



Stéréovision embarquée sur véhicule : de l'auto-calibrage à la détection d'obstacles

Vincent Lemonde

► To cite this version:

Vincent Lemonde. Stéréovision embarquée sur véhicule : de l'auto-calibrage à la détection d'obstacles. Réseaux et télécommunications [cs.NI]. INSA de Toulouse, 2005. Français. NNT : . tel-00012017

HAL Id: tel-00012017

<https://theses.hal.science/tel-00012017>

Submitted on 23 Mar 2006

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.

Stéréovision Embarquée sur Véhicule : de l'Auto-Calibrage à la Détection d'Obstacles.

THÈSE

présentée et soutenue publiquement le 10 novembre 2005

pour l'obtention du

Doctorat de l'Institut National Des Sciences Appliquées de Toulouse
(spécialité systèmes informatiques)

par

Vincent Lemonde

Composition du jury

<i>Rapporteurs :</i>	M. Didier AUBERT	Chargé de Recherche HDR, INRETS
	M. Roland CHAPUIS	Professeur, LASMEA
<i>Examineurs :</i>	M. Raja CHATILA	Directeur de Recherche, CNRS
	M. Jean-José ORTEU	Professeur, École des mines d'Albi-Carmaux
	M. Fawzi NASHASHIBI	Ingénieur de Recherche, École des mines de Paris
<i>Invité :</i>	M. Serge BOVERIE	Directeur R&D Siemens
<i>Directeur de thèse :</i>	M. Michel DEVY	Directeur de Recherche, CNRS

Remerciements

Bien qu'au début du manuscrit, ces quelques lignes représentent la fin de la rédaction. Le travail présenté ici n'est, bien sûr, pas l'œuvre d'une unique personne mais s'intègre dans une suite logique de travaux ainsi que dans un environnement social particulier. Je vais donc remercier l'ensemble des gens qui ont contribué à faire de ce document une thèse.

Mes premiers remerciements vont évidemment à **Malik Ghallab**, directeur du *Laboratoire d'Automatique et d'Architecture des Systèmes*, et à **Raja Chatila**, responsable du groupe *Robotique et Intelligence Artificielle*, pour m'avoir accueilli au sein de leur équipe.

Merci à l'ensemble du jury pour leur participation à clore mon aventure.

Historiquement (à mon échelle), le désir de faire une thèse vient de **Christophe Nicolle**, chercheur à l'Université de Bourgogne, qui m'a fait découvrir et aimer le milieu de la recherche. Par la suite, la rencontre avec **Dorian Garcia** m'a initié au domaine de la vision artificielle et m'a conforté dans l'idée que le milieu de la recherche était basé sur des gens ouverts, intéressants et sympathiques. C'est grâce à ces deux personnes que j'ai décidé de faire une thèse pour le plaisir. Merci à eux.

Je tiens à remercier **Michel Devy**, directeur de recherche, pour cette expérience unique. Merci pour cette collaboration, pour sa confiance, la liberté, son humour, les bières en bas du technocentre et ses conseils malgré une divergence notable de nos méthodes de travail.

Ce travail est le fruit d'une expérience importante dans le milieu industriel et je tiens à remercier les différents acteurs rencontrés au cours de ces contrats.

Je remercie vivement l'ensemble des relecteurs **Magali**, **Florane** (terriblement efficace), **Dany**, **Nono**, **Patrick**, **Thierry**, **Dorian** et **Michel** pour leur travail pas simple et qui m'a été très utile.

Merci aussi aux stagiaires qui m'ont aidé, particulièrement **Damien Quinton** très efficace.

L'environnement social proche permettant de ne pas se décourager, ou alors de le faire à plusieurs et donc de tenir le coup, est la base de l'achèvement de mon travail. Merci à mes amis **Thierry P.**, **Ludo**, **Aurélie**, **Sylvain A.**, **Sylvain J.**, **Sam**, **Jérôme**, **Arnaud**, **Jean-Gérard**, **Joan**, **Paulo**, **Manu Z.**, **Léo**, **Gabriel**, *Bertin* (pour avoir gardé **Jean-Michel** quelques semaines avec eux), **Andres**, **Thierry S.**, **Patrick**, **Viviane** et les autres. Chacun mériterait une page de remerciements personnalisée ! Particulièrement, merci à **Magali** et à ma famille (et chuya) pour m'avoir supporté (dans tous les sens du terme) pendant ces quelques années de doctorat.

Certains modèles 3D utilisés pour les images de synthèse sont issus de **Gilles Tran**¹. Merci à lui pour m'avoir permis d'utiliser ses modèles.

Pour terminer, je remercie l'ensemble du service technique du LAAS, en particulier sysadmin.

¹<http://www.oyonale.com>

Juste pour le plaisir

Table des matières

Table des figures	vii
-------------------	-----

Introduction générale

Bibliographie du chapitre	5
---------------------------	---

Chapitre 1

Contexte et positionnement

1.1	Les <i>Véhicules Intelligents</i> en Robotique	10
1.1.1	La détection d'obstacles en robotique	10
1.1.2	Spécificités du milieu automobile	14
1.2	Modélisation d'une caméra	16
1.2.1	Projection perspective : modèle Sténopé ou <i>trou d'épingle</i>	17
1.2.2	Distorsions	19
1.2.3	Calibrage d'une caméra	21
1.3	Modélisation du capteur stéréoscopique	23
1.3.1	Géométrie du capteur de stéréovision, géométrie épipolaire	24
1.3.2	calibration du banc	26
1.4	Stéréovision	27
1.4.1	Description algorithmique	28
1.4.2	L'implémentation multi-résolution et ses avantages	32
1.4.3	Modèle d'erreur	33

Bibliographie du chapitre	35
---------------------------	----

Chapitre 2

Recalibration d'un capteur stéréoscopique

2.1	Etat de l'art	40
2.1.1	Estimation de la matrice fondamentale	40

2.1.2	Ajustement de faisceaux	43
2.1.3	Conclusion	44
2.2	Evaluation de la qualité de calibration	45
2.3	Principe de la réestimation	48
2.3.1	Algorithme	49
2.3.2	Description du modèle à réestimer	50
2.4	Autocalibration du modèle d'erreur	53
2.5	Application à l'odométrie optique	53
2.6	Résultats	55
2.6.1	calibration ponctuel	56
2.6.2	calibration avec redondance temporelle	58
2.7	Conclusion et discussions	60

Bibliographie du chapitre 63

Chapitre 3

Détection d'obstacles

3.1	Etat de l'art	66
3.1.1	Capteurs actifs	67
3.1.2	Capteurs passifs	68
3.1.3	Conclusion	69
3.2	Détection des obstacles sur la voie de circulation	70
3.2.1	Détection de la route	70
3.2.2	Modélisation du profil longitudinal de route	70
3.2.3	Extraction des obstacles dans le profil longitudinal	74
3.2.4	Résultats	75
3.3	Caractérisation des obstacles	86
3.3.1	Extraction des dimensions des obstacles	87
3.3.2	Résultats	88
3.4	Conclusion	88

Bibliographie du chapitre 91

Chapitre 4

Modélisation de la place de parking

4.1	Description du problème	94
4.1.1	Repères et définitions	94

4.1.2	Spécifications	95
4.2	Etat de l'art	96
4.3	Evaluation des techniques issues de la Robotique	97
4.3.1	Extraction du plan du sol	98
4.3.2	Recalage des données entre plusieurs scènes	100
4.3.3	Conclusion	101
4.4	Détection des composantes de l'environnement	104
4.4.1	Extraction des obstacles	104
4.4.2	Détection de trottoirs	105
4.5	Modélisation de la place de parking	107
4.5.1	Modèle local	107
4.5.2	Modèle global	110
4.5.3	Mesure de la place libre	118
4.6	Résultats	118
4.6.1	Capteur <i>idéal</i> (selon le constructeur du système)	119
4.6.2	Capteur <i>idéal</i> (selon le développeur)	124
4.7	Conclusion	125

Bibliographie du chapitre

127

Chapitre 5

Expérimentations

5.1	Plateformes expérimentales	130
5.1.1	Véhicule expérimental	130
5.1.2	ATRV	132
5.1.3	Véhicule ARCOS	133
5.2	Recalibration d'un système stéréoscopique	133
5.2.1	Evaluation et recalibration sur des images d'intérieur	133
5.2.2	Recalibration sur véhicule	135
5.2.3	Conclusion	138
5.3	Estimation de l'odométrie optique	139
5.4	Détection d'obstacles en milieu routier	139
5.5	Modélisation de la place de parking	145
5.5.1	Banc de stéréovision sur l'axe central, à l'arrière	148
5.5.2	Banc stéréo sur le côté, à l'avant	148
5.5.3	Conclusion	148

Conclusions et Perspectives

1	Conclusions	153
2	Perspectives	154

Annexes	157
----------------	------------

Annexe A

Ajustement de faisceaux

A.1	Algorithme de <i>Levenberg-Marquardt</i>	160
A.2	Ajustement de faisceaux	161

Annexe B

Localisation de route

B.1	Présentation de l'approche choisie	167
B.2	Apprentissage et reconnaissance	168
B.2.1	Modèle de la route dans l'image	168
B.2.2	Apprentissage	169
B.2.3	Procédure de recherche	171
B.3	Localisation et suivi	174
B.3.1	Localisation du véhicule	174
B.3.2	Suivi de la route	175
B.4	Résultats	176

Bibliographie du chapitre	179
----------------------------------	------------

Table des figures

1.1	<i>Shakey</i> , premier robot mobile	10
1.2	<i>HRP-2</i> de l'entreprise Kawada Industries Inc	11
1.3	Mire composée d'un dièdre [gauche] et d'une simple grille [droite]	16
1.4	Modèle sténopé	17
1.5	Une caméra	21
1.6	Une séquence d'image de mire pour le calibrage d'une caméra	22
1.7	les trois référentiels tridimensionnels du capteur de stéréovision	23
1.8	Algorithme général de la stéréovision	29
1.9	Stéréocorrélation	30
1.10	Illustration de l'influence de la <i>profondeur</i> sur la disparité	30
1.11	Transformation CENSUS sur un pixel, avec une fenêtre de corrélation de 5×5 . . .	31
1.12	Pyramide de résolution sur une paire d'images	32
2.1	Evaluation de la qualité de calibration	46
2.2	Effet de la décalibration	47
2.3	Algorithme de recalibration avec approche robuste RANSAC.	50
2.4	Evaluation des paramètres de calibration	54
2.5	Objet de synthèse	56
2.6	Effet du bruit sur les points <i>3D</i>	57
2.7	Effet d'un fort bruit sur les points <i>3D</i>	58
2.8	Evaluation du facteur d'échelle sous l'effet du bruit	58
2.9	Effet du bruit sur les pixels	59
2.10	Effet du bruit sur les pixels	60
2.11	Effet des différents bruits sur la réestimation	61
2.12	Paire d'images de synthèse	61
2.13	Résultat de la réestimation sur la carte de disparité	62
2.14	Visualisation des distances entre appariement de points d'intérêt et appariement par stéréocorrespondance	62
2.15	Résultats de recalibration sur images de synthèse	62
3.1	Robustesse aux occultations de la détection de voie	71
3.2	Profil longitudinal de route en ligne droite	71
3.3	Etude d'une ligne image de la voie de circulation	72
3.4	Etude d'une ligne image de la voie de circulation en présence d'un obstacle . . .	73
3.5	Etude d'une ligne image de la voie de circulation présentant un dévers et/ou du roulis	73
3.6	Ligne iso-distante	74

