aide d'une fenêtre de texture. Pour suivre une paire de points, il est souhaitable d'utiliser deux textures (une pour chaque point), car l'apparence du même objet peut varier suivant le point de vue.

du même point 3-D $\mathbf{M}$, leurs apparences diffèrent, à cause de l'effet de parallaxe. Cette différence d'apparence nous a poussé à caractériser l'élément suivi par deux textures différentes T_l et T_r, de tailles $\mathbf{x}_l$ et $\mathbf{x}_r$, associées à chacune des deux vues. La fonction de coût (6.1) devient :

$$\sum_{\mathbf{n}\in\{\mathbf{l},\mathbf{r}\}} \sum_{\mathbf{x}_n} \left[I_n(\mathsf{W}_n(\mathbf{x}_n;\mathbf{p})) - T_n(\mathbf{x}_n) \right]^2 , \tag{6.13}$$

où $n \in \{l, r\}$ indique la vue gauche ou droite, et $\mathbf{p}$ est constitué des 4 coordonnées de la paire de points (m_l, m_r)

Mise à jour des paramètres

En utilisant l'approximation au premier ordre d'un développement de Taylor de la même façon que dans l'équation (6.13), l'équation (6.3) donne :

$$\sum_{\mathbf{n}\in\{\mathbf{l},\mathbf{r}\}} \sum_{\mathbf{x}_n} \left[I_n(\mathsf{W}_n(\mathbf{x}_n;\mathbf{p})) + \nabla I_n \frac{\partial \mathsf{W}_n}{\partial \mathbf{p}} \Delta\mathbf{p} - T_n(\mathbf{x}_n) \right]^2 , \tag{6.14}$$

où ∇I_n est le gradient de l'image I_n. La dérivée partielle de l'équation (6.14) par rapport à $\Delta\mathbf{p}$ s'écrit :

$$2 \sum_{\mathbf{n}\in\{\mathbf{l},\mathbf{r}\}} \sum_{\mathbf{x}_n} \left[\nabla I_n \frac{\partial \mathsf{W}_n}{\partial \mathbf{p}}\right]^{\mathsf{T}} \left[I_n(\mathsf{W}_n(\mathbf{x}_n;\mathbf{p})) + \nabla I_n \frac{\partial \mathsf{W}_n}{\partial \mathbf{p}} \Delta\mathbf{p} - T_n(\mathbf{x})\right]. \tag{6.15}$$

Au minimum, le vecteur des dérivées partielles doit être nul, ce qui conduit à la même forme d'équation que (6.6), mais où l'approximation par la méthode Gauss-Newton de la matrice Hessienne H et b s'écrivent cette fois :

$$H = \sum_{\mathbf{n}\in\{\mathbf{l},\mathbf{r}\}} \sum_{\mathbf{x}} \left[\nabla I_n \frac{\partial \mathsf{W}_n}{\partial \mathbf{p}}\right]^{\mathsf{T}} \left[\nabla I_n \frac{\partial \mathsf{W}_n}{\partial \mathbf{p}}\right], \tag{6.16}$$

$$b = \sum_{\mathbf{n}\in\{\mathbf{l},\mathbf{r}\}} \sum_{\mathbf{x}_n} \left[\nabla I_n \frac{\partial \mathsf{W}_n}{\partial \mathbf{p}}\right]^{\mathsf{T}} \left[T_n(\mathbf{x}_n) - I_n\left(\mathsf{W}_n\left(\mathbf{x}_n;\mathbf{p}\right)\right)\right]. \tag{6.17}$$

Il reste à définir deux fonctions W_l et W_r, ainsi que le vecteur de paramètres $\mathbf{p}$.

6.2.5 Vecteur de paramètres p

Suivi non-contraint

Sans aucune considération géométrique, suivre une paire de points dans une séquence stéréo est équivalent à suivre les deux points de manière indépendante. Dans le reste du manuscrit, cette approche sera prise comme référence pour l'évaluation des différentes méthodes. Nous l'appelons suivi *non-contraint*. La paire de points est décrite par un vecteur de paramètres $\mathbf{p}$ dont les 4 composantes sont constituées de la paire de coordonnées 3-D.

Rappelons que la relation entre les deux vues d'une même scène induit une contrainte géométrique forte (voir la partie 2.2.2 p. 26). Car, si pour tout point 3-D il existe une paire de points projetés dans les images (sans tenir compte de la limite de taille des capteurs), la réciproque n'est pas valide. En effet, il existe des paires de points 2-D ne correspondant à aucun point 3-D. Pour comprendre cela, associons une ligne de vue $l_n(\mathbf{m}_n)$ à chacun des deux points $\mathbf{m}_\mathrm{l}$ et $\mathbf{m}_\mathrm{r}$. Le point $\mathbf{M}$ reconstruit est à l'intersection des deux lignes de vues l_l et l_r. Mais si elles ne s'intersectent pas, aucune reconstruction n'est possible. Cependant, il existe une contrainte applicable aux points $\mathbf{m}_\mathrm{l}$ et $\mathbf{m}_\mathrm{r}$ qui permet de garantir que les lignes de vues $l_l(\mathbf{m}_\mathrm{l})$ et $l_r(\mathbf{m}_\mathrm{r})$ associées s'intersectent effectivement. Cette contrainte est appelée la contrainte épipolaire.

Un algorithme *non-contraint* peut fournir une paire de points 2-D qui ne vérifient pas la contrainte épipolaire. Il faut donc prendre en compte cette contrainte dans la formulation du problème pour limiter l'espace de recherche aux seules paires correspondant à des points 3-D. Cette contrainte, appelée contrainte épipolaire, réduit les degrés de liberté du problème.

Intégrer la contrainte épipolaire

Pour prendre en compte cette contrainte, le vecteur de paramètres $\mathbf{p}$ peut simplement se composer des coordonnées 3-D euclidiennes du point $\mathbf{M} = (X, Y, Z)$. Cela implique la définition de deux fonctions de projection $\mathbf{P}_n$ transformant le point $\mathbf{M}$ en $\mathbf{m}_n$ dans les images I_n (avec $n \in \{\text{gauche}, \text{droite}\}$). Cependant, nous

préférons utiliser une représentation de l'espace 3-D basée sur l'espace image. Par la suite, nous appellerons ce système de coordonnées espace 3-D *basé image*. La relation entre cet espace et le système de coordonnées de l'image est direct, permettant ainsi de conserver les caractéristiques homogènes et isotropiques [DD01] du bruit de mesure.

Pour comprendre l'espace 3-D *basé image*, il faut comprendre la notion de contrainte épipolaire. L'ensemble des points 3-D se projetant en un point image donné $\mathbf{m}_\mathrm{l}$ définissent une droite 3-D (cf. 2.2.2 p. 26), impliquant que son correspondant $\mathbf{m}_\mathrm{r}$ dans l'autre image appartient à une droite : la ligne épipolaire. Cette contrainte permet de définir un point 3-D, non pas avec des coordonnées euclidienne, mais avec les coordonnées du point $\mathbf{m}_\mathrm{l}$ associée à l'ordonnée sur la droite épipolaire (la disparité) du point $\mathbf{m}_\mathrm{r}$. Ces trois paramètres – abscisse, ordonnée et disparité – constituent les trois composantes du vecteur de paramètres $\mathbf{p}$.

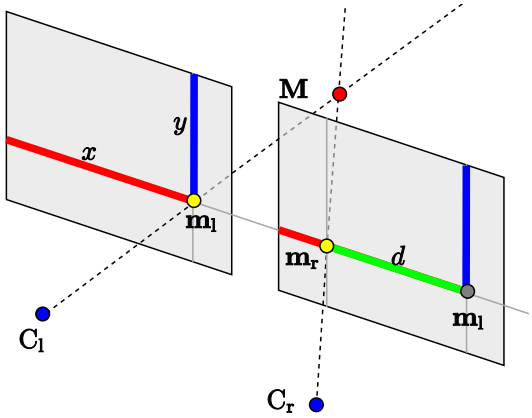


FIG. 6.13: $\mathbf{m}_l = (x_l, y_l)$ associé à la disparité d (l'ordonnée de $\mathbf{m}_r$ sur la droite épipolaire) constituent les trois dimensions du nouveau vecteur de paramètres $\mathbf{p}$.

Les applications automobiles requièrent, le plus souvent, une mesure 3-D métrée. Il existe une fonction de passage entre l'espace 3-D *basé image* et l'espace euclidien métré. La définition de cette fonction nécessite un calibrage fort du système (se référer au chapitre 2). Retenons surtout que la fonction de transfert n'est pas linéaire, et que les rapports de distances changent d'un espace à l'autre. La conséquence principale est qu'un ensemble de points reconstruits avec une précision constante dans l'espace basé image engendrera un ensemble de points 3-D dont la précision de reconstruction est proportionnelle à la distance entre les caméras et ces points.

La disparité

Dans la configuration standard, la disparité entre $\mathbf{m}_\mathrm{l}$ et $\mathbf{m}_\mathrm{r}$ est un vecteur 2-D défini par $\vec{d} = O_r\vec{\mathbf{m}}_\mathrm{r} - O_l\vec{\mathbf{m}}_\mathrm{l}$ [FP02], avec O_l et O_r les points principaux des deux images. Cependant, nous prenons pour hypothèse que la paire de caméras est dans une configuration fronto-parallèle (cf. Fig. 3.3), dans laquelle la ligne de base est alignée avec l'axe des abscisses et les axes optiques des deux caméras

sont parallèles. Dans cette configuration, les droites épipolaires sont parallèles aux lignes de l'image et par conséquent la disparité peut être exprimée par un nombre réel :

$$d = (x_r - O_{xr}) - (x_l - O_{xl}). \tag{6.18}$$

Si en plus les points principaux sont placés au même endroit dans les images, alors l'expression se simplifie, et $d = x_\mathrm{r} - x_\mathrm{l}$. Notons que la disparité nulle correspond à une distance infinie. Si les caméras ne sont pas dans cette configuration spécifique, il existe une transformation 2-D qui permet de transformer les images de sorte à retrouver cette configuration [LZ99, PKG99].

Mise à jour des paramètres

Pour calculer la mise à jour $\Delta\mathbf{p}$ des paramètres fournis par l'équation (6.6), les éléments suivants doivent être définis : les textures T_l et T_r, les déformations $\mathbf{W}_\mathrm{l}$ et $\mathbf{W}_\mathrm{r}$, et finalement la Jacobienne de chaque déformation $\partial\mathbf{W}_l/\partial\mathbf{p}$ et $\partial\mathbf{W}_r/\partial\mathbf{p}$. Les textures T_l et T_r sont des fenêtres carrées extraites des images à la date t, centrées autour de $\mathbf{m}_\mathrm{l}$ et $\mathbf{m}_\mathrm{r}$. Rappelons que les déformations W_n sont des fonctions 2-D qui mettent en correspondance un point de texture de T_n avec un point de l'image I_n (où $n \in \{\mathrm{l}, \mathrm{r}\}$). Dans notre cas, chaque déformation $\mathbf{W}_n$ est une translation 2-D dans l'image I_n définie par :

$$\mathbf{W}_n(\mathbf{x}_n; \mathbf{p}) = \mathbf{x}_n + \mathbf{P_n}(\mathbf{p}), \tag{6.19}$$

où $\mathbf{p} = (x, y, d)$, et $\mathbf{P_n}(\mathbf{p})$ les fonctions projetant $\mathbf{p}$ dans les n images. Ces deux fonctions s'écrivent :

$$\mathbf{P}_l(\mathbf{p}) = \begin{pmatrix} x \\ y \end{pmatrix}, \qquad \mathbf{P}_r(\mathbf{p}) = \begin{pmatrix} x+d \\ y \end{pmatrix}. \tag{6.20}$$

Les deux jacobiennes $\partial\mathsf{W}_\mathrm{l}/\partial\mathbf{p}$ et $\partial\mathsf{W}_\mathrm{r}/\partial\mathbf{p}$ associées aux fonctions de déformations de l'équation 6.20 sont :

$$\frac{\partial\mathsf{W}_\mathrm{l}}{\partial\mathbf{p}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \qquad \frac{\partial\mathsf{W}_\mathrm{r}}{\partial\mathbf{p}} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}. \tag{6.21}$$

Le terme de plus grande pente devient donc :

$$\nabla I_\mathrm{l} \frac{\partial\mathsf{W}_\mathrm{l}}{\partial\mathbf{p}} = \begin{bmatrix} I_\mathrm{lx} & I_\mathrm{ly} & 0 \end{bmatrix}, \tag{6.22}$$

$$\nabla I_\mathrm{r} \frac{\partial\mathsf{W}_\mathrm{r}}{\partial\mathbf{p}} = \begin{bmatrix} I_\mathrm{rx} & I_\mathrm{ry} & I_\mathrm{rx} \end{bmatrix}, \tag{6.23}$$

avec $I_{n\mathrm{x}}$ (resp. $I_{n\mathrm{y}}$) le gradient de l'image n selon l'axe x (resp. y). Etant donnée l' Eq. (6.23), nous développons l'approximation de la matrice Hessienne H de l'Eq. (6.16) :

$$H = H_\mathrm{l} + H_\mathrm{r}, \tag{6.24}$$

où H_l et H_r dépendent des Eq. (6.23) et s'écrivent :

$$H_\mathrm{l} = \sum_{\mathbf{x}} \begin{bmatrix} I_\mathrm{lx}^2 & I_\mathrm{lx} I_\mathrm{ly} & 0 \\ I_\mathrm{lx} I_\mathrm{ly} & I_\mathrm{ly}^2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{6.25}$$

$$H_\mathrm{r} = \sum_{\mathbf{x}} \begin{bmatrix} I_\mathrm{rx}^2 & I_\mathrm{rx} I_\mathrm{ry} & I_\mathrm{rx}^2 \\ I_\mathrm{rx} I_\mathrm{ry} & I_\mathrm{ry}^2 & I_\mathrm{rx} I_\mathrm{ry} \\ I_\mathrm{rx}^2 & I_\mathrm{rx} I_\mathrm{ry} & I_\mathrm{rx}^2 \end{bmatrix}. \tag{6.26}$$

Rappelons que le calcul de $\Delta\mathbf{p}$ nécessite l'inversion de la matrice H. Avec cette formulation, seule une matrice doit être inversée, là où il aurait fallu en inverser deux avec l'approche non-contrainte. Mais surtout, l'utilisation de coordonnées 3-D permet d'intégrer la contrainte épipolaire directement dans la formulation de la fonction de coût. L'espace de recherche est ainsi réduit d'une dimension, pour ne conserver que les vecteurs valides dans l'espace 3-D. L'algorithme 9 (section 8.2.1 p. 161) reprend l'algorithme 5 en adaptant l'extension précédemment décrite.