3.7	Etude d'une ligne image de la voie de circulation présentant un dévers et un obstacle	74
3.8	Estimation du profil de route	75
3.9	Histogrammes selon la profondeur et la hauteur d'une ligne image de la voie de circulation	76
3.10	Résultat du profil longitudinal de route sur image de synthèse	76
3.11	Résultat de la détection d'obstacles sur image de synthèse	77
3.12	Paire stéréo sur route avec une variation de pente à l'avant du véhicule	77
3.13	Reconstruction d'une ligne image de la route	78
3.14	Histogramme en Z d'une reconstruction d'une ligne image de la route	78
3.15	Profil longitudinal de la route	79
3.16	Représentation v -disparité	79
3.17	Paire stéréo sur route en dévers	80
3.18	Reconstruction d'une ligne image de la route	81
3.19	Histogramme en Z d'une reconstruction d'une ligne image de la route	81
3.20	Profil longitudinal de la route	82
3.21	Estimation de la route et extraction des obstacles	83
3.22	Représentation v -disparité	83
3.23	Résultat de détection d'obstacles sur un cas général	84
3.24	Profil de route sans correction [haut] et profil de route corrigé des dévers [bas]	85
3.25	Représentation v -disparité	86
3.26	Lissage de la surface de la route par une quadrique [gauche] et estimation de la route avec extraction des obstacles [droite]	86
3.27	Caractérisation des obstacles	87
3.28	Profil de route [gauche], projection des points P_i^{3D} sur le plan YZ [droite]	88
3.29	Boîte englobante image de l'obstacle détecté	88
4.1	Variété des configurations possibles de parking	96
4.2	Interface graphique du système de parking automatique de la Toyota Prius	97
4.3	Système de coordonnées sphériques utilisé pour extraire le sol	98
4.4	Histogramme bi-variable des normales	99
4.5	Histogramme mono-variable H'	100
4.6	Détection des obstacles par segmentation de plan	102
4.7	Recalage de scène avec l'algorithme ICP	103
4.8	Algorithme d'extraction et de classification des obstacles	103
4.9	Grille de différenciation des obstacles de pas constant Δd_{obs} à gauche, de pas variable en profondeur à droite	105
4.10	Algorithme d'extraction du trottoir	106
4.11	Projection des points du sol et du trottoir	106
4.12	Interpolation du sol par une courbe de profil (coupe transversale et cumul)	107
4.13	Modélisation des obstacles dans le repère d'inertie	109
4.14	Création des boîtes englobantes	110
4.15	Fusion des perceptions locales par un graphe de repère	112
4.16	Fusion de boîtes englobantes	113
4.17	Problème de l'incertitude sur les dimensions des boîtes englobantes	115
4.18	Principe de filtrage des positions du trottoir par une approche robuste	115
4.19	Affinement des boîtes englobantes par boîtes d'occupation volumique	116
4.20	Plan de la séquence de parking avec images de synthèse	119
4.21	Courbe de la valeur de la disparité en fonction de la distance	120

4.22	Détection du premier obstacle	121
4.23	Détection de l'espace libre	122
4.24	Détection du deuxième obstacle	122
4.25	Résultat de la détection du premier obstacle (vue du dessus)	122
4.26	Résultat de la détection de l'espace libre (vue du dessus)	123
4.27	Résultat de la détection du deuxième obstacle (vue du dessus)	123
4.28	Disparité en fonction de la distance pour un banc de 40cm de base.	124
4.29	Résultats de modélisation	124
5.1	Plate-forme expérimentale du LAAS-CNRS	130
5.2	Courbe de disparité en fonction de la distance pour une base de 1.30	131
5.3	Architecture du système embarqué	131
5.4	Dala	132
5.5	Dala (face avant)	133
5.6	Evaluation de la qualité de calibration	134
5.7	Paire stéréoscopique utilisée pour la recalibration	134
5.8	Effet de la recalibration sur les résultats de la stéréocorrélation	135
5.9	Carte de disparité après filtrage et courbes de résultats d'évaluation de calibration	136
5.10	Recalibration sur une séquence routière ARCOS	137
5.11	Recalibration sur une séquence routière ARCOS grâce à un véhicule proche . . .	138
5.12	Comparaison de l'odométrie visuelle avec l'odométrie optique	140
5.13	Suivi des points dans le temps	141
5.14	Courbes des disparités en fonction de la distance d'un point pour une base de 1 m	142
5.16	Détection d'un obstacle à 30m	142
5.15	Détection d'un obstacle à 11.5 m	143
5.17	Route sans obstacle	144
5.18	Détection d'un piéton	144
5.19	Détection d'un obstacle la nuit	145
5.20	Problème lié au capteur CCD	145
5.21	Courbes des disparités en fonction de la distance d'un point pour un base de 33 cm	146
5.22	Les étapes de la modélisation locale	147
5.23	Résultat de modélisation de place de parking (1)	149
5.24	Résultat de modélisation de place de parking (2)	150
B.1	Organigramme de l'approche	168
B.2	Modèle pour différentes valeurs du nombre d'échantillons	169
B.3	Représentation du modèle	170
B.4	Exemple d'évolution de l'algorithme	172
B.5	Filtre utilisé pour la mesure du gradient	173
B.6	Organigramme de l'étape de reconnaissance	174
B.7	Résultats sur les images Arcos	176
B.8	Changement de voie	177
B.9	Ligne droite	177
B.10	Ecart brutal du véhicule	177

Introduction générale

Un véhicule circulant sur une route est un système complexe, soumis à des contraintes temps réel critiques, composé d'un certain nombre de sous-systèmes qui interagissent entre eux : véhicule (capacités mécaniques), conducteur (comportement, perception, ...) et la route (interface roue-sol, signalisation...) sources de perturbations. Le domaine des ITS² offre donc un vaste terrain de recherche sur l'intégration de systèmes, faisant appel à des disciplines multiples (automatique, informatique, physique, électronique ...). L'ensemble des outils mathématiques et informatiques actuels trouve dans ce cadre un champ d'applications et de développements extrêmement intéressants. De plus, les départements de recherche et développement des constructeurs et grands équipementiers, pour qui l'électronique et l'informatique embarquées occupent de plus en plus de place, sont demandeurs d'études de fond pour des raisons allant de la sécurité au confort en passant par le marketing.

La complexité croissante de l'environnement routier et le besoin d'amélioration de la sécurité routière expliquent un nombre important de travaux sur l'aide à la conduite. La *voiture du futur* devra aider le conducteur à mieux comprendre son environnement pour l'aider dans ses décisions sans jamais le déresponsabiliser. Elle devra donc être capable de percevoir aussi bien et même mieux son environnement que ne peut le faire l'être humain seul et pallier ses déficiences. C'est le rôle de l'électronique embarquée qui mène sa révolution, d'année en année, dans le secteur automobile, l'équipement électronique pouvant représenter entre vingt et trente pour-cent du prix du véhicule. Ces équipements peuvent gérer le fonctionnement des parties mécaniques comme des systèmes de plus haut niveau permettant d'améliorer la sécurité et le confort. L'ABS³ est la première aide à la conduite démocratisée sur les voitures de série dès 1978. Depuis, la sécurité étant le maître mot de l'argumentaire marketing, les Airbags et ABS côtoient ESP⁴, ASR⁵, EBD⁶ et ACC⁷.

Le domaine des systèmes d'aide à la conduite présente trois niveaux de complexité. Le premier, système passif purement informatif, n'intervient jamais directement sur la conduite du véhicule mais complète la connaissance du conducteur. Le conducteur est informé d'un événement (présence d'un obstacle, état d'hypovigilance...) via des interfaces Homme-Machine de plus en plus sophistiquées (du simple *bip-bip* à l'écran ou au siège vibrant). Son application concerne, par exemple, la rétrovision pour la détection d'angle mort, la détection de la vigilance du conducteur, la détection d'obstacles, la détection prédictive de la sortie de route ou encore

²Intelligent Transportation Systems

³Automatic Breaking System

⁴Electronic Stability Program

⁵Automobile Shredder Residue, système d'antipatinage

⁶Electronic Brake force Distribution, système de répartition électronique de freinage

⁷Adaptive Cruise Control

l'aide au parking et bien d'autres encore. Le second niveau est représenté par les systèmes actifs qui interviennent uniquement en cas de détection de danger pour pallier une défaillance avérée du conducteur (afin de limiter l'impact de l'accident). L'exemple le plus connu est celui de l'ABS mais il existe aussi des applications telles que l'*air-bag* prédictif et le contrôle de stabilité de trajectoire, par exemple. Le troisième niveau concerne le contrôle automatique du véhicule, à mi-chemin entre sécurité et confort, comme le contrôle longitudinal urbain (*Stop and Go*) et l'ACC.

L'**acceptabilité juridique et sociale** de tels systèmes, surtout quand ils se rapprochent de systèmes automatiques, pose évidemment problème. Dès lors qu'un véhicule terrestre à moteur est impliqué dans un accident de la circulation, la responsabilité du conducteur est *a priori* engagée. Si un dispositif, semi-automatique ou automatique intervient pour atténuer les conséquences de cet accident, la question se pose de savoir si la responsabilité du conducteur reste entière ou si elle se partage avec celle du constructeur, fournisseur du dispositif intervenu qui se serait avéré moins efficace qu'escompté. Du fait de la multiplicité des intervenants dans la chaîne de responsabilité de nombreux textes de lois sont applicables. Ce problème complexe est traité avec la plus grande attention et a été abordé dans le projet ARCOS⁸ pour lequel un état de l'art juridique et des premières réponses ont été apportées pour la France. Il est évident que les réponses ne sont pas universelles et que chaque pays a ses lois qui peuvent souvent être contradictoires. On sait que le Japon est réputé pour accepter rapidement les innovations par rapport à l'Europe.

Pour l'ensemble de ces applications, le choix des capteurs est fortement dépendant (1) des données nécessaires à la réalisation de l'application, (2) des contraintes d'encombrement, (3) des contraintes d'énergie et de sécurité, (4) des contraintes de prix et (5) des contraintes de temps de traitement. Un grand nombre de capteurs différents sont donc utilisés dans le milieu automobile comme les radars, la télémétrie laser ou encore les ultrasons. Depuis quelques années, l'utilisation d'un nouveau capteur pour l'automobile, la caméra, est apparue. En effet, il y a encore quelques temps, la vision artificielle était restreinte aux applications nécessitant des temps d'exécution assez faible. Avec le développement de calculateurs de plus en plus performants, il est maintenant possible d'utiliser le traitement d'image pour des fonctions critiques telles que la détection d'obstacles ou la gestion des inter-distances. De plus, les technologies de caméras numériques, comme la technologie CMOS⁹, permettent leur utilisation malgré des conditions météorologiques difficiles (fort ensoleillement, nuit, neige, pluie ...) avec une miniaturisation de plus en plus importante. Bien qu'encore un peu cher (environ 5\$ pour une production en série), ce capteur trouve donc naturellement sa place au sein du domaine automobile, l'utilisation pour plusieurs fonctions distinctes tendant à en diminuer le coût. Dans cette thèse, nous avons fait le choix d'étudier les limites en complexité de la vision artificielle, spécialement les limites de l'exploitation d'informations tridimensionnelles obtenues par un capteur stéréoscopique.

L'**objectif majeur** de cette thèse est d'analyser les applications possibles de la stéréovision embarquée sur véhicule. Pour cela, le premier centre d'intérêt concerne le **calibrage automatique du capteur stéréoscopique** et le second traite de **deux applications majeures d'aide à la conduite**. La première application réalise une **fonction de détection d'obstacles en milieu routier** et la deuxième consiste en la **modélisation de l'environnement en milieu urbain pour une fonction d'aide au parking**. Les travaux réalisés dans le cadre de cette

⁸Action de Recherche pour la COnduite Sécurisée

⁹Complementary Metal Oxide Semiconductor

thèse ont pour vocation d'appliquer, au milieu automobile, les techniques issues de la robotique mobile en particulier, d'exploiter le savoir faire accumulé depuis de nombreuses années en perception, au sein du groupe **Robotique et Intelligence Artificielle**(RIA) du **Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS** (LAAS-CNRS).

Le **chapitre 1** de cette thèse présente une vue globale des techniques de **détection d'obstacles en robotique** et présente les spécificités du milieu automobile. Il donnera les spécifications des applications d'aide à la conduite auxquelles nous nous intéressons. Après cette mise en contexte de notre travail, ce chapitre présentera la **modélisation** et les méthodes de **calibrage exact** d'un banc stéréoscopique avec un état de l'art des méthodes à base de mires ainsi que des méthodes photogrammétriques.

Le **chapitre 2** traite le problème du **calibrage en ligne du capteur stéréoscopique**. Etant donné la finalité de système embarqué de notre travail, nous nous sommes intéressés aux techniques d'auto-calibrage décrites dans la littérature et une attention particulière a été portée sur la transparence de la méthode vis-à-vis de l'utilisateur. Ainsi, la qualité des résultats du capteur n'est plus dépendante de l'obsolescence du calibrage qui dérive avec le temps et les nombreux chocs et vibrations auxquels le véhicule est soumis [Lemondé et al., 2006].

Le **chapitre 3** est consacré à l'étude d'une **fonction de détection d'obstacles en milieu routier**. Après avoir présenté un état de l'art du domaine, nous proposons une approche de création de profil longitudinal de route prenant en compte le dévers et/ou le roulis et les variations de pente de la chaussée. L'approche retenue est générique à n'importe quelle géométrie de route et permet l'extraction des obstacles sans fausse détection [Lemondé et al., 2004, Lemondé et al., 2005a, M.Devy et al., 2005].

Le **chapitre 4** présente la réalisation d'une tâche de **perception de l'environnement urbain pour l'aide au parking**. Après avoir présenté les systèmes existants, nous décrirons notre approche *3D* permettant d'extraire une information d'*espace libre* au travers de la détection des obstacles de l'environnement. Ce chapitre met en œuvre une détection d'obstacles à courte portée avec une fusion temporelle des données acquises afin de construire, de manière incrémentale, un modèle *3D* complet de la place de parking [Lemondé et al., 2005b, Devy et al., 2004, Devy et al., 2005].

Les chapitres 1, 2, 3 et 4 se terminent par une évaluation des méthodes sur images ou données de synthèse (génération *povray*¹⁰) afin de tester les limites théoriques des algorithmes proposés.

Le **chapitre 5** sera consacré aux résultats sur images réelles. Il décrit la délicate tâche de la **validation des algorithmes sur images réelles** malgré l'extrême diversité des situations possibles. Après une présentation de l'environnement et d'exécution temps réel utilisé, nous verrons, les uns après les autres, les résultats expérimentaux des trois chapitres précédents. Les séquences d'images ont été acquises sur différentes plates-formes : le véhicule démonstrateur du LIVIC¹¹, exploité pour acquérir les base de données du projet, un véhicule *Renault Safrane* véhicule expérimental du LAAS-CNRS et un robot mobile du groupe RIA du LAAS-CNRS.

¹⁰www.povray.org

¹¹Laboratoire des Interactions Véhicule-Infrastructure-Conducteur localisé à Versailles-Satory

Enfin, la conclusion terminera cette étude par un bilan des méthodes analysées et proposera des extensions possibles dans un avenir proche ou lorsque les limites techniques actuelles seront dépassées.

Le lecteur trouvera dans les annexes quelques détails sur les méthodes utilisées dans ce travail.

Bibliographie du chapitre

- [Devy et al., 2004] M. Devy et V. Lemonde (2004). Détection de l'environnement par stéréo caméra arrière : spécifications fonctionnelles. Convention de recherche cnrs/2002/006, 70p., rapport laas n.04087, LAAS - CNRS.
- [Devy et al., 2005] M. Devy et V. Lemonde (2005). Détection de l'environnement par stéréo caméra arrière : rapport final. Convention de recherche cnrs/2002/006, 120p., rapport laas n.05143, LAAS - CNRS.
- [Lemonde et al., 2004] V. Lemonde et M. Devy (2004). Obstacle detection with stereovision. Dans *Mechatronics and Robotics 2004 (MECHROB'04)*, volume 3, pages 919–924, Aachen (Allemagne).
- [Lemonde et al., 2005a] V. Lemonde et M. Devy (2005a). Détection d'obstacles par stéréovision sur véhicules intelligents. Dans *Proc. 9èmes Journées Jeunes Chercheurs en Vision par Ordinateur (ORASIS'05), Rapport LAAS No05151, 9p.*, Fournols (France).
- [Lemonde et al., 2005b] V. Lemonde et M. Devy (2005b). Stereovision for parking assistance. Dans *European congress SENSACT*, Noisy-Le-Grand.
- [Lemonde et al., 2006] V. Lemonde et M. Devy (2006). Calibrage en ligne d'un capteur stéréo monté sur un véhicule. Dans *soumis au 15e congrès francophone AFRIF-AFIA Reconnaissance des Formes et Intelligence Artificielle*, Tours. En attente d'acceptation.
- [M.Devy et al., 2005] M.Devy et V.Lemonde (2005). Perception de l'environnement à l'avant du véhicule : fonction détection d'obstacles par stéréovision. Projet coopératif arcoss du programme predict, rapport laas no05358, 50p., LAAS - CNRS.

Notations

s	un scalaire
$\mathbf{v}$	un vecteur de $\mathbb{R}^n$
$\mathbf{v}^t$	le vecteur transposé, <i>i.e.</i> $\mathbf{v}^t = (x_1 \ x_2 \ \dots \ x_n)$
$\mathbf{A}$	une matrice
$\mathbf{M}$	un point de l'espace tridimensionnel $\mathbb{R}^3$, e.g. de coordonnées $M = (x \ y \ z)^t$
$\mathbf{m}$	un point de l'espace bidimensionnel $\mathbb{R}^2$, e.g. de coordonnées $m = (u \ v)^t$
$\hat{\mathbf{m}}$	un point d'une image dont les coordonnées $\hat{u}$ et $\hat{v}$ sont déterminées expérimentalement
$\tilde{\mathbf{m}}, \tilde{\mathbf{M}}$	les points $\mathbf{m}$ et $\mathbf{M}$ exprimés en coordonnées homogènes, <i>i.e.</i> respectivement dans les espaces projectifs $\mathbb{P}^2$ et $\mathbb{P}^3$
$\cong$	égalité projective, <i>i.e.</i> l'égalité définie à un facteur non nul près : $\tilde{\mathbf{m}} \cong s\tilde{\mathbf{m}}, \forall s \in \mathbb{R}^*$
$\mathcal{R}_c$	un référentiel de l'espace bi- ou tridimensionnel

Chapitre 1

Contexte et positionnement

Sommaire

1.1	Les <i>Véhicules Intelligents</i> en Robotique	10
1.1.1	La détection d'obstacles en robotique	10
1.1.2	Spécificités du milieu automobile	14
	Environnement routier	14
	Environnement urbain	15
1.2	Modélisation d'une caméra	16
1.2.1	Projection perspective : modèle Sténopé ou <i>trou d'épingle</i>	17
1.2.2	Distorsions	19
	Modélisation paramétrique	19
	Prise en compte des distorsions dans le modèle caméra	20
	Modélisation non paramétrique	20
1.2.3	Calibrage d'une caméra	21
	Méthode classique	21
	Méthode photogrammétrique	22
1.3	Modélisation du capteur stéréoscopique	23
1.3.1	Géométrie du capteur de stéréovision, géométrie épipolaire	24
	Matrice essentielle	25
	Matrice fondamentale	26
	Triangulation [Hartley et al., 1997]	26
1.3.2	calibration du banc	26
	calibration classique dit <i>linéaire</i>	27
	calibration dit <i>global</i>	27
	calibration photogrammétrique avec mire de mouchetis	27
1.4	Stéréovision	27
1.4.1	Description algorithmique	28
	CENSUS	31
	Choix du critère de corrélation	31
1.4.2	L'implémentation multi-résolution et ses avantages	32
	Architecture générale	32
	Description d'un niveau de la pyramide de résolution	32
	Corrélation avec estimée du niveau suivant	33
	Avantages de la multi-résolution	33
1.4.3	Modèle d'erreur	33

Ce chapitre présente le contexte général de nos travaux, ainsi que les techniques de base exploitées tout au long de notre thèse. Nous avons bénéficié des acquis importants qui existent dans le groupe Robotique et IA du LAAS-CNRS dans divers domaines, particulièrement dans le thème de la Perception. En effet, les recherches et développements dans l'automobile, exploitent de nombreuses techniques issues de la Robotique, que ce soit en Commande, en Planification ou en Perception. Les principales équipes qui interviennent sur les aspects technologiques en ce domaine, sont souvent issues de la Robotique (en France, LASMEA, équipe CAOR de l'Ecole des Mines de Paris, Heudiasic, divers projets de l'INRIA ...).

En section 1.1 de ce chapitre, nous précisons les liens qui existent entre la Robotique et le domaine des *Véhicules Intelligents* et présenterons les spécificités d'un Véhicule vis-à-vis d'un Robot. Nos travaux ont trait à la Vision Artificielle : le modèle du capteur visuel, la caméra, sera rappelé en section 1.2. Nous verrons aussi en section 1.3 comment est modélisé un banc stéréoscopique. Enfin, la section 1.4 décrira l'algorithme de stéréovision qui sera ensuite exploité dans les chapitres traitant de la détection d'obstacles et de la modélisation de l'espace libre.

1.1 Les Véhicules Intelligents en Robotique

1.1.1 La détection d'obstacles en robotique

C'est en 1966 qu'apparaît *Shakey* (cf. figure 1.1) le premier robot mobile, fruit du travail du *Stanford Research Institute*¹². Muni d'une caméra TV, d'un télémètre par triangulation, d'un capteur de contact et de deux roues motrices, il est le premier à matérialiser la robotique mobile.



FIG. 1.1 – *Shakey*, premier robot mobile

Celle-ci ne cesse d'évoluer et déjà trente ans nous séparent de *Shakey*. Depuis, *Pathfinder* a roulé sur Mars, les robots humanoïdes tels que *HRP-2* (cf. figure 1.2) font leur apparition et Sony vend même en milliers d'exemplaires son petit chien *Aibo*.

Malgré ces évolutions notables, la boucle *perception-décision-action*, n'est toujours pas complètement maîtrisée. Le lecteur pourra se reporter à [Laumond, 2001] pour un état des lieux de la robotique mobile. Cette section présente, quant à elle, la détection d'obstacles dans son contexte *robotique*.