6.2.6 Intégrer la relation entre distance et taille apparente

La disparité et la distance sont liées. Quand les coordonnées du point principal sont identiques dans les deux images, la disparité est définie par :

$$d = Bf/Z. \tag{6.27}$$

La ligne de base B (distance entre les deux centres optiques) et la distance focale f (la même pour les deux caméras) étant des valeurs constantes, une variation de disparité d dépend directement d'une variation de la distance Z entre l'objet et les caméras. De façon similaire, il existe une relation entre la taille apparente w d'un objet dans l'image et sa distance Z aux caméras (cf. figure. 6.14) :

$$w = Wf/Z, \tag{6.28}$$

où W désigne la taille réelle de la partie apparente de l'objet. Cette grandeur est équivalente à la taille de l'objet, si celui-ci est face aux caméras. Une grande variation de profondeur implique un changement significatif de la taille apparente w de l'élément dans les images. Dans le cas d'un suivi utilisant une fenêtre de taille fixe, le processus peut être imprécis ou échouer, car le changement d'aspect est trop important. Cette limitation n'est pas compatible avec l'application ACC, spécialement quand la distance par rapport à la cible devient faible.

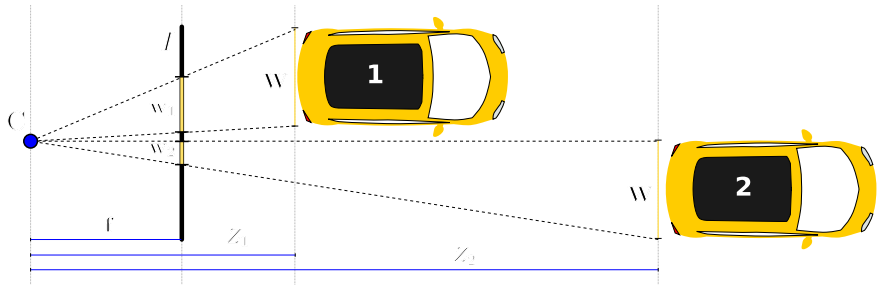


FIG. 6.14: La taille w d'un objet dans l'image dépend de sa taille réelle W mais aussi de la focale f et de sa distance Z aux caméras.

Par conséquent, nous proposons une méthode prenant en compte cette variation de taille dans l'image. Les variations de la taille apparente et de la disparité sont liées, et les équations (6.27) et (6.28) montrent que les variations de taille w et de disparité d dépendent toutes les deux des variations de profondeurs Z, conduisant à :

$$\frac{d}{\hat{d}} = \frac{Bf/Z}{Bf/\hat{Z}} = \frac{\hat{Z}}{Z} = \frac{Wf/Z}{Wf/\hat{Z}} = \frac{w}{\hat{w}}, \tag{6.29}$$

où $\hat{d}$ et $\hat{w}$ désignent respectivement la disparité et la taille apparente précédemment mesurées (date t). L'équation (6.29) permet d'exprimer la nouvelle taille apparente $w = \hat{w}.d/\hat{d}$, en fonction de la disparité. Les paramètres d'état de la cible reste donc $\mathbf{p} = (x, y, d)$, et la variation de la taille apparente peut être prise en compte dans les fonctions de déformation W_n.

Mise à jour des paramètres

Les deux fonctions de déformation W_n deviennent alors (une interpolation bilinéaire est utilisée pour échantillonner l'image) :

$$\mathsf{W}_n(\mathbf{x}_n; \mathbf{p}) = \left(\frac{d}{\hat{d}}\right) \mathbf{x}_n + \mathsf{P}_n(\mathbf{p}), \tag{6.30}$$

les matrices Jacobiennes deviennent :

$$\frac{\partial \mathsf{W}_\mathrm{l}}{\partial \mathbf{p}} = \begin{pmatrix} 1 & 0 & \frac{i}{\hat{d}} \\ 0 & 1 & \frac{j}{\hat{d}} \end{pmatrix}, \qquad \frac{\partial \mathsf{W}_\mathrm{r}}{\partial \mathbf{p}} = \begin{pmatrix} 1 & 0 & \frac{i}{\hat{d}} + 1 \\ 0 & 1 & \frac{j}{\hat{d}} \end{pmatrix}, \tag{6.31}$$

et les termes de plus grande pente :

$$\nabla I_n \frac{\partial \mathsf{W}_n}{\partial \mathbf{p}} = \begin{bmatrix} I_{n\mathrm{x}} & I_{n\mathrm{y}} & S_n \end{bmatrix}, \tag{6.32}$$

avec $n \in \{\mathrm{l}, \mathrm{r}\}$ et

$$S_\mathrm{r} = \frac{I_\mathrm{rx} i + I_\mathrm{ry} j}{\hat{d}} + I_\mathrm{rx}, \qquad S_\mathrm{l} = \frac{I_\mathrm{lx} i + I_\mathrm{ly} j}{\hat{d}}. \tag{6.33}$$

Reprenant l'équation (6.32), les matrices H_n de l'équation (6.24) deviennent :

$$H_n = \sum_{\mathbf{x}} \begin{bmatrix} I_{n\mathrm{x}}^2 & I_{n\mathrm{x}} I_{n\mathrm{y}} & I_{n\mathrm{x}} S_n \\ I_{n\mathrm{x}} I_{n\mathrm{y}} & I_{n\mathrm{y}}^2 & I_{n\mathrm{y}} S_n \\ I_{n\mathrm{x}} S_n & I_{n\mathrm{y}} S_n & S_n^2 \end{bmatrix}. \tag{6.34}$$

Cette extension est très avantageuse, car elle propose un modèle plus proche de la réalité, sans augmenter le nombre de paramètres. De plus, le surcoût de calcul reste très limité (S_l et S_r de l'équation 6.33). L'algorithme 14 (section 8.2.2 p. 164) étend l'algorithme 8.2.1 afin de tenir également compte des variations de tailles.

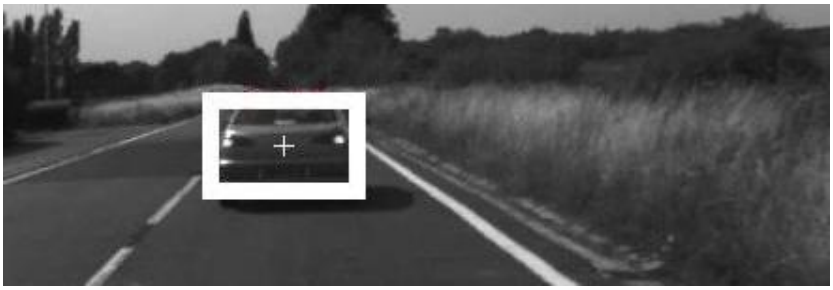
6.2.7 Utilisation de régions d'intérêt

L'algorithme de Lucas & Kanade, ainsi que les deux extensions précédemment présentées, sont conçues pour mesurer le déplacement d'un point. Or, un ensemble de points n'est pas la représentation la plus adaptée pour suivre un véhicule. En effet, dans l'algorithme Lucas & Kanade, les points sont caractérisés par des fenêtres de texture. L'utilisation de fenêtres d'une taille arbitraire, décorrélées de la taille apparente de la cible induit deux effets indésirables. Premièrement, la fenêtre de texture caractérisant un point peut recouvrir plusieurs objets. Quand ces objets ont des trajectoires différentes, le point suivi glisse entre les deux objets effectivement suivis. Deuxièmement, une fenêtre peut ne couvrir que partiellement la cible, ce qui est contradictoire avec les objectifs de précision.

Gérer des régions de taille définie

Une solution pour éviter ces limitations est de faire correspondre les fenêtres de texture T à une entité réelle, le véhicule. Au lieu de suivre des points 3-D, caractérisés par une fenêtre de taille arbitraire, nous proposons de suivre une région rectangulaire dont la taille est choisie pour s'adapter à l'élément suivi (voir la figure 6.15).

(a)



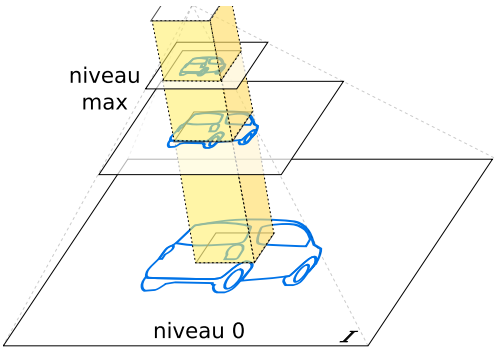
(b)

FIG. 6.15: Echantillonage d'une voiture par (a) 6 points de contrôles et (b) une région d'intérêt.

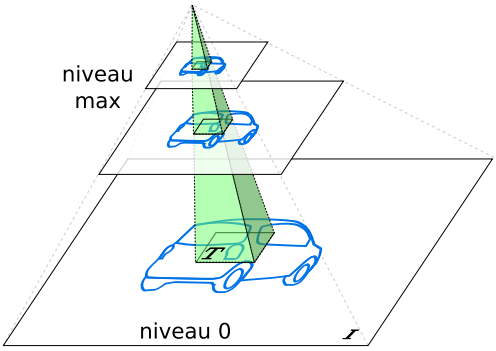
Modification de l'approche pyramidale

Pour suivre des mouvements amples, l'implantation de Bouguet [Bou00], comme la nôtre, utilise une approche pyramidale (voir la figure 6.16), qui estime un mouvement à une résolution grossière (typiquement 1/8 de la taille d'image originale) avant de raffiner à des échelles plus fines (1/4, 1/2, et 1).

En passant d'une échelle à l'autre, les dimensions de l'image sont divisées par deux. Le vecteur de paramètres $\mathbf{p}$ est mis à jour en conséquence. Si $\mathsf{W}(\mathbf{p})$ modélise une translation pure, alors les composantes de $\mathbf{p}$ sont divisées par deux. Par contre, dans l'implantation de Bouguet, la fenêtre de corrélation ($\sum_{\mathbf{x}}$ de l'équation 6.1) a une taille en constante en pixels, et ce, quelque soit le niveau de la pyramide. Par conséquent, comme l'illustre la figure 6.16(a), une fenêtre de $w \times h$ pixels à la résolution k recouvre une zone de $2^k w \times 2^k h$ dans l'image originale. Ce choix se justifie dans le cas du suivi de points, car la taille la fenêtre qui caractérise un point est choisie arbitrairement, suivant l'appréciation de l'utilisateur. Comme ce choix n'est pas relié au contenu de l'image, alors il n'a



(a)



(b)

FIG. 6.16: L'approche pyramidale de Bouguet [Bou00] (a) utilise toujours la même taille de fenêtre quelque soit le niveau de détail. La conséquence est que la fenêtre recouvre une plus grande région de l'image dans les niveaux les plus grossiers. Dans notre implantation (b), la taille de la fenêtre est mise à jour en fonction du niveau de résolution. Ainsi, la région considérée demeure la même.

pas besoin d'être mise en adéquation avec le niveau dans la pyramide (c.-à-d. la taille de l'image). Par contre, lorsque la taille de la fenêtre est liée au contenu de l'image, comme c'est le cas dans notre implantation, alors la taille de la fenêtre doit être mise en adéquation avec l'échelle de l'image, pour recouvrir toujours la même région de l'image originale (voir la figure 6.16(b)). Entre deux niveaux successifs de la pyramide, un rapport 2 lie les dimensions de la fenêtre en pixel. Rappelons que la fenêtre doit nécessairement être de dimensions impaires (en pixels), pour avoir un pixel central. Ainsi, au niveau les dimension de la fenêtre s'écrivent :

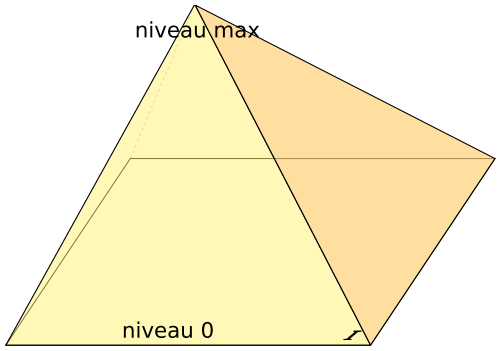
$$\begin{bmatrix} w_k \\ h_k \end{bmatrix} = \begin{bmatrix} \lfloor (w_0 - 1)/2^(k+1) \times 2 + 1 \rfloor \\ \lfloor (h_0 - 1)/2^(k+1) \times 2 + 1 \rfloor \end{bmatrix}, \tag{6.35}$$

avec (w_k, h_k) les dimensions en pixels de la fenêtre au niveau, et $\lfloor\rfloor$ l'opérateur d'arrondi à l'entier inférieur. De cette manière, le contenu de la fenêtre demeure équivalent (à l'échantillonnage près) quelque soit le niveau de la pyramide. Un des effets de bord de l'adaptation de la taille de la fenêtre est la génération

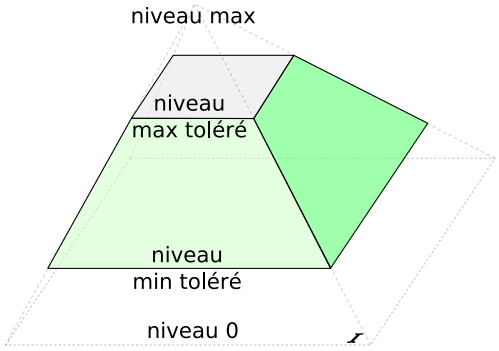
de fenêtres de tailles très différentes. Par exemple, un objet de grande taille va nécessiter une fenêtre de grande taille au niveau le plus résolu. Au contraire, un objet de taille petite ou moyenne entraîne la création de très petites fenêtres aux niveaux les plus faiblement résolus. Ces deux extrêmes constituent des cas problématiques pour l'algorithme de suivi. Dans le cas d'une texture petite, 3×3 pixels par exemple, l'information fournie par les 9 pixels n'est pas assez discriminante pour apparier la cible avec précision au cours du temps. D'un autre côté, une fenêtre très grande demandera un temps de calcul important. Pour contourner le problème, en fonction de la fenêtre traité, nous éliminons simplement les niveaux de la pyramide posant problème. En pratique, nous limitons les calculs à des fenêtres vérifiant :

$$25 \leq (w \times h) \leq 1600, \tag{6.36}$$

où w et h sont les largeur et hauteur en pixels.