La communauté Robotique a pour objectif principal, d'accroître l'**autonomie** des robots ; une telle machine doit disposer des capacités fonctionnelles et décisionnelles suffisantes pour exécuter des tâches en s'adaptant aux évolutions de l'environnement. En particulier, pour traiter de l'autonomie en Robotique Mobile, il faut doter un robot des capacités suivantes :

¹²ancien nom du *SRI International*



FIG. 1.2 – HRP-2 de l'entreprise Kawada Industries Inc

- l'apprentissage de l'espace navigable de l'environnement,
- la planification des mouvements requis par la tâche assignée à plus haut niveau au robot,
- et enfin, l'exécution des trajectoires planifiées de manière robuste, donc avec des capacités de détection d'obstacles non prévus dans le plan, et d'évitement des obstacles détectés.

Ces capacités seront mises en oeuvre dans différents contextes. Selon l'environnement, selon la tâche à exécuter, selon le niveau d'interaction avec un opérateur humain, plusieurs modes de navigation se distinguent. Le plus classique est la navigation en *mode planifié* : le robot doit disposer d'un modèle de l'environnement, soit fourni comme connaissances a priori, soit appris dans une phase préalable. Ce modèle lui indique l'espace navigable (sol libre en intérieur, terrain plutôt plat en extérieur) les obstacles statiques (murs et objets de type meubles, chaises... en intérieur ; terrain accidenté non navigable et objets de type rochers, arbres ... en extérieur). Il exploite ce modèle pour générer un chemin et/ou une trajectoire dans l'espace libre. Il exécute cette trajectoire, avec une fonction réactive de détection d'obstacles : la procédure d'évitement peut être plus ou moins complexe

- arrêt pour apprendre un modèle local et planifier une trajectoire d'évitement,
- déformation dynamique de la trajectoire en cours d'exécution : *elastic band* ...

En ce mode planifié, nous pouvons distinguer encore les approches métriques et topologiques.

- les techniques métriques sont les plus génériques : elles nécessitent un modèle géométrique précis de l'environnement. Une trajectoire est définie comme une courbe ou une bande sur le sol, sur laquelle le robot doit se déplacer à une certaine vitesse. Le robot est contrôlé par un asservissement en vitesse à l'aide de capteurs de position proprioceptifs (odométrie) et/ou extéroceptifs (localisation sur amers), par un asservissement en vitesse pour rester sur la courbe. Les obstacles sont des objets non présents dans le modèle, qui intersectent cette bande.
- Les techniques topologiques sont adaptées pour des environnements semi-structurés de type réseau de couloirs (en intérieur) ou de routes (en extérieur) ; le modèle est un graphe reliant différents lieux (portes, carrefours...) par des arêtes associées à un chemin prédéfini entre ces lieux (couloir en deux portes, chemin entre deux carrefours...). Une trajectoire est définie comme un chemin dans le graphe topologique. Des primitives spécifiques de déplacement (suivi de couloir, suivi de routes...), de type commande référencée capteur (vision, laser...) permettent au robot de se déplacer d'un lieu au suivant. Les obstacles sont des objets statiques ou dynamiques qui perturbent ces primitives : autre entité (robot, personnes...) se déplaçant sur la même route, objet statique barrant la route ...

Les approches métriques sont généralement indispensables dans des environnements encombrés,

typiquement pour un robot d'intérieur qui doit traverser une pièce ou un hall... La difficulté réside dans la fonction de localisation du robot. Les approches topologiques seront très adaptées si l'environnement comporte des lieux discriminants, connectés par des chemins que le robot peut parcourir en étant asservi sur des données sensorielles. La difficulté réside dans l'identification des lieux et la commande référencée capteur.

La navigation *en mode réactif* n'exploite pas de modèle de l'environnement. Typiquement, le robot suit un cap, ou se dirige vers un objet lointain perçu dans une image, en restant sur l'espace libre. La trajectoire idéale est la ligne droite entre la position courante et le point à atteindre. Les obstacles sont soit des trous (fossés...), des terrains plats, mais dangereux pour le robot (étendue d'eau, de sable...) ou des objets émergeant du sol, en général statiques (rochers, végétations...), qui intersectent cette droite. Le robot appliquera une stratégie locale de contournement afin de se diriger de nouveau dans la bonne direction. Ce mode est adapté uniquement dans des environnements plats (terrain navigable) et peu encombrés : typiquement, traverser un champ pour un robot agricole, un espace classifié libre pour un robot d'exploration planétaire...

En robotique, la notion d'obstacle dépend donc du mode de navigation, du type d'environnement, du type de représentation dont dispose le robot. Nous décrivons succinctement quelques méthodes étudiées par cette communauté, qui nous a inspiré dans nos travaux.

Apprentissage d'une carte de l'environnement L'évitement d'obstacles en mode planifié s'appuie sur une connaissance *a priori* du monde à l'aide d'une carte contenant les espaces libres et les objets statiques de l'environnement. Cette carte doit être construite dans une phase préalable, dite d'exploration. Il existe de nombreuses représentations, mais nous n'en retiendrons que quelques-unes.

Les cartes stochastiques [Smith et al., 1986]. Ce sont des représentations éparées, qui contiennent des entités exploitables pour localiser le robot : primitives géométriques (points, droites, plans...) ou objets remarquables (amers). Ces entités doivent être très discriminantes pour que le robot puisse les reconnaître sans ambiguïté et les exploiter pour se localiser.

Cette carte peut être créée préalablement par apprentissage. Selon le capteur, il existe plusieurs types de méthodes :

- en vision monoculaire, les connaissances 3D ne sont pas disponibles après une seule acquisition. Les méthodes dites de *Structure From Motion* exploitent de nombreuses vues de l'environnement, pour produire une reconstruction 3D de manière globale. Nous décrivons l'algorithme appelé *Bundle Adjustment* à la base de ces méthodes globales, en chapitre 2 et en annexe A. Dans le contexte Transport, ces techniques sont exploitées dans le projet MOBIVIP¹³ pour traiter de la localisation d'un véhicule en site urbain.
- avec un capteur 3D (typiquement, télémétrie laser ou stéréovision), la carte peut être créée au fur et à mesure des déplacements du robot par des techniques telles que le *SLAM*¹⁴.
- notons les travaux de notre collègue J.Sola [Sola et al., 2005] qui exploite le formalisme du SLAM en vision monoculaire, pour les appliquer également dans un contexte Transport.

Les grilles d'occupation [Moravec et al., 1985]. Ce sont des représentations discrètes et denses, exploitées sur sol plat, donc essentiellement pour des applications en milieu intérieur.

¹³Projet du PREDIT3, dédié aux Véhicules Individuels Public pour la Mobilité en centre ville

¹⁴Simultaneous Localization And Mapping

Le plan $2D$ du sol est discrétisé en cellules d'une taille donnée, dépendant de la précision des capteurs (typiquement, $10 \times 10\text{cm}$) ; chaque cellule contient une probabilité a posteriori que le sol soit occupé $P(Occup|Observations)$; après chaque acquisition, la grille est mise à jour par application de la règle de Bayes. Une telle grille donne surtout un modèle de l'espace libre, même si certains auteurs exploitent aussi cette représentation pour traiter de la localisation du robot. Une grille peut être

- globale : elle couvre l'espace de travail du robot et sert à planifier les trajectoires dans l'espace libre.
- ou locale et glissante : elle est construite autour de la position courante du robot, et sert essentiellement pour l'évitement des obstacles.

Initialement, les grilles étaient surtout utilisées pour fusionner des données sensorielles issues de capteurs ultra-sons, données très incertaines. Elfes a aussi exploité cette représentation pour traiter de la fusion entre données laser et données stéréo.

Les cartes d'élévation [Lacroix, 1995]. C'est une généralisation des grilles d'occupation, applicables pour les terrains accidentés. Pour chaque cellule, on connaît l'élévation du terrain, et éventuellement en vision, des attributs photométriques (couleur, texture...) : des techniques de classification peuvent ensuite être exploitées pour déterminer si une cellule est navigable sans problème (plat), navigable mais avec un mode de locomotion spécifique (fortes pentes...) ou non navigable. Une telle carte étiquetée est un modèle de l'espace navigable, exploité ensuite pour sélectionner une stratégie de déplacement dans l'environnement.

Les graphes topologiques [Cernatès, 2005]. Ce sont des représentations graphiques. Elles s'assimilent à des plans ou à des cartes routières utilisées par un opérateur ou un conducteur pour se guider dans un bâtiment ou dans un réseau routier.

A partir de l'instant où l'objectif est connu, le chemin devant être exécuté par le robot est planifié sur la base d'une de ces représentations : courbes dans l'espace libre, séquence de cellules à travers une grille ou une carte d'élévation, chemin dans un graphe topologique. Seuls sont pris en compte les obstacles présents dans le modèle de l'environnement.

Détection et Evitement d'obstacles Pendant l'exécution de la trajectoire en mode planifié, ou pendant son mouvement dans une direction donnée en mode réactif, le robot doit détecter pendant son mouvement, si la zone devant lui est libre d'obstacles. Ce traitement doit être rapide, afin d'éviter une collision, surtout si l'environnement est dynamique (autres robots ou véhicules), ou s'il s'agit d'un lieu public (robot guide de musée par exemple).

De ce fait, la détection d'obstacles est en général réalisée sur un robot, à l'aide de capteurs $3D$ très réactifs, avec une portée qui dépend de l'application : capteurs infrarouges (moins de 1m.), capteurs ULTRASONORES (jusqu'à quelques m.) ou télémètre laser (jusqu'à 10m). Avec l'augmentation de la puissance de calcul, des techniques de vision artificielle (monoculaire ou stéréo) sont exploitées aussi pour cette fonction.

Concernant la stratégie d'évitement, il existe de nombreuses méthodes. [Quinlan et al., 1993] proposent un évitement d'obstacles en simulant une bande élastique se déformant selon les règles de la physique en étant repoussée par les obstacles et attirée par son objectif. [Minguez et al., 2004] discrétisent les situations possibles en situations distinctes (environnement libre, objectif dans

un espace libre, objectif de l'autre coté d'un obstacle ...) et adaptent la réaction du robot à la situation. [Brock et al., 1999] ont développé le *global dynamic window approach* considérant en même temps les obstacles et les problèmes de cinématique et de dynamique propre au robot. De même, [Lamiraux et al., 2004] proposent un évitement d'obstacles pour robot non holonome.

1.1.2 Spécificités du milieu automobile

L'environnement routier se distingue de la robotique par (1) son caractère dynamique changeant particulièrement marqué et (2) par des distances de perception qui peuvent être importantes. En effet, le véhicule a pour vocation d'évoluer au sein d'un environnement partagé par de nombreux usagers de nature hétérogène tels que les voitures, les piétons, les camions et les motocyclettes. Les systèmes de perception ont pour rôle d'aider le conducteur à gérer toutes ces interactions. De plus, les vitesses atteintes par les différents protagonistes peuvent être très élevées. Ainsi, une situation que l'on peut considérer comme évoluant *normalement* peut très rapidement dégénérer à la moindre modification d'un des paramètres du système. Typiquement, le suivi d'un véhicule sur une route de campagne un soir d'automne peut se transformer en quelques secondes en une situation de crise suite au freinage brusque du véhicule devenu l'obstacle à éviter. Pour pouvoir réagir à de telles situations, le véhicule doit être capable de percevoir et donc de disposer de systèmes de perception adaptés aux fonctions d'aide à la conduite. Compte tenu de la complexité des situations, il n'est pas possible de connaître tous les paramètres, sur l'environnement, en même temps afin de garder une bonne réactivité. Il faut donc focaliser les capteurs sur les informations pertinentes à la fonction à laquelle ils sont dédiés.