(a)



(b)

FIG. 6.17: (a) L'approche pyramidale de Bouguet [Bou00] évalue le suivi à tous les niveau de la pyramide. (b) L'adaptation de la taille de la fenêtre de texture nous impose d'ignorer certain des niveaux aux deux extrémités de la pyramide.

Le fait d'ignorer certains niveaux de la pyramide se répercute sur l'amplitude des mouvements perceptibles par l'approche. Comme nous l'avons déjà précisé, l'approximation au premier ordre (équation 6.4) limite le suivi à des déplacements d'amplitude ± 1 pixel, et c'est pourquoi l'approche pyramidale est

envisagée. Avec 7 niveaux de pyramide, l'algorithme standard peut gérer des déplacements de ±64 pixels. De manière plus général, un mouvement de 0.5 pixel au niveau k correspond à un mouvement de 2^{k-1} pixels dans l'image d'origine. En limitant $k_{\max}$, le niveau maximal (le plus grossier) de la pyramide, nous limitons donc l'amplitude des mouvements perceptibles suivant la formule de l'équation 6.35.

Au contraire, en limitant le niveau minimal de la pyramide (c.-à-d. en utilisant pas l'image à pleine résolution) le suivi n'atteint pas sa précision maximale. Rappelons cependant que lorsque l'algorithme de Lucas & Kanade est utilisé sur une image en pleine résolution, le déplacement est mesuré avec une précision sous-pixelique de l'ordre de 0.1 pixel. En pratique, l'image en pleine résolution sera parfois ignorée, divisant la précision par deux.

6.3 Validation

L'objectif de cette partie est de présenter une évaluation des extensions proposées à l'algorithme *LK*. Pour mettre en évidence l'apport de la contrainte épipolaire et la prise en compte des variations de taille, nous avons comparé l'algorithme de Lucas & Kanade *non-contraint* avec plusieurs versions de notre algorithme.

La première utilisant des coordonnées 3-D *basées image*, la seconde tenant aussi en compte des variations d'échelle et la troisième utilisant régions d'intérêts plutôt que des points. Le tableau 6.1 donne les notations de ces trois versions.

Nous présentons une évaluation quantitatives obtenue sur des séquences synthétiques. Le résultats de ces évaluations met en évidence le gain de performance apporté par notre extension tenant compte de la contrainte épipolaire et des variations de taille. Dans les résultats, nous détaillons l'apport de chacun de ces aspects. En ce qui concerne la troisième partie de notre extension, à savoir le suivi de régions d'intérêts, elle est difficilement comparable à un suivi de points. Pour cette raison, nous ne présentons que des résultats qualitatifs sur des séquences réelles, mettant en avant l'intérêt d'une telle approche pour une application automobile.

Tab. 6.1: Notation de l'algorithme original et des 3 versions de notre extension.

méthode	notation	contrainte épipolaire	variation de taille	région
non-contraint	ULK	✗	✗	✗
3-D basé image	ELK	✓	✗	✗
avec grandissement	SLK	✓	✓	✗
avec région	RLK	✓	✓	✓

6.3.1 Évaluation quantitative

L'objectif de cette partie est de démontrer que la re-formulation binoculaire que nous proposons de l'algorithme Lucas & Kanade améliore la précision de cette algorithme. L'évaluation est effectuée de manière quantitative en appliquant les différents algorithmes sur plusieurs séquences d'images synthétiques.

Pour mettre en évidence l'apport de la contrainte épipolaire ELK et de la prise en compte des variations de taille SLK, nous avons comparé ces deux méthodes à l'algorithme sans contrainte ULK. D'une nature différente, l'intégration de régions d'intérêt RLK n'est pas comparée quantitativement aux autres approches.

Le protocole expérimental

Toutes nos évaluation quantitatives sont effectuées sur des séquences synthétiques. L'utilisation d'images de synthèse permet de contrôler facilement et précisément aussi bien la scène, les capteurs ou le bruit de mesure. La scène utilisée pour l'évaluation se compose d'un plan texturé faisant face aux caméras. Cette scène constitue un cas simple pour toutes les versions de l'algorithme de Lucas & Kanade. Entre deux pas de temps, le plan est déplacé longitudinalement (en profondeur) d'une distance ΔZ. Les translations sur cet axe sont les plus difficiles à estimer par des méthodes de suivi basées vision. En effet, les translation selon les axe X et Y engendrent la même translation dans les images, alors qu'une variation de profondeur engendre une translation, une variation de la disparité et de la taille apparente. Pour mettre en valeur la gestion de ces trois aspects par notre approche, nous avons donc choisi ce mouvement. La figure 6.18 schématise le déplacement du plan texturé face aux caméras. En position initiale, il se projette dans un carré de 512×512 pixels dans chaque image. Plusieurs variations de profondeur ΔZ sont testées. L'ajout d'un bruit blanc gaussien aux images permet de tester la robustesse des algorithmes face au bruit de mesure.

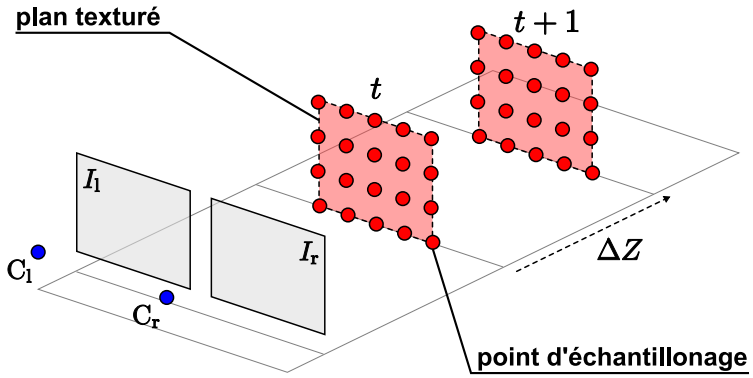


FIG. 6.18: Scène d'évaluation : un plan texturé, échantillonné par une grille de points connus, est déplacé d'une distance ΔZ sur l'axe longitudinal.

La qualité et la performance des différents algorithmes est évaluée sur 400 points « échantillons » répartis régulièrement sur le plan. Une répartition régulière nous permet de sélectionner des points fortement et des points faiblement texturés. Les textures peu contrastées constituant un réel défi pour les méthodes de suivi, et il est intéressant de les intégrer dans l'étude. Dans la totalité des résultats présentés ici, 7 niveaux sont utilisés dans le cadre de l'approche pyramidale, et les points sont systématiquement caractérisés par une fenêtre de 21×21 pixels, et ce, quelque soit le niveau.

La mesure d'erreur

L'erreur de mesure est obtenue en comparant le vecteur vitesse théorique à ceux mesurés par les différentes techniques. Suivant que l'on veuille caractériser une violation de la contrainte épipolaire ou non, plusieurs représentations du vecteur de vitesse sont possibles. La combinaison de deux vecteurs 2-D (utilisée par l'algorithme ULK non-contraint) permet d'exprimer deux rayons optiques ne s'intersectant pas, violant ainsi la contrainte épipolaire. Au contraire, l'utilisation d'un unique vecteur 3-D (comme dans ELK et SLK), garantit par construction cette contrainte. Pour pouvoir exprimer une violation de cette dernière, il est nécessaire d'utiliser deux vecteur 2-D. Soulignons néanmoins que, intégré au sein d'un processus complet de stéréo-vision, le jeu de paramètres sera *in fine* convertit en un point 3-D valide (cf. reconstruction 3-D 2.2.3 p. 28). C'est pour cette raison que nous jugeons superflu de caractériser la violation de la contrainte épipolaire, et nous avons préféré représenter les mesures des vitesses (et des erreurs associées) dans l'espace 3-D. Nous utilisons l'espace 3-D basé image pour présenter nos résultats, car c'est dans cet espace que sont effectué tous les calculs. Les trois axes sont : x, y et d (voir la figure 6.13 et 6.20). Tout comme lors du processus de reconstruction, une mesure violant la contrainte épipolaire est remplacée par la mesure valide la plus « proche ».

Analyse de l'erreur

La précision est généralement caractérisée par la moyenne des erreurs au carré (en anglais Root Mean Square, souvent abrégé *RMS*). Mais la détection et l'élimination des données aberrantes est néanmoins recommandée avant de calculer l'erreur *RMS*. Sur les graphiques de la figure 6.20, on remarque très bien que certaines erreurs de mesure dépassent largement les autres. Ce sont des données aberrantes. En d'autre termes, des points pour lesquelles l'algorithme a totalement échoué. Nous avons décidé de modéliser séparément les échecs. Au final, nous évaluons chaque algorithme suivant deux critères de performance, qui sont :

1. la précision du suivi et
2. le taux d'échecs.

Nous modélisons la distribution des erreurs par une mixture de deux gaussiennes : une gaussienne étroite modélisant la précision de l'algorithme et une gaussienne large dont le poids relatif (par rapport à la gaussienne étroite) modélise le taux d'échecs (voir la figure 6.19).

L'estimation de ces deux gaussiennes est effectuée par l'algorithme *EM* (se référer à l'ouvrage de Meer [Mee04] pour plus de détails). L'erreur *RMS* est déduite des matrices de covariances des deux gaussiennes modélisant l'erreur.

Les résultats

La figure 6.20 montre une représentation 3-D de l'erreur sous forme de nuages de points. Chacun de ces points représente l'erreur 3-D selon (x, y, d) sur un point d'échantillonnage. La colonne de gauche correspond à $\Delta Z = 1$ et celle de droite à $\Delta Z = 5$. Les deux ΔZ sont testés avec les trois versions de l'algorithme : ULK (en haut), ELK (au centre) et SLK (en bas).

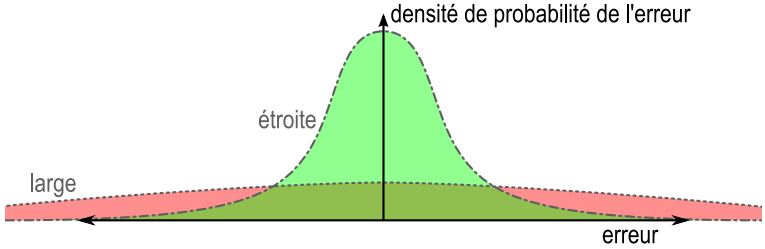


FIG. 6.19: La précision des algorithmes est modélisée par la mixture de deux gaussiennes : une pour la précision (en vert) et une pour le taux d'échecs (en rouge).

Nous constatons d'abord que l'écart de performance entre les différentes versions est d'autant plus marqué que ΔZ est important. Nous remarquons ensuite que le nuage produit par l'algorithme SLK se démarque nettement des deux autres, particulièrement dans le cas d'un grand déplacement. Ce qui signifie que la gestion des variations de taille contribue énormément à la qualité du processus.

La figure 6.21 reprend un plus grand nombre de résultats et les synthétise sous forme statistique. Ces histogrammes reprennent les résultats en mentionnant la proportion d'échec et la précision du suivi, avec la méthode décrite dans la partie précédente. La figure 6.21(a) exprime les performances en fonction de 5 variations de profondeur différentes. La figure 6.21(b) donne les mêmes critères, mais en fonction de la variance du bruit blanc gaussien ajouté aux observations.

Le résultat présenté par le graphique de la figure 6.21(a) montre que l'approche ELK utilisant des coordonnées *3-D basées image* n'améliore pas la précision du suivi (*RMS inliers*). Par contre, le pourcentage d'échecs (*% outliers*) passe de 50 à 40 dans le pire des cas. Cette constatation s'explique par le fait que la contrainte 3-D est intrinsèquement prise en compte dans la formulation du problème. L'espace des solutions ainsi réduit élimine toutes les solutions mal conditionnées. Comme attendu, la seconde approche améliore énormement la précision de suivi (*RMS inliers*) et le pourcentage d'échecs (*% outliers*). Le gain est d'autant plus important que la variation de taille apparente est importante. Notons cependant que le mouvement utilisé (ΔZ) est celui permettant de mettre en valeur les différences enrte ULK, ELK et SLK. La différence de précision entre deux algorithmes aurait été beaucoup plus faible avec une translation selon l'axe X ou Y, puisque pour ces mouvements, la taille apparente ne varie pas.

Le graphique de la figure 6.21(b) présente les mêmes résultats, mais en faisant varier cette fois le bruit de mesure. Le bruit est exprimé par le rapport signal/bruit (*Signal Noise Ratio* en anglais) en unité décibel (*dB*), donné par l'équation suivante :

$$10 \times \log_{10}\left(\frac{S}{N}\right), \tag{6.37}$$

où S est l'amplitude maximale du signal et N l'amplitude du bruit. Lorsque le bruit est de moyenne nulle, l'amplitude est égale à l'écart type. Il est intéres-