Ce travail de thèse se consacre à deux fonctions d'aide à la conduite que sont la **détection d'obstacles en milieu routier** et la **perception d'une place de parking en milieu urbain**. Pour chacune de ces fonctions, il est donc important de connaître le contexte de fonctionnement pour limiter dans la mesure du possible la perception à traiter l'information nécessaire et suffisante. C'est pourquoi nous décrivons, dans les deux paragraphes suivants, les contextes et hypothèses sur les environnements routier et urbain des deux fonctions traitées.

Environnement routier

L'environnement routier étudié dans ce travail est caractérisé de routes très larges (plusieurs voies de circulation comme sur les autoroutes et roades de certaines villes) aux larges rayons de courbure. La détection d'obstacles en milieu routier a pour objectif de mesurer la distance entre le véhicule porteur du système et le premier obstacle sur la même voie de circulation. Les spécifications de cette fonction ont été décrites dans [Tournié, 2002] lors du projet ARCOS. En voici les grandes lignes :

- le plus petit objet à percevoir est un piéton de $0.3m$ de large et $1m$ de haut au minimum,
- la distance maximale de perception est de $43.2m$,
- la distance minimale de perception est de $6m$,
- le rayon de courbure maximal de la route est de $120m$,
- un espace latéral de $0.5m$ entre deux objets à $45m$ doit pouvoir être distingué,
- un espace en profondeur de $2.5m$ doit être distingué entre deux objets à $45m$,
- le champ de vue du capteur doit être d'au moins 31° (largeur de $3.5m$ à une distance minimale de $6m$),
- la précision sur la distance et sur la vitesse du véhicule précédent doit être de $\pm 5\%$,
- le système développé doit retourner un résultat de traitement fiable au moins tous les

100ms (10Hz) avec une confirmation des données sur 10 trames soit un système complet à 100Hz,

- il ne faut pas de fausses alarmes,
- le capteur doit pouvoir fonctionner de jour comme de nuit (avec éclairage des feux de croisement).

L'impact de ces spécifications sur le capteur a été étudié dans [Aubert et al., 2003]. La conclusion de ce document est qu'il n'existe pas de capteur *parfait* remplissant l'ensemble des contraintes spécifiées. La solution proposée est alors soit un couplage de capteur de manière à ce que les déficiences des uns soient compensées par les qualités des autres, soit un relâchement de certaines contraintes. L'objectif de cette thèse n'est pas de réaliser un système complet et commercial mais plutôt de proposer une solution basée sur le traitement d'image pouvant servir de brique de base à un système plus avancé. Nous avons donc retenu uniquement la conclusion sur les spécifications du capteur stéréoscopique. Les caractéristiques principales du banc doivent être :

- une distance inter-caméra de 1m,
- des focales de l'ordre de 8.5mm,
- une matrice CCD ou CMOS 1/2",

sans nécessité de convergence des caméras. Le système ainsi décrit répond à l'ensemble des contraintes en considérant qu'il est possible de traiter un écart de 1 pixel en disparité afin de remplir la contrainte de résolution en profondeur de 2.5m à 45m. L'emplacement de ce capteur trouve naturellement sa place à l'avant du véhicule. La difficulté est de trouver un emplacement pouvant accueillir des caméras espacées d'un mètre et dont le risque de chocs est raisonnablement faible. Les possibilités sont de les fixer (1) dans les phares, bien que la pollution lumineuse et le nettoyage puissent poser problème ou alors (2) dans l'habitacle en haut du pare-brise, comme cela est fait dans la plupart des démonstrateurs de laboratoire (au LIVIC en particulier). Cette dernière solution est la plus adaptée à un système expérimental de stéréovision mais ne peut s'envisager sur un véhicule de tourisme, les caméras étant trop exposées aux passagers. Il est clair qu'un constructeur devrait intégrer au mieux un tel système (par exemple dans les montants de part et d'autre du pare-brise). Nous verrons en chapitre 3 et 5 le choix de cette position.

Environnement urbain

L'environnement urbain, dans le contexte parking, relève d'hypothèses bien différentes de l'environnement routier. En effet, le véhicule évolue à basse vitesse au milieu d'un environnement quasiment statique lors d'une recherche de place de parking. Le comportement typique est le passage du véhicule le long d'une route en ligne droite avec des véhicules garés éventuellement des deux côtés. L'identification des obstacles est alors effectuée lors de cette phase de recherche. La place libre est, quant à elle, reconnue comme telle, après que le véhicule l'ait légèrement dépassée et ait déjà reconnu le deuxième obstacle la délimitant. Les vitesses mises en jeu dans ce cas sont voisines de 20km/h. De plus, l'environnement est souvent statique si on néglige le passage de quelques piétons. Le temps de traitement devient donc moins critique qu'une fonction en milieu routier, fonction fortement liée à la sécurité.

Contrairement à l'environnement routier, les obstacles se répartissent autour du véhicule et non plus seulement à l'avant. Le capteur n'est donc plus obligatoirement à l'avant du véhicule mais peut être déporté sur un côté ou se focaliser sur une certaine zone de l'environnement au prix de plusieurs capteurs si on veut surveiller l'ensemble de l'espace proche. Les solutions sont alors multiples : soit on utilise un seul capteur, soit on en utilise plusieurs. Dans le premier cas, cela implique des contraintes sur le champ de perception qui doit être élevé et défavorise

la résolution des données mais satisfait les contraintes de coûts et de compacité incontournables dans le milieu automobile. Dans le deuxième cas, les contraintes sur le capteur même peuvent être relâchées en gagnant en performances mais cela élève le coût et la complexité d'intégration du système.

Pour résumer, l'environnement urbain se caractérise par des obstacles proches (inférieurs à 10m) dont la répartition englobe l'ensemble de l'environnement du véhicule. Le caractère dynamique n'est plus un élément essentiel et peut même être supposé nul. A cela, il faut ajouter que la fonction d'aide au parking est liée au confort dont la criticité est moindre qu'une fonction liée à la sécurité du véhicule. Cette thèse fera alors l'hypothèse d'un monde statique, des travaux futurs devront combler cette lacune. Les chapitres 4 et 5 aborderont le problème de la spécification du système en fonction des choix possibles.

1.2 Modélisation d'une caméra

En vision par ordinateur, dès lors que l'on souhaite obtenir des informations métriques, il est nécessaire de calibrer la caméra. Le calibrage d'une seule caméra (applications monoculaires) revient à estimer ses paramètres intrinsèques ainsi que sa position par rapport au référentiel du monde. Configurer un capteur stéréoscopique revient à calibrer les deux caméras (paramètres intrinsèques de chacune des caméras) et la position et l'orientation relative des deux caméras.

De nombreux travaux ont été menés concernant le calibrage et la reconstruction tridimensionnelle avec un capteur stéréoscopique ou une caméra en mouvement. Les premières méthodes nécessitaient une mire de calibrage de géométrie parfaitement connue pour calibrer hors-ligne le système de vision [Toscani, 1987, Faugeras et al., 1987]. Les mires utilisées étaient des dièdres comme en figure 1.3[gauche] puis réduit à une seule face (figure 1.3[droite]). Les contraintes sur la précision de réalisation de la mire se sont relâchées avec [Garcia, 2001] pour la faire totalement disparaître [Cornille, 2005].

Les sections qui vont suivre vont reprendre la description des méthodes classiques de calibrage avec la modélisation associée du capteur dans le cas d'une caméra unique puis dans le cas d'un banc stéréoscopique. Nous verrons aussi les méthodes plus récentes basées sur les techniques issues de la photogrammétrie.

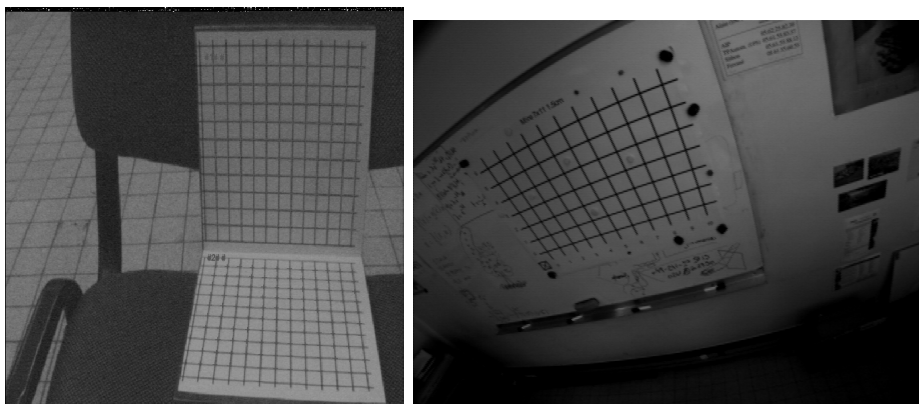


FIG. 1.3 – Mire composée d'un dièdre [gauche] et d'une simple grille [droite]

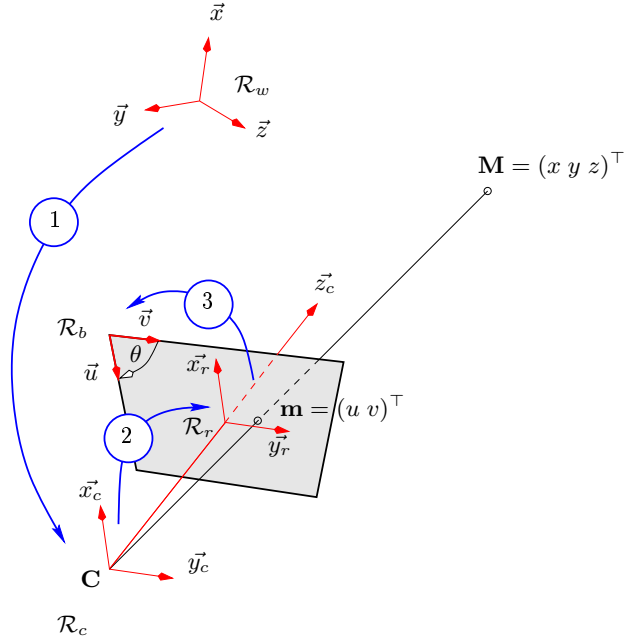
1.2.1 Projection perspective : modèle Sténopé ou *trou d'épingle*

FIG. 1.4 – Modèle sténopé. Le plan image est ici situé entre le centre optique et la scène, mais le modèle pourrait être représenté de manière équivalente, au sens géométrique, avec le centre optique situé entre la scène et le plan image. L'image produite serait alors simplement à l'envers, c'est le cas par exemple de la rétine humaine.

Le modèle sténopé représente une caméra par un **plan image** correspondant à la surface photosensible (CCD ou CMOS dans le cas d'une caméra numérique), un **centre optique** (origine 3D du repère caméra) et d'un point principal (point 2D projeté du centre optique sur le plan image). Un point de la scène est projeté suivant la droite le reliant au centre optique. Le **point image** se trouve à l'intersection de cette droite et du plan image. Le modèle géométrique classique d'une caméra est une projection perspective, résultat de la composition des trois transformations élémentaires [Faugeras, 1993] présentées en figure 1.4. La première est un simple changement de repère du point $\mathbf{M} = (x, y, z)^t$ exprimé dans le repère du monde $\mathcal{R}_w$ en un point $\mathbf{M}_c = (x_c, y_c, z_c)^t$ exprimé dans le repère de la caméra $\mathcal{R}_c$. Ces paramètres sont appelés **extrinsèques** et sont ceux de la transformation rigide entre $\mathcal{R}_w$ et $\mathcal{R}_c$:

$$\tilde{\mathbf{M}}_c \cong \underbrace{\begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}}_T \tilde{\mathbf{M}} \quad (1.1)$$

avec $\mathbf{T} \cong \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ 0^t & 1 \end{pmatrix}$

avec $\mathbf{R}$ matrice de rotation et $\mathbf{t}$ matrice de translation.