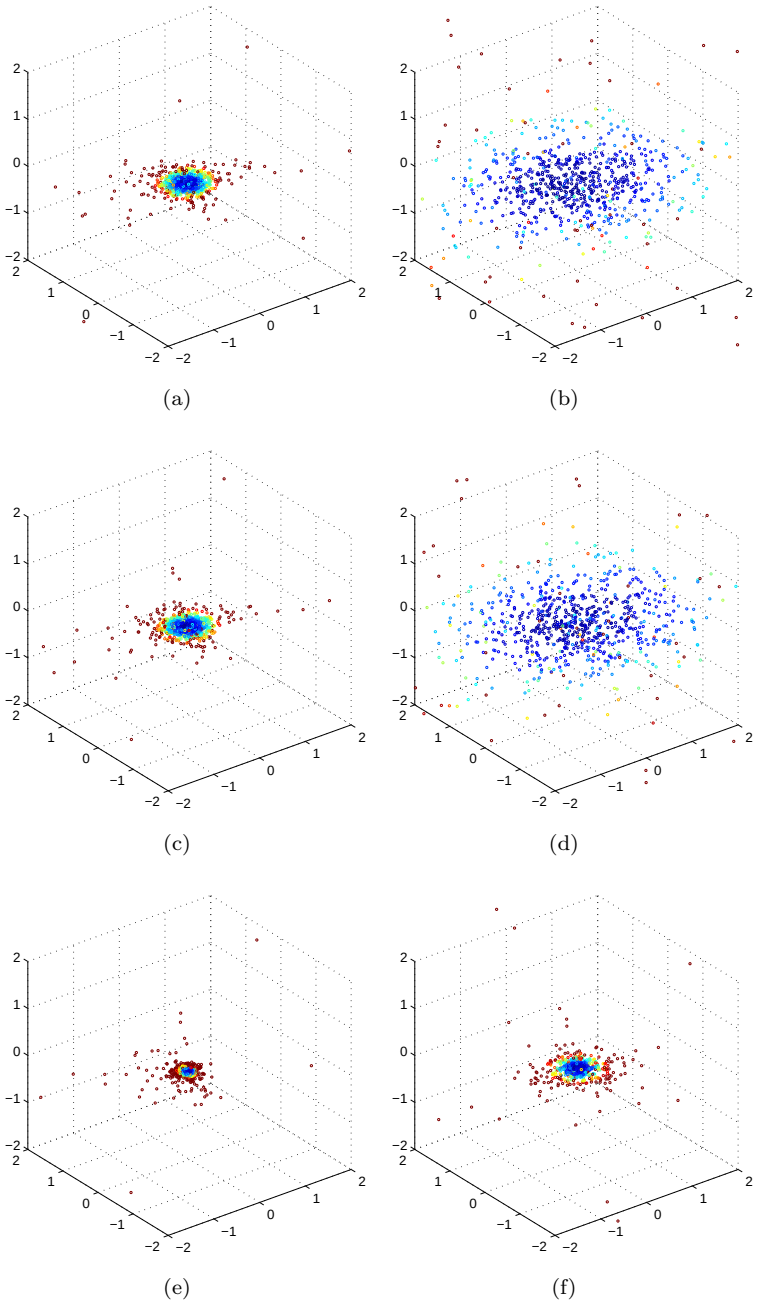
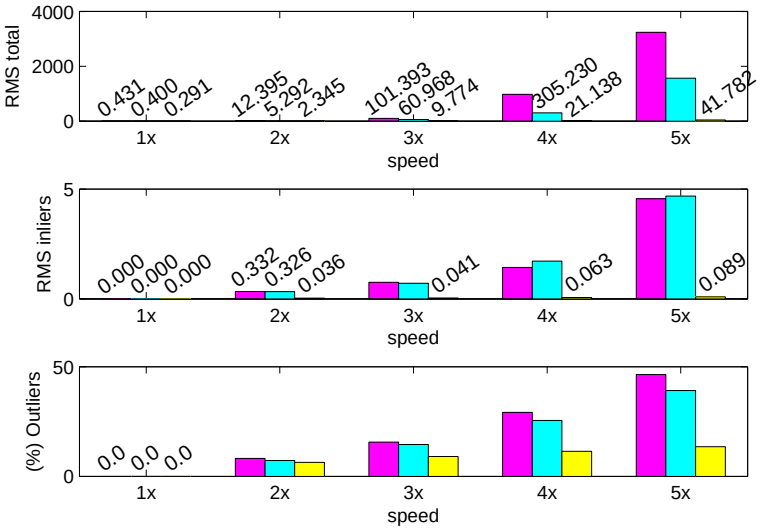
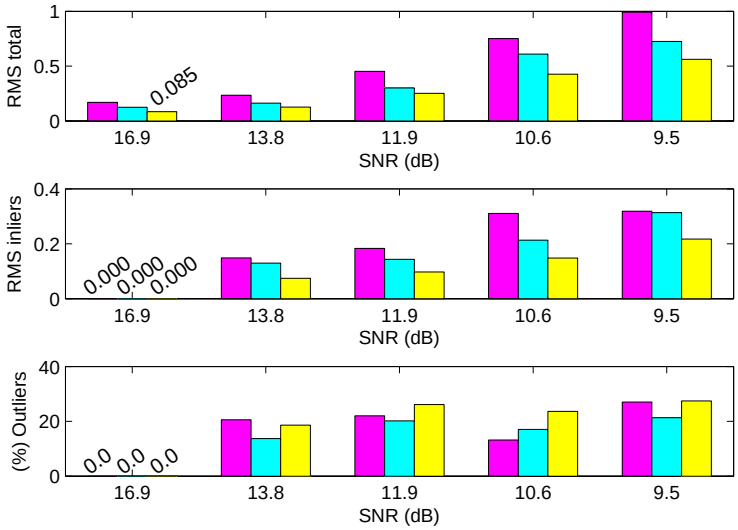


FIG. 6.20: Répartition de l'erreur en (x, y, d) pixel. $\Delta Z = 1$ pour (a),(c) et (e). $\Delta Z = 5$ pour (b),(d) et (f), pour les trois version de l'algorithme : (a,b) non-conraint, (c,d) intégrant la contrainte épipolaire et (e,f) avec les variations de taille apparente.



(a) Erreur en fonction de ΔZ, SNR=∞.



(b) Erreur en fonction du bruit, $\Delta Z = 1\times$

FIG. 6.21: Comparaison des trois algorithmes listés dans le tableau 6.1 en fonction de (a) ΔZ ou (b) du rapport signal bruit (en décibel). Les colonnes de gauche (magenta) correspondent à ULK, celles du centre (cyan) à ELK et celles de droite (jaune) à SLK.

sant de noter que la présence de bruit, même important, affecte modérément la qualité du suivi.

6.3.2 Évaluation qualitative

L'objectif essentiel de cette partie est de démontrer que les extensions que nous proposons conservent leur performances sur une séquence réelle. En l'absence de capteur de « référence » sur le véhicule d'essai, nous ne somme malheureusement pas en mesure de fournir une évaluation quantitative des performances en conditions réelles. Par conséquent, les résultats que nous présentons dans cette partie sont purement qualitatifs. Nous profitons de cette séquence pour illustrer l'apport des régions d'intérêts RLK.

La séquence que nous avons choisie (voir la figure 6.22) montre quelques images de la poursuite d'un véhicule cible. Au cours de la séquence, la cible s'éloigne jusqu'à ne mesurer plus que quelques pixels dans l'image. Le véhicule porteur rattrape ensuite la cible, puis la dépasse. Une telle séquence constitue un vrai défit pour les techniques de suivi par vision. En effet, les variations importantes de distance engendrent d'importantes variations de taille apparente.

La figure 6.22 présente quelques images issues d'une séquence de 440 paires d'images. La colonne de gauche affiche le résultat du suivi effectué sur une région, par la version RLK de notre algorithme. La colonne de droite montre le résultat du suivi effectué avec ULK, ELK et SLK sur 6 points d'échantillonnage de la cible. Les différentes versions du suivi de point sont colorées :

- en magenta pour ULK,
- en cyan pour ELK et
- en jaune pour SLK.

Rappelons que, même si la figure 6.22 représente le suivi par une fenêtre 2-D, c'est un suivi 3-D qui est effectué. A tout moment, la distance de la cible et sa vitesse sont mesurés.

Le suivi par point fait apparaître un problème inhérent au suivi par flux optique : la dérive (*drift* en anglais). La dérive se manifeste par un écart entre la cible réelle et le point qui la représente. Cet écart grandit avec le temps, jusqu'a ce que la cible soit complètement perdue, et q'un autre objet soit suivi à sa place. Ce phénomène provient du fait que la cohérence photométrique lie l'apparence de la cible entre les dates t et $t + \Delta_t$, et autorise donc des petites variations de l'apparence. La cible suivie à la date t peut donc avoir une apparence très différente de celle détectée initialement à la date 0. Dans le cas de la figure 6.22, la fenêtre caractérisant un point recouvre le véhicule, mais parfois une partie de la route. Le cas échéant, le suivi donne un résultat intermédiaire, suivant à la fois la route et la cible, jusqu'a dériver entièrement sur la route. Comme le montre la colonne de gauche, lorsque la taille de la cible est spécifiée par une région d'intérêt, le phénomène de dérive ne se produit plus.

6.4 Conclusion

Dans ce chapitre, nous avons présenté une nouvelle approche de mesure de déplacements relatifs par stéréo-vision dans l'espace 3-D basé image. D'un coté, la stéréoscopie permet de percevoir les 3 dimensions de cet espace et de l'autre, les flux optiques permettent de mesurer le déplacement. Alors que beaucoup les

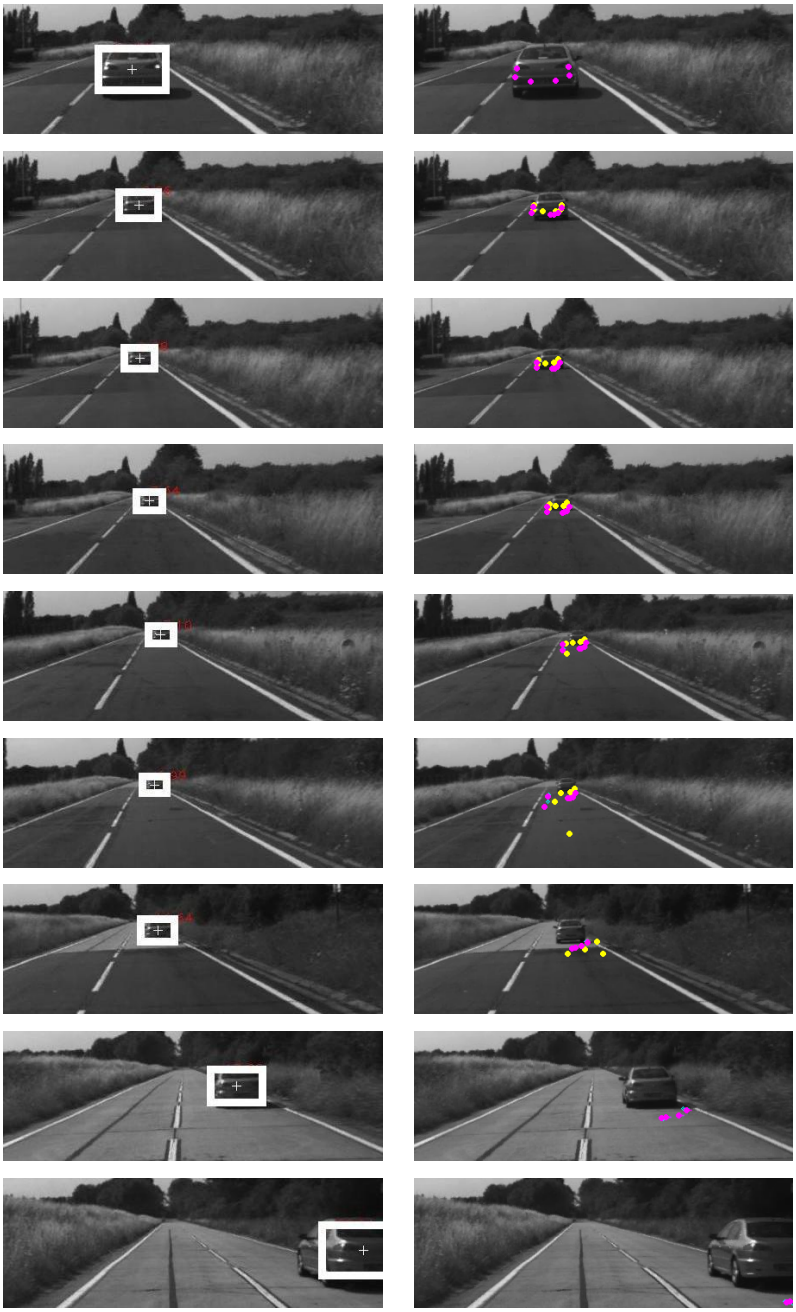


FIG. 6.22: Le véhicule est manuellement sélectionné sur la première paire d'images, puis suivi sur plus de 440 acquisitions. Le suivi de points dérive progressivement vers la route (colonne de droite) alors que le suivi d'une région d'intérêt reste stable sur l'ensemble de la séquence (colonne de gauche).

approches utilisant ces deux principes le font séparément, nous les avons réuni dans formulation unique, qui étend l'algorithme monoculaire de Lucas & Kanade, en l'adaptant au cas d'un système binoculaire calibré. Note adaptation se compose de 3 volets, qui sont l'intégration de la contrainte épipolaire, du changement de taille apparente et des régions d'intérêts. Grâce à cela, le flux de scène estimé est plus précis que celui obtenu par la fusion de deux flux optiques. La méthode s'applique à n'importe quel système de stéréo-vision, à condition que les images soient rectifiées sous certaines conditions (focales égales et disparité nulle à l'infini). La gestion de fenêtres de taille définit permet d'adapter l'algorithme au suivi de véhicule. Les résultats obtenus sur des séquences synthétiques et réelles montrent les gains de précision et de robustesse apportés par la fusion de la stéré-vision et du flux optique.

Chapitre 7

Conclusions et perspectives

La très récente percée des systèmes « intelligents » sur des véhicules de série du marché (comme l'ACC), démontre l'intérêt que portent les acteurs de l'industrie automobile à ce type de système. L'intégration de tels systèmes nécessite la maîtrise des technologies de perception de l'environnement du véhicule. Ce dernier point demeure un point difficile, pour lequel la technologie ne cesse d'évoluer. La complexité de la tâche de perception provient essentiellement de la grande diversité des scénarios routiers, mais aussi des conditions extrêmes auxquelles le véhicule est confronté au cours de sa vie. Ces aspects font de la tache de perception un sujet de recherche toujours étudié, aussi bien par les acteurs industriels qu'académiques. Les systèmes actuellement sur le marché se basent sur des technologies dites « actives » (comme les systèmes basés sur RADAR ou LIDAR, se référer à la partie 1.2 p. 1). Les premiers systèmes basés sur un (ou plusieurs) RADAR se voient de plus en plus concurrencés par ceux basés sur un LIDAR. L'utilisation des technologies basées sur la vision reste anecdotique. Pourtant, ces capteurs offrent des avantages technologiques et économiques. Demain, ni la puissance de calcul, ni la qualité des capteurs n'empêcheront de tels systèmes d'équiper les véhicules de série. Cependant, d'autres difficultés restent à surmonter. Identifier, comprendre et tenter de résoudre ces difficultés est l'objectif premier de cette thèse.

7.1 Le bilan

Le propos fondamental de cette thèse est l'étude des problèmes liés à l'intégration d'un système stéréoscopique sur un véhicule de série. Pour concrétiser les exigences envers un tel système nous avons pris le cas d'une application de suivi de véhicule comme exemple. Pour cette application, nous avons implanté et testé les 5 fonctions requises, illustrées par la figure 7.1. Ces fonctions à maîtriser sont :

1. le calibrage,
2. la rectification,
3. la mise en correspondance stéréoscopique,
4. la segmentation d'obstacles et
5. la mesure de déplacement (vitesse) des obstacles.

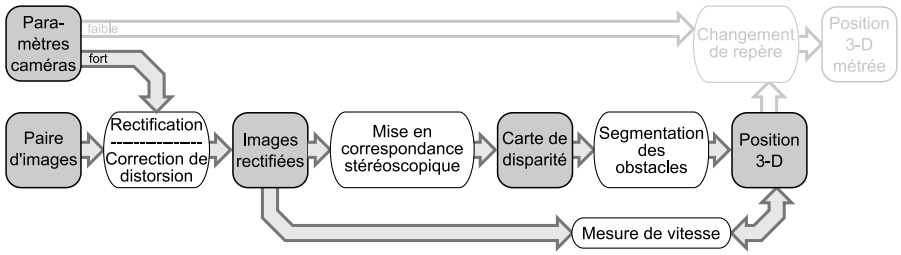


FIG. 7.1: Implantation d'un système de détection et de suivi de cible pour une application de suivi de véhicule à basse vitesse.

Chacune de ces 5 fonctions a fait l'objet d'un chapitre. Pour chacune d'elles, nous avons évalué les performances des méthodes de l'état de l'art et/ou proposé une méthode originale (plus ou moins selon les fonctions) que nous avons implantée et validée.

7.2 Contributions

Parmi ces 5 briques fonctionnelles, nous avons identifié trois difficultés majeures à relever. Ces trois difficultés concernent le calibrage, la rectification, et la mesure de vitesse. La résolution de chacune de ces difficultés fait l'objet d'une contribution.

La première contribution concerne le calibrage. L'utilisation d'une tête stéréoscopique implique qu'elle ai été calibrée au préalable. Si le calibrage n'est plus correct (par exemple à cause des variations importantes de température ou de la déformation de la caisse) alors le système risque de devenir inopérant, ou pire, de fournir de fausses informations. Dans un grand nombre de publications traitant de la stéréo-vision pour des applications automobiles, ce point est considéré comme acquis, alors que nous avons mis en évidence dans le chapitre 2 que, le calibrage d'un système embarqué à bord d'un véhicule de série nécessite une attention particulière. Nous avons présenté une méthode permettant d'évaluer la qualité d'un calibrage. Tous les aspects de la vie du système sont considérés, en commençant par la procédure de calibrage (partie 2.3), puis les éventuelles conséquences d'un « décalibrage » (partie 2.4.1), et enfin la possibilité de re-calibrer le système en ligne (partie 2.4.5). La méthode que nous proposons se base sur des simulations, ce qui permet d'explorer de multiples configurations et situations. Les résultats présentés permettent d'identifier les principales difficultés auxquelles il faut faire face pour obtenir et maintenir un système stéréoscopique calibré. Ainsi, nous avons mis en avant l'importance de construire une mire de calibrage qui occupe toute la zone de perception du système (partie 2.3.3). D'autre part, il se dégage clairement que certains paramètres des caméras doivent faire l'objet de toutes les attentions. Ainsi, nous avons établi que la vergence des caméras ne doit pas varier au cours du temps sous peine de modifier la perception des distances (partie 2.3.3). Plus problématique, une variation de ce paramètre n'engendre qu'une faible erreur de reprojection (caractérisant la violation de la contrainte épipolaire), ce qui rend la détection du

dysfonctionnement et le calibrage en ligne difficile (partie 2.4.6).

La deuxième contribution concerne la mise en correspondance stéréoscopique (chapitre 4) et l'étape préliminaire de rectification (chapitre 3). Dans ces deux chapitres, nous avons présenté et clarifié le rapport qui existe entre le choix d'une rectification et la qualité des surfaces reconstruites. En effet, comme nous l'avons mis en évidence dans la partie 4.2, le processus de mise en correspondance stéréoscopique par fenêtres de corrélation repose sur l'hypothèse d'une surface coïncidant avec une surface d'iso-disparité (surface se projetant à disparité constante dans les deux images). Nous avons donc caractérisé les surfaces d'iso-disparités en fonction des rectifications. Ainsi, nous montrons qu'en choisissant la rectification qui permet de faire coïncider les iso-disparités avec des surfaces verticales (resp. horizontales) de la scène, la reconstruction atteint une meilleur qualité pour les obstacles frontaux (resp. horizontaux, comme la route). Finalement, dans la partie 4.2, nous montrons qu'utiliser simultanément les deux rectifications (verticales et horizontales) améliore la qualité de la reconstruction, mais surtout, nous permet d'étiqueter chaque point de la scène comme appartenant à une surface verticale ou horizontale. Dans le chapitre 5 nous montrons que cette segmentation fourni une information efficace pour la segmentation des obstacles potentiels.

La troisième contribution concerne la mesure de mouvement 3-D. Certaines applications, comme l'ACC (voir les systèmes d'aide à la conduite dans la partie 1.2), nécessitent une estimation précise de la vitesse relative au véhicule cible. La solution généralement proposée consiste à détecter la cible au cours du temps et à mettre en correspondance les détections successives pour en déduire sa vitesse. Mais ce type d'approche donne une estimation trop imprécise. Nous avons donc développé une algorithme spécifiquement dédié à la mesure de déplacements 3-D relatifs (vitesse) par stéréo-vision. Alors que l'information stéréoscopique et la mesure de mouvement sont généralement calculées séparément, nous combinons efficacement ces deux informations dans une seule formulation. Pour cela, nous étendons au cas stéréoscopique l'algorithme de mesure du flux optique 2-D proposé par Lucas & Kanade [LK81]. Notre extension se compose de 3 volets, qui prennent en compte la contrainte épipolaire (développée dans la partie 6.2.4), les variations de taille apparente (partie 6.2.6), et des régions d'intérêts dans l'image (partie 6.2.7). Les expérimentations sur des séquences stéréoscopiques synthétiques et réelles prouvent le gain apporté par notre approche, puisque dans le pire des cas, la précision atteint 0.1 pixel avec 10% d'échec, contre 5 pixels de précision avec 40% d'échec pour l'algorithme *LK* standard effectué indépendamment sur les deux flux d'images.

7.3 Perspectives

Les travaux présentés dans cette thèse explorent les 5 fonctionnalités de la figure 7.1. Le travail exploratoire est plus ou moins abouti selon les fonctionnalités, et pour certaines d'entre elles, nos travaux peuvent être améliorés.

Le premier point à améliorer concerne les expérimentations sur le calibrage. Même si nous avons définit une méthodologie d'évaluation basée sur des simulations, il manque une étape d'expérimentations en conditions réelles, pour vérifier que le protocole de calibrage remplit les spécifications attendues. D'autre part, ce calibrage nous permettrait d'avoir une reconstruction 3-D métrique (et

non basée image), indispensable pour quantifier les performances générales du système.

Le deuxième point concerne la segmentation des obstacles. Nous proposons une segmentation basée sur l'orientation locale de la surface. Cette segmentation donne d'excellent résultats sur des séquences synthétiques, mais les imperfections de notre modèle de correction de distortion nous ont empêché de valider cet algorithme sur des séquences réelles. Lorsque la correction n'est pas parfaite, la corrélation est approximative et n'est plus assez discriminante pour déterminer laquelle des rectifications convient le mieux. Le problème se manifeste particulièrement en l'absence de texture dans l'image. Comme la segmentation est évaluée indépendamment pour chaque pixel, il en résulte une carte de segmentation très imparfaite, où les régions peu texturées sont étiquetées aléatoirement. Ce problème ressemble aux problèmes de segmentation de fond, et nous pourrions tenter de le résoudre par les mêmes approches. Kolmogorov *et al.* [KCB+05] et Sun *et al.* [SZTS06] proposent de caractériser le problème sous la forme d'une énergie globale à minimiser. La segmentation est ainsi améliorée par la prise en compte d'une cohérence spatiale, permettant de propager l'information vers les régions où elle fait défaut. La minimisation globale est possible en temps réel par des approches de coupe de graphe (*graph cut* en anglais).

Le troisième et dernier point concerne le processus de mesure de déplacement 3-D. Le calibrage en conditions réelles nous a fait défaut, et nous n'avons pas pu comparer notre système au LIDAR, réputé précis. Notre approche, se base sur l'algorithme LK, et comme toutes les approches de suivi par apparence, elle atteint ses limites lorsque la cible change d'apparence. Néanmoins, certains auteurs proposent d'étendre LK pour gérer les changements d'illuminations [Bar06], ou encore les variations de couleurs [DTT98]. Il serait intéressant de fusionner ces extensions et la nôtre.

Chapitre 8

Annexe

8.1 Disparité et coût sous-pixelique

La fonction parabole est définie :

$$f(x) = ax^2 + bx + c$$

avec a, b et c les trois paramètres, et

$$C(x) = f(x - d_0) \qquad \text{et} \qquad f(x) = C(x + d_0),$$

où C est la fonction de corrélation pour une ordonnée donnée (les abscisses sont négligées). Nous savons que la fonction passe par les trois point $x \in \{-1, 0, 1\}$. et la corrélation associée. Nous en déduisons :

$$\begin{cases} f(0) = c \\ f(+1) = a + b + c \\ f(-1) = a - b + c \end{cases}$$

ce qui donne

$$\begin{cases} a = \dfrac{f(+1) + f(-1)}{2} - f(0) \\ b = \dfrac{f(+1) - f(-1)}{2} \\ c = f(0) \end{cases}$$

Le minimum $d_0 + \delta_x$ correspond au x qui annule la dérivée de f :

$$\begin{aligned} C'(d_0 + \delta_x) = f'(\delta_x) &= 0 \\ 2a(\delta_x) + b &= 0 \\ \delta_x &= -\frac{b}{2a} \\ \delta_x &= -\frac{1}{2}\frac{f(+1) - f(-1)}{f(+1) + f(-1) - 2f(0)} \\ \delta_x &= -\frac{1}{2}\frac{C(d_0 + 1) - C(d_0 - 1)}{C(d_0 + 1) + C(d_0 - 1) - 2C(d_0)} \end{aligned}$$

La corrélation, elle aussi, peut être affinée. En effet, elle peut être interpolée pour la valeur de disparité non entière $d_0 + \delta_x$:

$$C(d_0 + \delta_x) = f(\delta_x) = a{\delta_x}^2 + b\delta_x + c$$

ce qui, en fonction de $f(-1)$, $f(0)$ et $f(1)$, donne :

$$-\frac{1}{8}\frac{f(1)^2 - 2f(1)f(-1) + f(-1)^2 - 8f(0)f(1) - 8f(0)f(-1) + 16f(0)^2}{f(1) + f(-1) + 2f(0)}.$$

Rappelons que dans le cadre d'une implantation optimisée, ce calcul est appliqué à tous les pixels d'un coup. Pour limité les allocations de mémoires, nous reformulons le calcul de la manière suivante :

$$C(d_0 + \delta_x) = -\frac{((f(1) - 2)f(1) + f(-1))f(-1)}{8 \times shape} + f(0).$$
$$\text{avec } shape = f(1) + f(-1) + 2f(0).$$

Sachant que *shape* est déjà calculé comme critère de qualité.

8.2 Algorithmes *LK* étendus à la stéréoscopie

8.2.1 Version *ELK*

Cette section comporte 5 algorithmes (9, 10, 11, 12 et 13) décrivant en détails l'implantation de la version ELK (se référer au tableau 6.1 p. 145 pour le détail des différentes versions) de notre algorithme de mesure de déplacement 3-D. Pour plus de commentaires sur certaines opérations se référer à l'implantation de Bouguet [Bou00] et à l'algorithme 5 (p. 133), la décrivant.

Algorithme 9 : Extension de l'algorithme *LK* pour le suivi de points 3-D.

```
pour niveau l ← level to 0 faire
    // Calcul du facteur multiplicateur au niveau l de la pyramide
    s ← FacteurEchelle(l);
    // Mise a l'échelle de l'image à la date t
    (I_l, I_r) ← Pyramide( PaireImages(t), s);
    // Mise a l'échelle de l'image à la date t + Δ_t
    (J_l, J_r) ← Pyramide( PaireImages(t + Δ_t), s);
    v ← Point(i,t + Δ_t);
    // Boucle parallélisable
    pour tous les points i faire
        // i est le point à suivre
        si Status(i) = Mauvais alors ne pas traiter ce point;
        // u et v sont les vecteurs d'état de i aux dates t et t + Δ_t.
        u ← Point(i,t) ×s;
        // Copie ou mise à jour de v.
        si l = level alors v ← v ×s;
        sinon v ← 2 × v;
        // Extraction de la fenêtre autour de i à la date t.
        (patchIl,patchIr) ← SousRegion((I_l, I_r),u,taille);
        // et calcul de son gradient dans les deux directions.
        (patchIlx,patchIrx) ← GradientX( patchIl,patchIr);
        (patchIly,patchIry) ← GradientY( patchIl,patchIr);
        // Liste des pixels à l'intérieur de la fenêtre et de l'image
        v ← ItérationLK_stereo( patchIl, patchIr, patchIlx, patchIrx, patchIly,
        patchIry, J_l, J_r, v, taille);
        Point(i,t) ← v;
    fin
fin
```

Algorithme 10 : *ItérationLK_stereo* de l'algorithme 9.