La deuxième transformation projette le point $\mathbf{M}_c$ sur le plan rétinien en un point $\mathbf{m}_r = (x_r, y_r)^t$, projection perspective de rapport la **distance focale** f . Cette projection s'écrit :

$$\tilde{\mathbf{m}}_r \cong \underbrace{\begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}}_{\mathbf{P}} \tilde{\mathbf{M}}_c \quad (1.2)$$

La troisième et dernière transformation décrit l'échantillonnage effectué du plan rétinien par la surface photosensible. Elle permet d'exprimer un point rétinien en un point image de coordonnées pixel $\mathbf{m} = (u, v)^t$ [Horaud et al., 1995] :

$$\tilde{\mathbf{m}} \cong \underbrace{\begin{pmatrix} k_u & 0 & u_0 \\ 0 & k_v & v_0 \\ 0 & 0 & 1 \end{pmatrix}}_{\mathbf{A}} \tilde{\mathbf{M}}_r \quad (1.3)$$

en supposant une orthogonalité parfaite des lignes et des colonnes de l'image, hypothèse rarement fautive. Les paramètres k_u et k_v sont les facteurs d'échelle vertical et horizontal (en pixel/unité de longueur), u_0 et v_0 sont les coordonnées dans l'image du point principal. $\mathbf{A}$ est une transformation affine dont les quatre paramètres non nuls sont appelés **intrinsèques**.

La composition de ces trois transformations permet l'expression d'un point image en fonction de ses coordonnées dans le repère du monde :

$$\tilde{\mathbf{m}} \cong \underbrace{\overline{\mathbf{AP}}}_{\mathbf{K}} \tilde{\mathbf{TM}} \quad (1.4)$$

Il est d'usage d'écrire $\mathbf{K}$, **matrice des paramètres intrinsèques**, avec l'expression suivante :

$$\begin{aligned} \mathbf{K} &\cong \overline{\mathbf{AP}} \\ &\cong \begin{pmatrix} k_u & 0 & u_0 \\ 0 & k_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \\ &\cong \underbrace{\begin{pmatrix} k_u f & 0 & u_0 \\ 0 & k_v f & v_0 \\ 0 & 0 & 1 \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}}_{\mathbf{P}} \\ &\cong \begin{pmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \end{aligned} \quad (1.5)$$

Les équations de projection d'un point du monde $\mathbf{M} = (x, y, z)^t$ en un point image $\mathbf{m} = (u, v)^t$ s'écrivent alors :

$$u = \alpha_u \frac{r_{11}x + r_{12}y + r_{13}z + t_x}{r_{31}x + r_{32}y + r_{33}z + t_z} + u_0 \quad (1.6)$$

$$v = \alpha_v \frac{r_{21}x + r_{22}y + r_{23}z + t_y}{r_{31}x + r_{32}y + r_{33}z + t_z} + v_0 \quad (1.7)$$

1.2.2 Distorsions

Un rayon optique n'intersecte généralement pas le plan image au point déterminé par approximation linéaire. Les écarts entre cette approximation et la réalité optique ne sont généralement pas négligeables lorsque la distance du point objet à l'axe optique est grande et lorsque les angles d'inclinaison des rayons incidents sont plus petits [Perez, 1994]. A cela il faut ajouter une grande difficulté d'assembler toutes les lentilles d'une optique de façon parfaitement parallèle, les centres optiques parfaitement alignés. L'approche classique de la littérature [Weng et al., 1992, Zeller, 1996, Garcia, 2001] consiste à modéliser les distorsions par un modèle paramétrique en complétant le modèle sténopé par des paramètres de correction de distorsion. Depuis, [Cornille, 2005, Cornille et al., 2005] considère une approche non paramétrique mieux adaptée au cas des systèmes optiques complexes. Nous verrons dans les deux paragraphes suivants les principes de chacune des méthodes.

Modélisation paramétrique

Si on pose $\mathbf{m}_r$ le point idéal donné par le modèle sténopé et $\mathbf{m}_d$ le point réel *distordu*, la relation entre ces deux points s'écrit :

$$\mathbf{m}_d = \mathbf{m}_r + \Delta \quad (1.8)$$

où $\Delta = (\delta_x, \delta_y)^t$ est appelé **vecteur d'aberration** ou **distorsion**. Ce vecteur est lui-même la somme de plusieurs distorsions [Weng et al., 1992] :

$$\Delta = \Delta_R + \Delta_D + \Delta_P$$

où :

- $\Delta_R = (\delta_{R_x}, \delta_{R_y})^t$, **distorsion radiale**, symétrique par rapport au point central due à la courbure des lentilles,
- $\Delta_D = (\delta_{D_x}, \delta_{D_y})^t$, **distorsion de décentrage**, due aux imperfections d'alignement des centres optiques des différentes lentilles,
- $\Delta_P = (\delta_{P_x}, \delta_{P_y})^t$, **distorsion prismatique**, due à un mauvais parallélisme des lentilles entre elles.

En pratique, les distorsions de décentrage et prismatique sont négligeables et leurs estimations peuvent rendre le calibrage instable. Dans la suite de ce document, par abus de langage nous parlerons de **distorsion** pour nommer la distorsion radiale.

Les équations de la distorsion radiale représentant le déplacement du point le long d'un rayon centré sur le point principal s'écrivent :

$$\delta_{R_x} = x \sum_{k=1}^{\infty} r_k (x_r^2 + y_r^2)^k \quad (1.9)$$

$$\delta_{R_y} = y \sum_{k=1}^{\infty} r_k (x_r^2 + y_r^2)^k \quad (1.10)$$

avec r_k appelés **coefficients de distorsion**. Un ordre 3 convient largement à la plupart des caméras pour la modélisation des aberrations géométriques. Les équations s'écrivent alors :

$$\delta_{R_x} = x \left(r_1 (x_r^2 + y_r^2)^1 + r_2 (x_r^2 + y_r^2)^2 + r_3 (x_r^2 + y_r^2)^3 \right) \quad (1.11)$$

$$\delta_{R_y} = y \left(r_1 (x_r^2 + y_r^2)^1 + r_2 (x_r^2 + y_r^2)^2 + r_3 (x_r^2 + y_r^2)^3 \right) \quad (1.12)$$

Prise en compte des distorsions dans le modèle caméra

La prise en compte des distorsions induit l'ajout d'une transformation non linéaire dans le modèle de caméra. Un point de l'image distordu m n'est plus directement lié à ses coordonnées rétinienne idéales $\mathbf{m}_r$, mais au point distordu $\mathbf{m}_d$ coordonnées rétinienne distordues. Le système s'écrit alors :

$$\tilde{\mathbf{m}} \cong \mathbf{A}\tilde{\mathbf{m}}_d \implies \begin{cases} u &= \alpha_u x_d + u_0 \\ v &= \alpha_v y_d + v_0 \end{cases} \quad (1.13)$$

A partir de 1.8 et 1.12 nous pouvons réécrire les équations du point rétinien distordu $\mathbf{m}_d$ en fonction de ses coordonnées idéales $\mathbf{m}_r$:

$$\mathbf{m}_d = \mathbf{m}_r + \Delta \implies \begin{cases} x_d &= x_r + x_r \left(r_1 (x_r^2 + y_r^2)^1 + r_2 (x_r^2 + y_r^2)^2 + r_3 (x_r^2 + y_r^2)^3 \right) \\ y_d &= y_r + y_r \left(r_1 (x_r^2 + y_r^2)^1 + r_2 (x_r^2 + y_r^2)^2 + r_3 (x_r^2 + y_r^2)^3 \right) \end{cases} \quad (1.14)$$

Par contre, les relations entre les coordonnées rétinienne $\mathbf{m}_r$ et le point 3D du monde $\mathbf{M}_c$ restent inchangées.

Notons $\mathbf{k}$ le vecteur des paramètres intrinsèques de la matrice $\mathbf{K}$ et $\mathbf{d}$ le vecteur des coefficients de distorsion :

$$\begin{aligned} \mathbf{k} &= (\alpha_u \quad \alpha_v \quad u_0 \quad v_0) \\ \mathbf{d} &= (r_1 \quad r_2 \quad r_3) \end{aligned}$$

Le modèle de caméra peut maintenant être vu comme une fonction f non linéaire décrite par :

$$m = f(\mathbf{k}, \mathbf{d}, \mathbf{R}, \mathbf{t}, \mathbf{M}) \quad (1.15)$$

Modélisation non paramétrique

Dans le cas de systèmes optiques complexes, tels que les microscopes électroniques à balayage, [Cornille, 2005] a montré qu'il est préférable de modéliser la distorsion de façon non-paramétrique en utilisant des fonctions splines [Nielsen, 1998]. L'objectif est alors de déterminer la fonction de distorsion qui traduit au mieux la façon dont l'image idéale est distordue.

L'estimation de la fonction de correction de distorsion est intégrée au processus de calibrage basée sur une méthode d'ajustement de faisceaux. La fonction est estimée itérativement en minimisant les erreurs de reprojection. La convergence de la méthode est accélérée par l'utilisation d'une méthode de relaxation [Kulsrud, 1961, Ostrowski, 1953].

En pratique, afin d'avoir un nombre suffisant de points à l'estimation des splines, la mire traditionnelle (dont le nombre de points est trop faible), est remplacée par une mire de mouchetis déplacée librement dans le champ de vision des caméras. L'image est divisée en $m \times n$ rectangles, appelés *patches*, pour lesquels les distorsions sont approximées par une surface spline bi-cubique.

Cette méthode se montre d'une redoutable précision comparée aux méthodes classiques, dites paramétriques. Elle est donc particulièrement adaptée aux applications nécessitant une grande précision. L'inconvénient réside dans l'utilisation d'une mire qui doit couvrir une grande partie du champ de vision des caméras. Nous n'appliquerons donc pas cette technique dans le cadre de cette thèse.

1.2.3 Calibrage d'une caméra

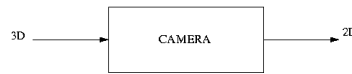


FIG. 1.5 – Une caméra

Le calibrage d'une caméra revient à estimer la fonction de transfert qui transforme un point $3D$ de la scène en un point $2D$ de l'image (cf. figure 1.5), on parle alors de **calibrage fort**.

Les méthodes développées jusqu'à ces dernières années exploitent des points tridimensionnels répartis sur un objet appelé mire de calibrage. Les toutes premières méthodes utilisaient une mire de calibrage volumique précise (cf. figure 1.3[gauche]) en forme de dièdre utilisée pour sa répartition des points dans l'espace. L'estimation de tous les paramètres était alors possible à partir d'une seule image. En revanche, la précision de construction requise était trop importante et son utilisation était peu aisée. Les méthodes suivantes ont alors utilisé des mires planes précises dont les déplacements étaient connus, à l'aide d'une table de translation micrométrique. L'estimation des paramètres nécessitait alors au moins deux images mais la précision de la construction était plus facile à garantir (par procédé de microlithographie) et la manipulation plus aisée du fait de son faible encombrement. Les méthodes les plus récentes utilisent maintenant des mires imprécises, de faible coût de réalisation et beaucoup plus faciles à manipuler.

Les deux paragraphes suivants présentent les méthodes dites *classiques* et les méthodes *photogrammétriques* permettant une réestimation des points de la mire.