Data : patchI, patchIx, patchIy, J, v, taille
Result : v
begin
  $(p_{li}, p_{ri}) \leftarrow$ `ListePixels(`I_l, I_r,u,*taille*`)`;
  tant que *iteration* $j \leq$ *maxIteration* **faire**
    (patchJl,patchJr) $\leftarrow$ `SousRegion(`(J_l, J_r),v,*taille*`)`;
    $(p_{lj}, p_{rj}) \leftarrow$ `ListePixels(`*patchJl*,*patchJr*`)`;
    $(p_l, p_r) \leftarrow (p_l i, p_r i) \cap (p_l j, p_r j)$;
    $b_x \leftarrow b_y \leftarrow b_d \leftarrow 0$;
    si (p_l, p_r) *inchangé* **alors**
      $b_x \leftarrow b_y \leftarrow b_d \leftarrow 0$;
      pour tous les *pixel* p_r **faire**
        $t \leftarrow$ patchIr(p_r) - patchJr(p_r);
        $(b_x, b_y) \leftarrow$ `MiseAJourB(` b_x, b_y, t, *patchIrx*(p_r), *patchIry*(p_r)`)`;
      fin
      $b_d \leftarrow b_x$;
      pour tous les *pixel* p_l **faire**
        $t \leftarrow$ patchIl(p_l) - patchJl(p_l);
        $(b_x, b_y) \leftarrow$ `MiseAJourB(` b_x, b_y, t, *patchIlx*(p), *patchIly*(p)`)`;
      fin
    sinon
      $H_{11} \leftarrow H_{12} \leftarrow H_{13} \leftarrow H_{22} \leftarrow H_{23} \leftarrow H_{33} \leftarrow 0$;
      pour tous les *pixel* p_r **faire**
        $t \leftarrow$ patchIr(p_r) - patchJr(p_r);
        `// Se référer aux algorithmes 11 et 12 pour plus de détails`
        $(b_x, b_y) \leftarrow$ `MiseAJourB(` b_x, b_y, t, *patchIrx*(p_r), *patchIry*(p_r)`)`;
        $(H_{11}, H_{12}, H_{22}) \leftarrow$ `MiseAJourH(` H_{11}, H_{12}, H_{22}, t, *patchIlx*(p_r), *patchIly*(p_r)`)`;
      fin
      $b_d \leftarrow b_x$;
      $H_{13} \leftarrow H_{33} \leftarrow H_{11}$;
      $H_{23} \leftarrow H_{12}$;
      pour tous les *pixel* p_l **faire**
        $t \leftarrow$ patchIl(p_l) - patchJl(p_l);
        $(b_x, b_y) \leftarrow$ `MiseAJourB(` b_x, b_y, t, *patchIlx*(p_l), *patchIly*(p_l)`)`;
        $(H_{11}, H_{12}, H_{22}) \leftarrow$ `MiseAJourH(` H_{11}, H_{12}, H_{22}, t, *patchIlx*(p_l), *patchIly*(p_l)`)`;
      fin
    fin
    déterminant $\leftarrow$
    $H_{11} \times H_{22} \times H_{33} - H_{11} \times H_{23}^2 - H_{12}^2 \times H_{33} + 2H_{12} \times H_{13} \times H_{23} - H_{12}^2 \times H_{22}$
    ;
    si *déterminant* $< \epsilon_a$ **alors**
      `Status(`i`)` = **Mauvais** ;
      return ;
    déterminant $\leftarrow$ 1/ déterminant ;
    $(H_{11}^{-1}, H_{12}^{-1}, H_{13}^{-1}, H_{22}^{-1}, H_{23}^{-1}, H_{33}^{-1}) \leftarrow$
    `Inversion(`$H_{11}, H_{12}, H_{13}, H_{22}, H_{23}, H_{33}$, *déterminant*`)` ;
    $\Delta_x \leftarrow H_{11}^{-1} \times b_x + H_{12}^{-1} \times b_y + H_{13}^{-1} \times b_d$;
    $\Delta_y \leftarrow H_{12}^{-1} \times b_x + H_{22}^{-1} \times b_y + H_{23}^{-1} \times b_d$;
    $\Delta_d \leftarrow H_{13}^{-1} \times b_x + H_{23}^{-1} \times b_y + H_{33}^{-1} \times b_d$;
    $v \leftarrow v + (\Delta_x, \Delta_y, \Delta_d)$;
    si $\Delta_x^2 + \Delta_y^2 + \Delta_d^2 < \epsilon_b$ **alors return** v ;
  fin
  return v;
end

Algorithme 11 : $MiseAJourB$ de l'algorithme 10.

Data : b_x, b_y, t, p, patchIx(p), patchIy(p)
Result : b_x, b_y
begin
  $b_x \leftarrow b_x + t\times$ patchIx(p) ;
  $b_y \leftarrow b_y + t\times$ patchIy(p) ;
  return b_x, b_y;
end

Algorithme 12 : $MiseAJourH$ de l'algorithme 10.

Data : H_{11}, H_{12}, H_{22}, t, patchIx(p), patchIy(p)
Result : H_{11}, H_{12}, H_{22}
begin
  $H_{11} \leftarrow H_{11}$ + patchIx(p) $\times$ patchIx(p) ;
  $H_{12} \leftarrow H_{12}$ + patchIx(p) $\times$ patchIy(p) ;
  $H_{22} \leftarrow H_{22}$ + patchIy(p) $\times$ patchIy(p) ;
end

Algorithme 13 : $Inversion$ optimisée pour la matrice H symétrique dans l'algorithme 10.

Data : $H_{11}, H_{12}, H_{13}, H_{22}, H_{23}, H_{33}, D$
Result : $H_{11}^{-1}, H_{12}^{-1}, H_{13}^{-1}, H_{22}^{-1}, H_{23}^{-1}, H_{33}^{-1}$
begin
  $H_{11}^{-1} \leftarrow (H_{22} \times H_{33} - H_{23} \times H_{23}) \times D$;
  $H_{12}^{-1} \leftarrow (H_{13} \times H_{23} - H_{12} \times H_{33}) \times D$;
  $H_{13}^{-1} \leftarrow (H_{12} \times H_{23} - H_{13} \times H_{22}) \times D$;
  $H_{22}^{-1} \leftarrow (H_{11} \times H_{33} - H_{13} \times H_{13}) \times D$;
  $H_{23}^{-1} \leftarrow (H_{12} \times H_{12} - H_{11} \times H_{23}) \times D$;
  $H_{33}^{-1} \leftarrow (H_{11} \times H_{22} - H_{12} \times H_{12}) \times D$;
end

8.2.2 Version *SLK*

Cette section comporte 4 algorithmes (14, 15, 16 et 17) décrivant en détails l'implantation de la version SLK (se référer au tableau 6.1 p. 145 pour le détail des différentes versions) de notre algorithme de mesure de déplacement 3-D. Pour plus de commentaires sur certaines opérations se référer à l'implantation de Bouguet [Bou00] et à l'algorithme 5 (p. 133), la décrivant.

Algorithme 14 : Extension de l'algorithme *LK* pour le suivi de points 3-D.

```
pour niveau l ← level to 0 faire
    s ← FacteurEchelle(l);
    (I_l, I_r) ← Pyramide( PaireImages(t), s);
    (J_l, J_r) ← Pyramide( PaireImages(t + Δ_t), s);
    v ← Point(i,t + Δ_t);
    pour tous les points i faire
        // i est le point à suivre
        si Status(i) = Mauvais alors ne pas traiter ce point;
        u ← Point(i,t) ×s;
        si l = level alors v ← v ×s;
        sinon v ← 2 × v;
        // d_0^{-1} est l'inverse de la disparité à la date t mise à
           l'échelle de la pyramide
        d_0^{-1} ← 1/(u_d × s) ;
        (patchIl,patchIr) ← SousRegion((I_l, I_r),u,taille);
        // et calcul de son gradient dans les deux directions.
        (patchIlx,patchIrx) ← GradientX( patchIl,patchIr);
        (patchIly,patchIry) ← GradientY( patchIl,patchIr);
        v ← ItérationLK_stereo_echelle( patchIl, patchIr, patchIlx, patchIrx,
        patchIly, patchIry, J_l, J_r, v, d_0^{-1}, taille);
        Point(i,t) ← v;
    fin
fin
```

Algorithme 15 : *ItérationLK_stereo_echelle* de l'algorithme 14.

Data : patchIl, patchIr, patchIlx, patchIly, patchIrx, patchIry, patchJl, patchJr, v,d_0^{-1}, taille
Result : v
begin
  $(p_{li}, p_{ri}) \leftarrow$ `ListePixels(`*patchIl, patchIr*`)`;
  `// M est la variation de disparité`
  $M \leftarrow v_d * d_0^{-}1$;
  tant que *iteration* $j \leq$ *maxIteration* **faire**
    `// L'extraction de la région d'intérêt doit tenir compte du facteur d'agrandissement` M`.`
    (patchJl,patchJr) $\leftarrow$ `SousRegionRedimensionnee(`(J_l, J_r),M,v,*taille*`)`;
    $(p_{lj}, p_{rj}) \leftarrow$ `ListePixels(`*patchJl,patchJr*`)`;
    $(p_l, p_r) \leftarrow (p_l i, p_r i) \cap (p_l j, p_r j)$;
    $b_x \leftarrow b_y \leftarrow b_d \leftarrow 0$;
    si (p_l, p_r) *inchangé* **alors**
      $(b_x, b_y, b_d) \leftarrow$ `MiseAJourB(` b_x, b_y, b_d, *patchIrx*(p_r), *patchIry*(p_r)`)`;
      $(b_x, b_y, b_d) \leftarrow$ `MiseAJourB(` b_x, b_y, b_d, *patchIlx*(p_l), *patchIly*(p_l)`)`;
    sinon
      $(b_x, b_y, b_d) \leftarrow$ `MiseAJourB(` b_x, b_y, b_d, *patchIrx*(p_r), *patchIry*(p_r)`)`;
      $(b_x, b_y, b_d) \leftarrow$ `MiseAJourB(` b_x, b_y, b_d, *patchIlx*(p_l), *patchIly*(p_l)`)`;
      $(H_{11}, H_{12}, H_{13}, H_{22}, H_{23}, H_{33}) \leftarrow$ `MiseAJourH(` H_{11}, H_{12}, H_{13}, H_{22}, H_{23}, H_{33}, *patchIr, patchJr, patchIrx*(p_r), *patchIry*(p_r)`)`;
      $(H_{11}, H_{12}, H_{13}, H_{22}, H_{23}, H_{33}) \leftarrow$ `MiseAJourH(` H_{11}, H_{12}, H_{13}, H_{22}, H_{23}, H_{33}, *patchIl, patchJl, patchIlx*(p_l), *patchIly*(p_l)`)`;
    fin
    déterminant $\leftarrow$
    $H_{11} \times H_{22} \times H_{33} - H_{11} \times H_{23}^2 - H_{12}^2 \times H_{33} + 2H_{12} \times H_{13} \times H_{23} - H_{12}^2 \times H_{22}$
    ;
    si *déterminant* $< \epsilon_a$ **alors**
      `Status(`i`)` = **Mauvais** ;
      Arrêt ;
    déterminant $\leftarrow$ 1/ déterminant ;
    `// la fonction d'inversion reste identique à celle décrite par l'algorithme 13`
    $(H_{11}^{-1}, H_{12}^{-1}, H_{13}^{-1}, H_{22}^{-1}, H_{23}^{-1}, H_{33}^{-1}) \leftarrow$
    `Inversion(`$H_{11}, H_{12}, H_{13}, H_{22}, H_{23}, H_{33}$, *déterminant*`)` ;
    $\Delta_x \leftarrow H_{11}^{-1} \times b_x + H_{12}^{-1} \times b_y + H_{13}^{-1} \times b_d$;
    $\Delta_y \leftarrow H_{12}^{-1} \times b_x + H_{22}^{-1} \times b_y + H_{23}^{-1} \times b_d$;
    $\Delta_d \leftarrow H_{13}^{-1} \times b_x + H_{23}^{-1} \times b_y + H_{33}^{-1} \times b_d$;
    $v \leftarrow v + (\Delta_x, \Delta_y, \Delta_d)$;
    si $\Delta_x^2 + \Delta_y^2 + \Delta_d^2 < \epsilon_b$ **alors Arrêt** ;
  fin
end

Algorithme 16 : $MiseAJourB$ de l'algorithme 15.

Data : b_x, b_y, b_d, p, patchI, patchJ, patchIx, patchIy
Result : b_x, b_y, b_d
begin
  pour tous les *pixel* p **faire**
    $(x, y) \leftarrow$ coord(p);
    $t \leftarrow$ patchI(p) - patchJ(p) ;
    $b_x \leftarrow b_x + t\times$ patchIx(p) ;
    $b_y \leftarrow b_y + t\times$ patchIy(p) ;
    $b_d \leftarrow b_d + t \times d_0^{-1}\times(x\times$patchIx($p$) $+y\times$ patchIy(p));
end

Algorithme 17 : $MiseAJourH$ de l'algorithme 15.

Data : b_x, b_y, b_d, p, patchI, patchJ, patchIx, patchIy
Result : b_x, b_y, b_d
begin
  pour tous les *pixel* p **faire**
    $(x, y) \leftarrow$ coord(p);
    $S \leftarrow d_0^{-1}\times$ ($x\times$patchIx(p) $+y\times$ patchIy(p)) + patchIx(p) ;
    $H_{11} \leftarrow H_{11}$ + patchIx(p) $\times$ patchIx(p) ;
    $H_{12} \leftarrow H_{12}$ + patchIx(p) $\times$ patchIx(p) ;
    $H_{13} \leftarrow H_{13}$ + patchIx(p) $\times S$;
    $H_{22} \leftarrow H_{22}$ + patchIy(p) $\times$ patchIy(p) ;
    $H_{23} \leftarrow H_{23}$ + patchIy(p) $\times S$;
    $H_{33} \leftarrow H_{33} + S \times S$;
end

Bibliographie

[Agu13] F. Aguilonius. *Opticorum libri sex philosophis juxta ac mathemacticis utiles, Antverp : Ex officina Plantiniana, apud Viduam et filios J. Moreti,*. 1613. 63

[AK71] J. M. Allman and J. H. Kaas. Representation of the visual field in the caudal third of the middle temporal gyrus of the owl monkey (aotus trivirgatus). pages 85–105, 1971. 118

[AKI05] Motilal Agrawal, Kurt Konolige, and Luca Iocchi. Real-time detection of independent motion using stereo. pages 672–677, 2005. 108, 109, 120, 121

[AS99] S. Avidan and A. Shashua. Non-rigid parallax for 3D linear motion. volume 2, pages 62–66, 1999. 119

[ATV00] A. Fusiello, E. Trucco, and A. Verri. A compact algorithm for rectification of stereo pairs. *Machine Vision and Applications*, 12(1) :16–22, 2000. 60

[Avi04] Shai Avidan. Support vector tracking. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*. IEEE, 2004. 128, 129

[BAHH92] James R. Bergen, P. Anandan, Keith J. Hanna, and Rajesh Hingorani. Hierarchical model-based motion estimation. In *ECCV*, pages 237–252, 1992. 126, 127

[Bar06] A. Bartoli. Direct Image Regisrtation With Gain and Bias. In *LIMA3D*, Verona, Italy, 2006. 128, 158

[BB97] M. Bertozzi and A. Broggi. GOLD : a parallel real-time stereo vision system for generic obstacle and lane detection. *IEEE Transactions on Image Processing. In press.*, 1997. 70

[BBB05] M. Bertozzi, E. Binelli, and A. Broggi. Stereo vision-based approaches for pedestrian detection. In *Computer Vision and Pattern Recognition, 2005 IEEE Computer Society Conference*, 2005. 104

[BBF98] M. Bertozzi, A. Broggi, and A. Fascioli. Stereo inverse perspective mapping : Theory and applications. In *Image and Vision Computing Journal*, pages 585–590, 1998. 107

[BBF01] Massimo Bertozzi, Alberto Broggi, and Alessandra Fascioli. Self-calibration of a stereo vision system for automotive applications. In *Procs. IEEE Intl. Conf. on Robotics and Automation*, volume 4, pages 3698–3703, Seoul, Korea, May 2001. 52

[BBFL02] M. Bertozzi, A. Broggi, A. Fascioli, and P. Lombardi. Vision-based pedestrian detection : will ants help. In *Procedings of IEEE Intelligent Vehicles Symposium*, 2002. 104

[BBFN00] M. Bertozzi, A. Broggi, A. Fascioli, and S. Nichele. Stereo vision-based vehicle detection. In *IEEE Intelligent Vehicles Symposium*, pages 39–44, Detroit (MI), USA, October 2000. 121, 122

[Bey92] H.A. Beyer. Accurate calibration of ccd cameras. In *Proceedings of Conference on Computer Vision and Pattern Recognition CVPR92*, pages 96–101, Urbana-Champaign, USA, 1992. 11, 14, 15, 31, 39

[BFSB92] J. L. Barron, D. J. Beauchemin Fleet, S.S., and T. A. Burkitt. Performance of optical flow techniques. In *Proceedings of Conference on Computer Vision and Pattern Recognition CVPR92*, pages 236–242, Urbana-Champaign, USA, 1992. 122

[BHD97] M. Betke, E. Haritaoglu, and L. Davis. Highway scene analysis in hard real time, 1997. 104

[BJ96] Michael J. Black and Allan D. Jepson. Eigentracking : Robust matching and tracking of articulated objects using a view-based representation. In *ECCV (1)*, pages 329–342, 1996. 126, 127

[Bou] J.Y. Bouguet. *Camera Calibration Toolbox for Matlab.* http ://www.vision.caltech.edu/bouguetj. 12, 36

[Bou00] Jean-Yves Bouguet. Pyramidal implementation of the Lucas-Kanade feature tracker. Technical report, Intel Corp., Microprocessor Research Labs, 2000. 126, 129, 133, 142, 143, 144, 161, 164

[BPCL06] Christophe Braillon, C. Pradalier, J. Crowley, and Christian Laugier. Real-time moving obstacle detection using optical flow models. In *Proc. of the IEEE Intelligent Vehicle Symp.*, pages 466–471, Tokyo (JP), June 2006. IEEE. 106

[Bro71] D.C. Brown. Accurate calibration of ccd-cameras. In *Photogrammetric Engineering*, pages 9855–866, 1971. 14, 15, 31

[CDA07] Community database on accidents on the roads in europe, 2007. `http://europa.eu.int/comm/transport/care/index_en.htm`. 1

[CGBCL06] T. Chateau, V. Gay Belille, F. Chausse, and J.T. Lapreste. Real-time tracking with classifiers. In *WDV06*, pages 218–231, 2006. 129

[CGS+04] N. Cornille, D. Garcia, M.A. Sutton, S.R. McNeill, and J.-J. Orteu. Calibrage d'un meb en vue de la mesure précise de formes et de déformations 3D. In *Colloque Photomécanique*, pages 9855–866, 2004. 16

[CK98] J.P. Costeira and T. Kanade. A multibody factorization method for independently moving objects. In *International Journal of Computer Vision*, volume 29 (3), pages 159–179, 1998. 119

[DD01] David Demirdjian and Trevor Darrell. Motion estimation from disparity images. pages 213–218, 2001. 138

[Dev97] F. Devernay. *Vision stéréoscopique et propriétés différentielles des surfaces.* Thèse de doctorat, Institut National Polytechnique, INP, Grenoble, France, février 1997. 62, 78

[DF94] Frédéric Devernay and Olivier Faugeras. Computing differential properties of 3-D shapes from stereoscopic images without 3-D models. In *Proc. CVPR*, pages 208–213, Seattle, WA, June 1994. IEEE Comp.Soc. 87

[DHS02] T. Dang, C. Hoffmann, and C. Stiller. Fusing optical flow and stereo disparity for object tracking. In *IEEE Transactions on intelligent transportation systems*, pages 112–117, 2002. 122

[DMG06] Frederic Devernay, Diana Mateus, and Matthieu Guilbert. Multi-camera scene flow by tracking 3-d points and surfels. In *CVPR (2)*, pages 2203–2212. IEEE Computer Society, 2006. 122, 127, 128

[DRR03] James Davis, Ravi Ramamoothi, and Szymon Rusinkiewicz. Space-time stereo : A unifying framework for depth from triangulation. In *In Proceedings of IEEE Computer Vision and Pattern Recognition*, 2003. 122

[DTT98] Frank Dellaert, Chuck Thorpe, and Sebastian Thrun. Super-resolved texture tracking of planar surface patches. In *IEEE/RSJ Int'l Conf. on Intelligent Robotic Systems*, October 1998. 128, 158

[DW06] B. Jähne D. Withopf. Learning algorithm for real-time vehicle tracking. In *Proc. of the IEEE Intelligent Transportation System Society*, Toronto (Canada), Sept. 2006. IEEE. 129

[ETC98] G. Edwards, C. Taylor, and T. Cootes. Interpreting face images using active appearance models. In *Proc. of the 3rd Int. Conf. on Automatic Face and Gesture Recognition, Nara, Japan.*, pages 300–305, 1998. 126, 127, 128

[Fau92] Olivier D. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig. In *ECCV*, pages 563–578, 1992. 28

[FH02] U. Franke and S Heinrich. Fast obstacle detection for urban traffic situations. In *Intelligent Transportation Systems, IEEE Transactions*, volume 3, pages 173–181, September 2002. 108, 109, 121

[FHZF93] O. Faugeras, B. Hotz, Z. Zhang, and P. Fua. Real time correlation-based stereo : Algorithm, implementation and applications. Rapport de recherche RR-2013, Institut National de Recherche en Informatique et en Automatique, INRIA, août 1993. 80

[FP02] D. A. Forsyth and J. Ponce. *Computer Vision : a modern approach.* Prentice-Hall, 2002. 10, 52, 138

[FWM98] J.M. Ferryman, A.D. Worrall, and S.J. Maybank. Learning enhanced 3D models for vehicle tracking. In *BMVC98*, pages xx–yy, 1998. 119

[Gar01] D. Garcia. *Mesure de formes et de champs de déplacements tridimensionnels par stéréo-corrélation d'images.* PhD thesis, Institut National des Sciences Appliquées, Toulouse, France, 2001. 52

[GBG01] S.B. Gokturk, J.-Y. Bouguet, and R. Grzeszczuk. A data-driven model for monocular face tracking. In *IEEE International Conference on Computer Vision*, pages 701–708, 2001. 120

[GBS05] Itay Gat, Meny Benady, and Amnon Shashua. A monocular vision advance warning system for the automotive aftermarket. In *Intel-*

ligent Vehicle Initiative of the world congress SAE. SAE, 2005. 1, 5

[Gle97] Michael Gleicher. Projective Registration with Difference Decomposition. In *Proceedings of the IEEE Conference in Computer Vision and Pattern Recognition (CVPR)*, pages 331–337. IEEE Computer Society, 1997. 126, 128

[GN01] Michael Grossberg and Shree Nayar. A general imaging model and a method for finding its parameters. In *IEEE International Conference on Computer Vision*, pages 108–115, 2001. 17

[Gou05] M. Gouiffes. *Apports de la Couleur et des Modèles de Réflexion pour l'Extraction et le Suivi de Primitives.* PhD thesis, Université de Poitier, Avril 2005. 127, 128

[Har94] Richard I. Hartley. Projective reconstruction from line correspondences. In *CVPR*, pages 903–907, 1994. 28

[HB96] Gregory D. Hager and Peter N. Belhumeur. Real-time tracking of image regions with changes in geometry and illumination. In *CVPR*, pages 403–, 1996. 125, 126, 127, 128

[Hei02] S. Heinrich. Fast obstacle detection using flow/depth constraint. In *in Proceedings of IEEE Intelligent Vehicles Symposium*, 2002. 108

[HKT+98] U. Handmann, T. Kalinke, C. Tzomakas, M. Werner, and W. von Seelen. An image processing system for driver assistance. In *IV'98, IEEE International Conference on Intelligent Vehicles*, pages 481–486, Stuttgart, Germany, 1998. IEEE. 104

[HLPA04] N. Hautière, R. Labayrade, M. Perrollaz, and D. Aubert. Road scene analysis by stereovision : a robust and quasi-dense approach. In *icarv*, pages 1–6, 2004. 87, 107, 108

[HM98] H. Hattori and A. Maki. Stereo matching with direct surface orientation recovery. In *In Ninth British Machine Vision Conference*, pages 356–366, September 1998. 89

[HPN99] B. Heigl, D. Paulus, and H. Niemann. Tracking Points in Sequences of Color Images. In *Pattern Recognition and Image Understanding*, pages 70–77, Herrsching, Germany, 1999. Infix, Sankt Augustin. 127

[HS04] M.A. Sutton H.W. Schreier, D. Garcia. Advances in light microscope stereo vision. In *Experimental Mechanics*, volume 44, pages 278–288, June 2004. 16

[Hub81] Peter J. Huber. *Robust Statistics.* John Wiley and Sons, 1981. 52

[HUW06] Z. Hu, K. Uchimura, and J. Wang. Moving obstacles extraction with stereo global motion model. In *ICPR06*, pages I : 79–83, 2006. 108

[HZ04] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, March 2004. 28, 29, 33, 69

[Int01] Intel. *Open Source Computer Vision Library Reference Manual*, 2001. 16, 33, 36, 60, 111, 113, 118, 133

[ISK92] Toshio Ito, Tatsua Sakagami, and Shiro Kawakatsu. A real time distance headway measurement method using stereo and optical flow. In *Proc. of the IEEE Intelligent Vehicle Symp.*, Detroit (USA), June 1992. IEEE. 122

[JHG+03] Xu J., Wang H., J.I. Guzman, Ng T.C., Shen J., and Chan C.W. Isodisparity profile processing for real-time 3D obstacle identification. *Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE*, 1 :288–292, 12-15 Oct. 2003. 87, 107

[KCB+05] Vladimir Kolmogorov, Antonio Criminisi, Andrew Blake, Geoff Cross, and Carsten Rother. Bi-layer segmentation of binocular stereo video. In *in Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 407–414, 2005. 158

[KDNT92] D. Koller, K. Daniilidis, H.H. Nagel, and T. Thorhallsson. Model-based object tracking in traffic scenes. In *ECCV92*, pages 437–452, 1992. 119

[KK95] P.J. Kellman and M.K. Kaiser. Extracting object motion during observer motion : Combining constraints from optic flow and binocular disparity. *JOSA-A*, 12(3) :623–625, March 1995. 122

[KLM94] D. Koller, Q.-T. Luong, and J. Malik. Using binocular stereopsis for vision-based vehicle control. In *Proceedings of the Intelligent Vehicles Symposium*, pages 237–242, 1994. 105

[LAA97] Wen-Hung Liao, Shanti J. Aggarwal, and Jake K. Aggarwal. The reconstruction of dynamic 3D structure of biological objects using stereo microscope images. *Mach. Vis. Appl.*, 9(4) :166–178, 1997. 120

[LAT02] R. Labayrade, D. Aubert, and J.-P. Tarel. Real time obstacle detection on non flat road geometry through 'v-disparity' representation. In *Proceedings of IEEE Intelligent Vehicle Symposium*, Versailles, France, 2002. http ://perso.lcpc.fr/tarel.jean-philippe/iv02.html. 106, 107, 110

[LCCG07] B. Leibe, N. Cornelis, K. Cornelis, and L.J. Van Gool. Dynamic 3D scene analysis from a moving vehicle. In *Computer Vision and Pattern Recognition, 2007 IEEE Computer Society Conference*, pages 1–8, 2007. 121

[Lem05] V. Lemonde. *Stéréovision Embarquée sur Véhicule : de l'Auto-Calibrage à la Détection d'Obstacles.* PhD thesis, Institut National des Sciences Appliquées, Toulouse, France, 2005. 52, 53, 87, 106, 107

[LF05] V. Lepetit and P. Fua. Monocular Model-Based 3D Tracking of Rigid Objects : A Survey. *Foundations and Trends in Computer Graphics and Vision*, 1(1) :1–89, 2005. 120

[LK81] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*, pages 674–679, 1981. 89, 118, 128, 157

[LL96] Mengxiang Li and Jean-Marc Lavest. Some aspects of zoom lens camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(11) :1105–1110, 1996. 15, 31

[LTH+05] J. Lou, T.N. Tan, W.M. Hu, H. Yang, and S.J. Maybank. 3-d model-based vehicle tracking. *IP*, 14(10) :1561–1569, October 2005. 119

[LWLW06] Zhenjiang Li, Kunfeng Wang, Li Li, and Fei-Yue Wang. A review on vision-based pedestrian detection for intelligent vehicles. In *IEEE International Conference on Vehicular Electronics and Safety*, pages 57–62, 2006. 104

[LZ99] C. Loop and Z. Zhang. Computing rectifying homographies for stereo vision. pages 125–131, 1999. 62, 69, 139

[MDCG07] Julien Morat, Frédéric Devernay, Sebastien Cornou, and Javier Ibanez Guzman. Evaluation method for automotive stereo-vision systems. In *Proceedings of IEEE Intelligent Vehicles Symposium*, june 2007. 31

[Mee04] Peter Meer. *Robust Techniques for Computer Vision*, chapter 4. IMSC Press Multimedia Series. Prentice Hall, 1st edition, July 2004. 52, 147

[Mil97] Steven Mills. Stereo-motion analysis of image sequences. pages 515–520, December 1997. 122

[MMB+98] R. Mandelbaum, L. McDowell, L. Bogoni, B. Reich, and M. Hansen. Real-time stereo processing, obstacle detection, and terrain estimation from vehicle-mounted stereo cameras. In *Proceedings of the 4th IEEE Workshop on Applications of Computer Vision (WACV98)*, October 1998. 64

[MR01] F. Maraninchi and Y. Rémond. Argos : an automaton-based synchronous language. *Computer Languages*, (27) :61–92, 2001. 40, 105, 109

[MS97] S. Malassiotis and M.G. Strintzis. Model-based joint motion and structure estimation from stereo images. 65(1) :79–94, January 1997. 120