Méthode classique

L'approche classique consiste à estimer les paramètres de la caméra à partir de la géométrie d'une mire connue avec exactitude. Les paramètres à estimer sont alors les seuls paramètres intrinsèques (4 paramètres intrinsèques et 3 paramètres de distorsion si on se limite à une correction de la distorsion radiale d'ordre 3) et extrinsèques (3 paramètres pour l'orientation et 3 paramètres pour la position).

La figure 1.6 présente une séquence de huit images d'une mire de calibration composée de 24 points, nœuds d'intersections de droites horizontales et verticales. Plusieurs primitives peuvent être utilisées pour l'extraction de points caractéristiques : des croix avec la méthode *Peuchot* [Peuchot, 1994], des centres d'ellipse avec l'estimateur de *Lavest* [Lavest et al., 1999] ou encore la méthode de *Brand* [Brand, 1995] et d'autres méthodes décrites dans [Blaszka, 1997].

Une fois les points de mire extraits, l'estimation des paramètres du modèle de caméra est posée comme un processus de minimisation. La fonction à minimiser est généralement la somme des erreurs quadratiques entre les coordonnées des points extraits dans les images et les coordonnées calculées par le modèle de caméra pour chaque point correspondant de la mire. En pratique, des algorithmes d'optimisation non-linéaire tels que l'algorithme de *Levenberg-Marquardt* [Levenberg, 1944] sont utilisés couplés à une estimation initiale des paramètres par des méthodes linéaires telles que celles de [Ravn et al., 1993], [Tsai, 1987] ou [Zhang, 1998].

L'inconvénient majeur de ces méthodes est la nécessité d'une connaissance précise de la

géométrie de la mire. Le paragraphe suivant lève cette contrainte par une méthode photogrammétrique présentée dans [Garcia, 2001].

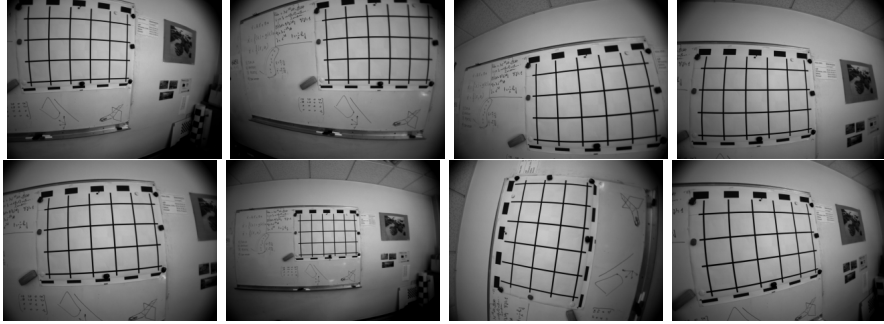


FIG. 1.6 – Une séquence d’image de mire pour le calibrage d’une caméra

Méthode photogrammétrique

L’utilisation de méthodes photogrammétriques dans le processus de calibration rend possible l’utilisation de mires imprécises [Lavest et al., 1999] pour estimer en même temps tous les paramètres du modèle de caméra ainsi que les points tridimensionnels de la mire. La fonction à minimiser s’enrichit alors des points de la mire pour s’écrire :

$$\theta = \min_{\theta} \sum_{i=1}^n \sum_{j=1}^p \|\hat{\mathbf{m}}_i^j - f(\mathbf{k}, \mathbf{d}, \mathbf{R}_i, \mathbf{t}_i, \mathbf{M}_j)\|^2 \quad (1.16)$$

avec :

- $\theta = f(\mathbf{k}, \mathbf{d}, \mathbf{R}_{1..n}, \mathbf{t}_{1..n}, \mathbf{M}_{1..p})$,
- $\hat{\mathbf{m}}_i^j$ coordonnées mesurées de la projection du j -ème point ($j = 1..p$) de la i -ème vue ($i = 1..n$) sur le plan image de la caméra,
- $\mathbf{k}$ vecteur des paramètres intrinsèques,
- $\mathbf{R}_i$ orientation de la caméra de la i -ème vue,
- $\mathbf{t}_i$ position de la caméra de la i -ème vue,
- $\mathbf{M}_j$, j -ème point 3D de la mire.

De la même manière que les méthodes de calibration classique, cette minimisation est réalisée en utilisant un algorithme d’optimisation non linéaire tel que l’algorithme de *Levenberg-Marquardt* dont les estimées de $\mathbf{k}$, $\mathbf{R}_i$ et $\mathbf{t}_i$ sont fournies par la méthode de *Ravn*. Le modèle 3D utilisé pour créer la mire est utilisé comme estimée initiale. La précision de ce modèle n’est donc plus une contrainte puisque l’ensemble des points est réestimé. Les paramètres de distorsion sont, quant à eux, initialisés à zéro.

Cette minimisation conduit à une solution définie à une similitude près, c’est-à-dire à un déplacement rigide près ainsi qu’à un facteur d’échelle près. [Garcia, 2001] démontre l’unicité de la solution en fixant deux points du modèle dans le processus de minimisation et en contraignant un troisième, non colinéaire aux deux premiers, en fixant $z = 0$.

Cette méthode basée sur une approche photogrammétrique rend l’estimation plus stable et plus facile à réaliser que les méthodes classiques de calibration. Nous verrons en section suivante comment on peut l’appliquer à un banc de stéréovision.

1.3 Modélisation du capteur stéréoscopique

La fonction de transfert liée à la description de la caméra dans la section précédente montre qu'à tout point de l'espace tridimensionnel est associé un point de l'espace bidimensionnel de l'image. La troisième dimension est alors perdue et aucune information de profondeur n'est donc disponible. C'est-à-dire qu'à un point de l'image est associée une infinité de points $3D$, tous le long de la droite tridimensionnelle passant par le centre optique et le point image. Par **triangulation** grâce à l'utilisation d'un deuxième capteur, il est alors possible de retrouver cette information de profondeur en calculant l'intersection des deux droites tridimensionnelles. Par conséquent, il est nécessaire d'exprimer la géométrie des deux capteurs dans un même référentiel. C'est pourquoi il est important de connaître la géométrie du banc de stéréovision.

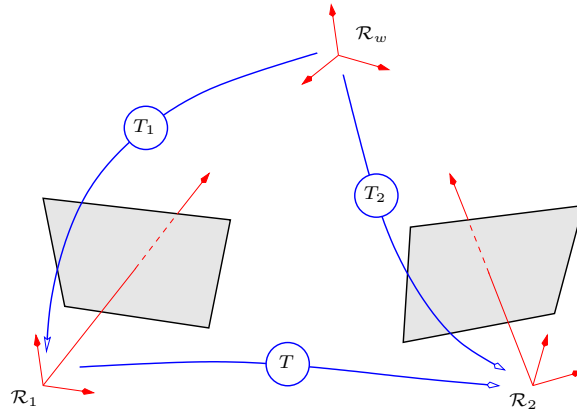


FIG. 1.7 – les trois référentiels tridimensionnels du capteur de stéréovision

Notons $\mathcal{R}_w$ le référentiel de la scène et $\mathcal{R}_1$ et $\mathcal{R}_2$ respectivement les référentiels de la caméra gauche et de la caméra droite. La figure 1.7 illustre ces trois référentiels ainsi que les trois transformations rigides permettant d'exprimer un point dans un autre référentiel. Suivant ces notations nous pouvons écrire :

$$\begin{aligned}\tilde{\mathbf{M}}_{c1} &\cong \mathbf{T}_1 \tilde{\mathbf{M}} \\ \tilde{\mathbf{M}}_{c2} &\cong \mathbf{T}_2 \tilde{\mathbf{M}} \\ \tilde{\mathbf{M}}_{c2} &\cong \mathbf{T} \tilde{\mathbf{M}}_{c1}\end{aligned}\tag{1.17}$$

Les trois transformations $\mathbf{T}$, $\mathbf{T}_1$ et $\mathbf{T}_2$ sont donc liées et il est possible d'écrire l'une par rapport à l'autre :

$$\tilde{\mathbf{T}}_1 \cong \mathbf{T}^{-1} \mathbf{T}_2 \tag{1.18}$$

$$\tilde{\mathbf{T}}_2 \cong \mathbf{T} \mathbf{T}_1 \tag{1.19}$$

$$\tilde{\mathbf{T}} \cong \mathbf{T}_2 \mathbf{T}_1^{-1} \tag{1.20}$$

Un point $\mathbf{M}$ de la scène visible dans les deux caméras donne alors deux points $\mathbf{m}_1$ et $\mathbf{m}_2$ dans chacune des deux caméras. L'utilisation du modèle géométrique de la caméra et les relations de

dépendances entre les trois repères $\mathcal{R}_w$, $\mathcal{R}_1$ et $\mathcal{R}_2$ nous permettent d'écrire :

$$\tilde{\mathbf{m}}_1 \cong \mathbf{K}_1 \mathbf{T}_1 \tilde{\mathbf{M}} \quad (1.21)$$

$$\begin{aligned} \tilde{\mathbf{m}}_2 &\cong \mathbf{K}_2 \mathbf{T}_2 \tilde{\mathbf{M}} \\ &\cong \mathbf{K}_2 \mathbf{T} \mathbf{T}_1 \tilde{\mathbf{M}} \end{aligned} \quad (1.22)$$

1.3.1 Géométrie du capteur de stéréovision, géométrie épipolaire

Dans cette section de description de la géométrie du capteur stéréoscopique, nous cherchons à lier géométriquement les deux points $2D$ dans le plan rétinien $\mathbf{m}_{r1} = (x_{r1}, y_{r1})^t$ et $\mathbf{m}_{r2} = (x_{r2}, y_{r2})^t$ des points images $\mathbf{m}_1$ et $\mathbf{m}_2$, points homologues correspondants au point $3D$ $\mathbf{M}_{c2} = (x_{c2}, y_{c2}, z_{c2})^t$. Pour garder une notation matricielle plus aisée, nous supposons pour valeur de focale $f = 1$ sans perte de généralité puisque, comme le montre [Horaud et al., 1995], la focale disparaît au cours des calculs. Nous savons que l'expression de $\mathbf{m}_{r2}$ est :

$$\mathbf{m}_{r2} = \begin{pmatrix} x_{r2} \\ y_{r2} \end{pmatrix} = \frac{1}{z_{c2}} \begin{pmatrix} x_{c2} \\ y_{c2} \end{pmatrix} \quad (1.23)$$

Notons $\mathbf{r}_i^t$ le vecteur ligne correspondant à la i -ème ligne de la rotation $\mathbf{R}$ de la transformation $\mathbf{T}$. Des équations 1.17 et 1.23 nous pouvons exprimer les coordonnées dans le plan rétinien droit d'un point de l'espace défini dans le repère $\mathcal{R}_{c1}$ de la caméra gauche :

$$x_{r2} = \frac{x_{c2}}{z_{c2}} \quad (1.24)$$

$$= \frac{r_1^t M_{c1} + t_x}{r_3^t M_{c1} + t_z} \quad (1.25)$$

$$y_{r2} = \frac{y_{c2}}{z_{c2}} \quad (1.26)$$

$$= \frac{r_2^t M_{c1} + t_y}{r_3^t M_{c1} + t_z} \quad (1.27)$$

En utilisant $\tilde{\mathbf{m}}_{r1} \cong \mathbf{P} \tilde{\mathbf{M}}_{c1}$, l'expression précédente devient :

$$x_{r2} = \frac{z_{c1} r_1^t \tilde{\mathbf{m}}_{r1} + t_x}{z_{c1} r_3^t \tilde{\mathbf{m}}_{r1} + t_z} \quad (1.28)$$

$$y_{r2} = \frac{z_{c1} r_2^t \tilde{\mathbf{m}}_{r1} + t_y}{z_{c1} r_3^t \tilde{\mathbf{m}}_{r1} + t_z} \quad (1.29)$$