[MSKI06] Shunji Miyahara, Jerry Sielagoski, Anatoli Koulinitch, and Faroog Ibrahim. Target tracking by a single camera based on range-window algorithm and pattern matching. *SAE TECHNICAL PAPER SERIES*, (3-6) :1561–1569, April 2006. 121

[MT93] D. Metaxas and D. Terzopoulos. Shape and nonrigid motion estimation through physics-based synthesis. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 15 (6), pages 580–591, 1993. 119, 120

[Nag87] HH Nagel. On the estimation of optical flow : relations between different approaches and some new results. 33 :299–324, 1987. 126

[NDF+04] S. Nedevschi, R. Danescu, D. Frentiu, T. Marita, F. Oniga, C. Pocol, R. Schmidt, and T. Graf. High accuracy stereo vision system for far distance obstacle detection. In *Intelligent Vehicles Symposium*, pages 292–297, june 2004. 82, 83, 108, 121, 122

[NDM+05] S. Nedevschi, R. Danescu, T. Marita, F. Oniga, C. Pocol, Socol T., R. Schmidt, and T. Graf. Driving environment perception using stereovision. In *Procedeeings of IEEE Intelligent Vehicles Symposium, (IV2005)*, pages 331–336, june 2005. 82

[NDM+07] Sergiu Nedevschi, Radu Danescu, Tiberiu Marita, Florin Oniga, Ciprian Pocol, Stefan Sobol, Corneliu Tomiuc, Cristian Vancea, Marc Michael Meinecke, Thorsten Graf, Thanh Binh To, and Marian Andrzej Obojski. A sensor for urban driving assistance systems based on dense stereovision. In *Procedings of IEEE Intelligent Vehicles Symposium*, pages 279–283, june 2007. 107

[NMO+06] S. Nedevschi, T. Marita, F. Oniga, R. Schmidt, and T. Graf. Camera calibration method for far range stereovision sensors used in vehicles. In *Intelligent Vehicles Symposium, IEEE Transactions*, pages 356–363, june 2006. 40

[NSG+04] Nedevschi, Schmidt, Graf, Danescu, Frentiu, Marita, Oniga, and Pocol. 3D lane detection system based on stereovision. In *Procedings of IEEE Intelligent Transportation Systems Conference*, pages 161–166, 2004. 107

[NTH+07] Hiroaki Nakai, Nobuyuki Takeda, Hiroshi Hattori, Yasukazu Okamoto, and Kazunori Onoguchi. A practical stereo scheme for obstacle detection in automotive use. In *Proc. of the IEEE Intelligent Vehicle Symp.*, Istanbul (Turkey), June 2007. IEEE. 89, 106

[NVMG06] S. Nedevschi, C. Vancea, T. Marita, and T. Graf. On-line calibration method for stereovision systems used in vehicle applications. In *Intelligent Transportation Systems, IEEE Transactions*, pages 957–962, sep 2006. 52

[OEC] Organization for economic cooperation & development. http ://www.oecd.org. 1

[PB07] M. Perriollat and A. Bartoli. A quasi-minimal model for paper-like surfaces. In *Computer Vision and Pattern Recognition, CVPR apos 07, New Orleans*, volume 2, pages 17–22, june 2007. 127

[PH91] P. Pentland and B. Horowitz. Recovery of nonrigid motion and structure. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 13 (7), pages 730–742, 1991. 119

[PKF07] Jean-Philippe Pons, Renaud Keriven, and Olivier Faugeras. Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. *International Journal of Computer Vision*, 72(2) :179–193, apr 2007. 121

[PKG99] Marc Pollefeys, Reinhard Koch, and Luc J. Van Gool. A simple and efficient rectification method for general motion. In *IEEE International Conference on Computer Vision*, pages 496–501, 1999. 58, 139

[PS04] Marc Pollefeys and Sudipta N. Sinha. Iso-disparity surfaces for general stereo configurations. In *ECCV (3)*, pages 509–520, 2004. 64

[RFG07] Clemens Rabe, Uwe Franke, and Stefan Gehrig. Fast detection of moving objects in complex scenarios. In *Proc. of the IEEE Intelli-*

gent Vehicle Symp., Istanbul (Turkey), June 2007. IEEE. 108, 121, 122

[RH94] L. Robert and Martial Hebert. Deriving orientation cues from stereo images. In *in Proceedings of European Conference on Computer Vision*, May 1994. 88, 93, 107

[RSL05] Srikumar Ramalingam, Peter Sturm, and Suresh Lodha. Towards complete generic camera calibration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego, California*, volume 1, pages 1093–1098, jun 2005. 17

[SAH07] Nicolas Soquet, Didier Aubert, and Nicolas Hautière. Road segmentation supervised by an extended v-disparity algorithm for autonomous navigation. In *proceedngs of the IEEE Intelligent Vehicles Symposium*, 2007. 107

[SAP01] J. Salvi, X. Armangue, and J. Pages. A survey addressing the fundamental matrix estimation problem. In *ICIP01*, pages II : 209–212, 2001. 28

[SC94] R. Szeliski and J. Coughlan. Hierarchical spline-based registration. In *CVPR*, pages 194–201, 1994. 126, 128

[SI97] Stan Sclaroff and John Isidoro. Active blobs. In *Proceedings of the IEEE Conference in Computer Vision and Pattern Recognition (CVPR)*, number 1997-008. IEEE Computer Society, 5, 1997. 127, 128

[SL97] A. Bainbridge Smith, , and R.G. Lane. Determining optical flow using a differential method. *IVC*, 15 :11–22, 1997. 126

[SMBD02] Z. Sun, R. Miller, G. Bebis, and D. DiMeo. A real-time precrash vehicle detection system. In *Proceedings of the 2002 IEEE Workshop on Applications of Computer Vision*, Orlando, FL, Dec. 2002. 104

[SMS03] Gideon P. Stein, Ofer Mano, and Ammon Shashua. Vision-based acc with a single camera : Bounds on range and range rate accuracy. In *Proc. of the IEEE Intelligent Vehicle Symp.* IEEE, 2003. 119

[SS02] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1-3) :7–42, 2002. http ://vision.middlebury.edu/stereo/. 75, 78, 84

[SSP94] Y. Q. SHI, C. Q. SHU, and J. N. PAN. Unified optical flow field approach to motion analysis from a sequence of stereo images. *Pattern recognition*, 27 (12) :1577–1590, 1994. 120, 122

[ST94] J. Shi and C. Tomasi. Good features to track. In *CVPR*, 1994. 124, 127

[SZB99] K. Sobottka, P. Zuber, and H. Bunke. Shape-based template matching for robust obstacle tracking in low-resolution range image sequences. In *Advanced Concepts for Intelligent Vision Systems (ACIVS)*, 1999. 121

[Sze94] R. Szeliski. Image mosaicing for tele-reality applications. In *WACV*, pages 44–53, 1994. 125, 126, 128

[SZTS06] J. Sun, W.W. Zhang, X. Tang, and H.Y. Shum. Background cut. In *in Proceedings of European Conference on Computer Vision*, volume 2, pages 628–641, 2006. 158

[THB03] L. Torresani, A. Hertzman, and C. Bregler. Learning non-rigid 3D shape from 2d motion. In *Perception*, volume 13 (7), pages 1555–1562, 2003. 120

[tie05] tierone.com. Market for automotive adaptive cruise control to soar in coming decade ; tier one report projects huge increase in installation rate, to $2.4b annually, 2005. `http://www.tierone.com/accmtrpr.html`. 2

[TM97] P. Torr and D. Murray. The development and comparison of robust methods for estimating the fundamental matrix. In *Intl. J. of Computer Vision*, pages 271–300, 1997. 28, 52

[TM04] A. Talukder and L. Matthies. Real-time detection of moving objects from moving vehicles using dense stereo and optical flow. In *International Conference on Intelligent Robots and Systems*, volume 4, pages 3718–3725, 2004. 108, 109, 120, 122

[TMB04] Gwenaëlle Toulminet, Stéphane Mousset, and Abdelaziz Bensrhair. Fast and accurate stereo vision-based estimation of 3D position and axial motion of road obstacles. *Int. J. Image Graphics*, 4(1) :99–126, 2004. 107

[TMHF99] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle adjustment — a modern synthesis. In B. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms : Theory and Practice*, number 1883 in LNCS, pages 298–373, Corfu, Greece, September 1999. Springer-Verlag. 40, 51

[Tsa87] R.Y. Tsai. A versatile camera calibration technique for hight-accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses. In *IEEE, Robotics and Automation*, volume 3, pages 323–334, 1987. 11, 14, 15, 31, 34

[TSK01] Hai Tao, Harpreet S. Sawhney, and Rakesh Kumar. Dynamic depth recovery from multiple synchronized video sequences. In *In Proceedings of IEEE Computer Vision and Pattern Recognition*, 2001. 122

[Tuk77] J. W. Tukey. *Exploratory Data Analysis*. Addison-Wesley, 1977. 52

[Ull79] S. Ullman. The interpretation of visual motion. *MIT Press*, 1979. 119

[Ull84] S. Ullman. Maximizing the rigidity : the incremental recovery of 3-d shape and nonrigid motion. 13 :730–742, 1984. 119

[VBR+99] Sundar Vedula, Simon Baker, Peter Rander, Robert Collins, and Takeo Kanade. Three-dimensional scene flow. In *IEEE International Conference on Computer Vision*, pages 722–729, 1999. 121

[WCH92] J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10) :965–980, 1992. 14, 15, 31

[WD86] A. M. Waxman and J. H. Duncan. Binocular image flows : Steps toward stereo-motion fusion. volume 8(6), pages 715–729, Nov 1986. 108, 120, 122

[WDH+07] Glenn Widmann, Michele Daniels, Lisa Hamilton, Lawrence Humm, Bryan Riley, Jan Schiffmann, David Schnelker, and William Wishon. Comparison of lidar-based and radar-based adaptive cruise control systems. In *SAE transactions*, volume 109, N°7, pages 126–139, 2007. 5

[Wil98] T. Williamson. *A High-Performance Stereo Vision System for Obstacle Detection.* PhD thesis, September 1998. 89, 107

[XCK04] J. Xiao, J. Chai, and T. Kanade. A closed-form solution to non-rigid shape and motion recovery. In *in Proceedings of European Conference on Computer Vision*, volume 13 (7), pages 233–246, 2004. 120

[YAW03] Qian Yu, Helder Araujo, and Hong Wang. Stereo-vision based real-time obstacle detection for urban environments. In *Proceedings of Int. Conf. on Advanced Robotics*, pages 1671–1676, July 2003. 87, 107

[YC90] G.S. Young and R. Chellappa. 3-d motion estimation using a sequence of noisy stereo images : Models, estimation, and uniqueness results. *IEEE*, 12(8) :735–759, August 1990. 120

[YC06] Chih-Hsien Yeh and Yung-Hsin Chen. Development of vision-based lane and vehicle detecting systems via the implementation with a dual-core dsp. *Intelligent Transportation Systems, 2006. Proceedings. 2006 IEEE*, pages 1179–1184, 2006. 104

[YPL04] R. Yang, M. Pollefeys, and S. Li. Improved real-time stereo on commodity graphics hardware. In *Proceedings of Computer Vision and Pattern Recognition Workshop*, pages 36–36, June 2004. 80, 106

[ZCS03] L. Zhang, B. Curless, and S. Seitz. Spacetime stereo : Shape recovery for dynamic scenes. In *In Proceedings of IEEE Computer Vision and Pattern Recognition*, 2003. 122

[ZF92a] Z. Zhang and O. Faugeras. *3D Dynamic Scene Analysis.* Springer-Verlag, 1992. 119

[ZF92b] Z.Y. Zhang and O.D. Faugeras. Estimation of displacements from two 3-d frames obtained from stereo. *IEEE*, 14(12) :1141–1156, December 1992. 120

[ZFD97] Z. Zhang, O. Faugeras, and R. Deriche. An effective technique for calibrating a binocular stereo through projective reconstruction using both a calibration object and the environment. *Videre - Journal of Computer Vision Research, vol. 1, no. 1, pp. 58-68*, 1997. 31

[Zha96] Zhengyou Zhang. Determining the epipolar geometry and its uncertainty : A review. Technical Report 2927, INRIA, 1996. 28, 60

[Zha00] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11) :1330–1334, 2000. 15, 31

[ZK00] Y. Zhang and C. Kambhamettu. Integrated 3D scene flow and structure recovery from multiview image sequences. In *In Proceedings of IEEE Computer Vision and Pattern Recognition*, pages 2674–2681, 2000. 121, 122

[ZK01] Y. Zhang and C. Kambhamettu. On 3D scene flow and structure estimation. In *In Proceedings of IEEE Computer Vision and Pattern Recognition*, volume 2, pages 778–785, 2001. 121, 122

Résumé

La complexité croissante de l'environnement routier et le souci d'amélioration de la sécurité routière expliquent l'intérêt que porte les constructeurs automobiles aux travaux sur l'aide à la conduite. De nombreux systèmes équipent déjà les véhicules de la rue. Alors que la perception de l'état du véhicule (vitesse, position, etc.) est maîtrisée, celle de l'environnement reste une tache difficile. Parmi tous les capteurs susceptibles de percevoir la complexité d'un environnement urbain, la stéréo-vision offre à la fois des performances intéressantes, une spectre d'applications très larges (détection de piéton, suivi de véhicules, détection de ligne blanches, etc.) et un prix compétitif. Pour ces raisons, *Renault* s'attache à identifier et résoudre les problèmes liés à l'implantation d'un tel système dans un véhicule de série, notamment pour une application de suivi de véhicules.

La première problématique à maîtriser concerne le calibrage du système stéréoscopique. En effet, pour que le système puisse fournir une mesure, ses paramètres doivent être correctement estimés, y compris sous des conditions extrêmes (forte températures, chocs, vibrations, ...). Nous présentons donc une une méthodologie d'évaluation permettant de répondre aux interrogations sur les dégradations de performances du système en fonction du calibrage.

Le deuxième problème concerne la détection des obstacles. La méthode mis au point utilise d'une originale les propriétés des rectifications. Le résultat est une segmentation de la route et des obstacles.

La dernière problématique concerne la calcul de vitesse des obstacles. Une grande majorité des approches de la littérature approxime la vitesse d'un obstacle à partir de ses positions successives. Lors de ce calcul, l'accumulation des incertitudes rendent cette estimation extrêmement bruitée. Notre approche combine efficacement les atouts de la stéréo-vision et du flux optique afin d'obtenir directement une mesure de vitesse 3-D robuste et précise.