Ces équations représentent le lien entre un point de l'image droite et son **stéréo-correspondant** dans l'image gauche en fonction des paramètres intrinsèques et extrinsèques du capteur stéréoscopique et de la *profondeur* du point $\mathbf{M}$ dans le référentiel de la caméra gauche. En éliminant z_{c1} de ces deux équations, nous pouvons déduire l'expression suivante reliant $\tilde{\mathbf{m}}_{r1}$ et $\tilde{\mathbf{m}}_{r2}$:

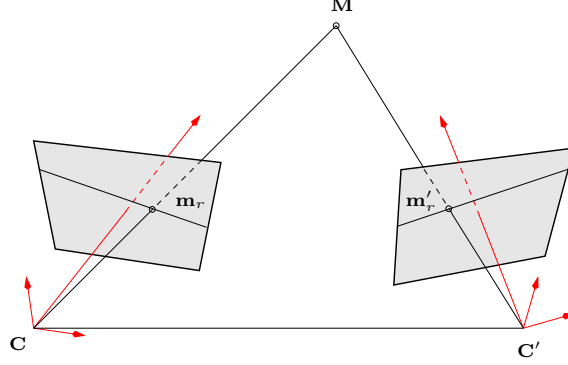
$$(t_z r_2^t \tilde{\mathbf{m}}_{r1} - t_y r_3^t \tilde{\mathbf{m}}_{r1}) x_{r2} + (t_x r_3^t \tilde{\mathbf{m}}_{r1} - t_z r_1^t \tilde{\mathbf{m}}_{r1}) y_{r2} = t_x r_2^t \tilde{\mathbf{m}}_{r1} - t_y r_1^t \tilde{\mathbf{m}}_{r1} \quad (1.30)$$

Cette équation représente le lieu des projetés dans le plan rétinien droit pouvant correspondre à un projeté $\mathbf{m}_{r1}$ dans le plan rétinien gauche. C'est l'équation de la droite appelée **ligne épipolaire** droite (cf. figure 1.3.1). L'ensemble des droites épipolaires d'une image se coupe en un point unique appelé **épipole** droit $\mathbf{e}_2 = (x_{e2}, y_{e2})^t$ de coordonnées :

$$x_{e2} = \frac{t_x}{t_z} \quad (1.31)$$

$$y_{e2} = \frac{t_y}{t_z} \quad (1.32)$$

Ce point est aussi le projeté du centre optique de la caméra gauche sur le plan rétinien de la caméra droite. Nous pourrions aussi définir la droite épipolaire gauche et l'épipole gauche de la même manière.



Matrice essentielle

De l'équation 1.30, nous pouvons tirer l'expression suivante :

$$(x_{r2} \ y_{r2} \ 1) \begin{pmatrix} a \\ b \\ c \end{pmatrix} = 0 \quad (1.33)$$

avec a , b , et c fonctions de la transformation $\mathbf{T}$ et du point $\mathbf{m}_{r1}$:

$$a = (t_y r_{31} - t_z r_{21})x_{r1} + (t_y r_{32} - t_z r_{22})y_{r1} + (t_y r_{33} - t_z r_{23}) \quad (1.34)$$

$$b = (t_z r_{11} - t_x r_{31})x_{r1} + (t_z r_{12} - t_x r_{32})y_{r1} + (t_z r_{13} - t_x r_{33}) \quad (1.35)$$

$$c = (t_x r_{21} - t_y r_{11})x_{r1} + (t_x r_{22} - t_y r_{12})y_{r1} + (t_x r_{23} - t_y r_{13}) \quad (1.36)$$

d'où la forme matricielle suivante :

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \underbrace{\begin{pmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{pmatrix}}_{(\mathbf{t})_{\times}} \underbrace{\begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}}_{\mathbf{R}} \begin{pmatrix} x_{r1} \\ y_{r1} \\ 1 \end{pmatrix} \quad (1.37)$$

$\mathbf{E}$

La matrice $(\mathbf{t})_{\times}$ est une matrice antisymétrique de rang deux, et $\mathbf{E} = (\mathbf{t})_{\times} \mathbf{R}$. En utilisant cette équation, la droite épipolaire 1.33 peut se mettre sous la forme suivante :

$$\tilde{\mathbf{m}}_{r2}^t \mathbf{E} \tilde{\mathbf{m}}_{r1} = 0 \quad (1.38)$$

La matrice $\mathbf{E}$ est appelée **matrice essentielle** et décrit la géométrie épipolaire gauche-droite. La géométrie épipolaire droite-gauche s'écrit simplement :

$$\tilde{\mathbf{m}}_{r1}^t \mathbf{E}^t \tilde{\mathbf{m}}_{r2} = 0 \quad (1.39)$$

Matrice fondamentale

La matrice essentielle décrit la géométrie épipolaire dans le plan rétinien. Nous allons maintenant écrire les mêmes relations dans le repère image. Pour alléger les notations, posons $(\mathbf{A}^{-1})^t \equiv \mathbf{A}^{-t}$. A partir de l'équation 1.38 et des relations $\tilde{\mathbf{m}}_1 = \mathbf{A}_1 \tilde{\mathbf{m}}_{r1}$ et $\tilde{\mathbf{m}}_2 = \mathbf{A}_2 \tilde{\mathbf{m}}_{r2}$, nous obtenons :

$$\tilde{\mathbf{m}}_2^t \underbrace{\mathbf{A}_2^{-t} \mathbf{E} \mathbf{A}_1^{-1}}_{\mathbf{F}} \tilde{\mathbf{m}}_1 = 0 \quad (1.40)$$

La matrice $\mathbf{F}$ est appelée **matrice fondamentale**. A partir de deux points $\mathbf{m}_1$ et $\mathbf{m}_2$, [Zhang, 1998] calcule la distance algébrique d_1 entre le point $\mathbf{m}_2$ et sa droite épipolaire associée :

$$d_1 = \frac{\tilde{\mathbf{m}}_2^t \mathbf{D} \tilde{\mathbf{m}}_1}{\sqrt{\tilde{\mathbf{m}}_1^t \mathbf{F}^t \mathbf{Z} \mathbf{Z}^t \mathbf{F} \tilde{\mathbf{m}}_1}} \quad \text{avec} \quad \mathbf{Z} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} \quad (1.41)$$

et réciproquement :

$$d_2 = \frac{\tilde{\mathbf{m}}_2^t \mathbf{D} \tilde{\mathbf{m}}_1}{\sqrt{\tilde{\mathbf{m}}_2^t \mathbf{F}^t \mathbf{Z} \mathbf{Z}^t \mathbf{F} \tilde{\mathbf{m}}_2}} \quad (1.42)$$

Triangulation [Hartley et al., 1997] :

Posons :

$$\mathbf{T}_1 \cong \begin{pmatrix} r_1^{1t} & t_x^1 \\ r_2^{1t} & t_y^1 \\ r_3^{1t} & t_z^1 \\ 0^t & 1 \end{pmatrix} \quad \text{et} \quad \mathbf{T}_2 \cong \begin{pmatrix} r_1^{2t} & t_x^2 \\ r_2^{2t} & t_y^2 \\ r_3^{2t} & t_z^2 \\ 0^t & 1 \end{pmatrix} \quad (1.43)$$

Les équations 1.21 et 1.22 expriment les coordonnées des deux points $\mathbf{m}_1 = (u_1, v_1)^t$ et $\mathbf{m}_2 = (u_2, v_2)^t$, projections d'un point $\mathbf{M}$ dans l'image gauche et droite. Nous pouvons écrire le système suivant :

$$u_1 = \alpha_u^1 \frac{r_1^{1t} \mathbf{M} + t_x^1}{r_3^{1t} \mathbf{M} + t_z^1} + u_0^1 \quad v_1 = \alpha_v^1 \frac{r_2^{1t} \mathbf{M} + t_y^1}{r_3^{1t} \mathbf{M} + t_z^1} + u_0^1 \quad (1.44)$$

$$u_2 = \alpha_u^2 \frac{r_1^{2t} \mathbf{M} + t_x^2}{r_3^{2t} \mathbf{M} + t_z^2} + u_0^2 \quad v_2 = \alpha_v^2 \frac{r_2^{2t} \mathbf{M} + t_y^2}{r_3^{2t} \mathbf{M} + t_z^2} + u_0^2 \quad (1.45)$$

Il est donc possible de calculer la position du point $\mathbf{M}$ à partir des coordonnées des deux points de l'image et de la connaissance parfaite de la géométrie du capteur.

1.3.2 calibration du banc

La calibration d'un banc stéréoscopique s'intéresse à la connaissance des paramètres intrinsèques $\mathbf{K}_1$ et $\mathbf{K}_2$ de chacune des caméras ainsi qu'à la position et l'orientation relative des deux caméras, transformation rigide notée $\mathbf{T}$. Le but de la calibration est de rendre possible la triangulation afin d'effectuer des mesures sur l'environnement perçu.

Nous verrons dans les paragraphes suivants les différentes manières d'estimer cette transformation $\mathbf{T}$. Les méthodes classiques calibrent généralement les deux caméras indépendamment puis estiment la transformation $\mathbf{T}$. Une autre approche, présentée dans [Garcia, 2001], consiste à prendre en compte globalement le problème de la calibration et d'estimer en même temps $\mathbf{T}$ et les paramètres intrinsèques de chacune des caméras.

calibration classique dit *linéaire*

Les méthodes classiques consistent à calibrer chacune des caméras indépendamment avec les méthodes présentées en section 1.2.3 pour déterminer les paramètres intrinsèques $\mathbf{K}_i (i = 1..2)$, extrinsèques $\mathbf{T}_j^i (j = 1..n \text{ pour } n \text{ vues et } i = 1..2)$ et les coefficients de distorsion. $\mathbf{T}$ peut alors être calculée avec n'importe quelle paire de matrices des paramètres extrinsèques à l'aide de 1.46 :

$$\tilde{\mathbf{T}} \cong \mathbf{T}_2 \mathbf{T}_1^{-1} \quad (1.46)$$

Cette méthode présente l'inconvénient de reposer sur le choix arbitraire d'une paire de matrices extrinsèques. Différentes heuristiques peuvent pousser à choisir telle ou telle paire comme la paire à la plus faible erreur globale de reprojection des points de la mire dans les deux images. D'autres méthodes consistent à tirer partie de la redondance apportée par toutes les paires. [Lorusso et al., 1997] ont testé plusieurs méthodes et ont montré qu'aucune technique particulière n'était plus performante qu'une autre.

calibration dit *global*

La méthode globale présentée par [Garcia, 2001] estime $\mathbf{T}$ en utilisant l'équation 1.19. La fonctionnelle à minimiser s'écrit alors :

$$\theta = \min_{\theta} \sum_{i=1}^n \sum_{j=1}^p [\delta_1(i, j) + \delta_2(i, j)] \quad (1.47)$$

avec :

$$\begin{aligned} \theta &= (\mathbf{k}_1, \mathbf{k}_2, \mathbf{d}_1, \mathbf{d}_2, \mathbf{R}_{1..n}, \mathbf{t}_{1..n}, \mathbf{R}, \mathbf{t}, \mathbf{M}_{1..p}) \\ \delta_1(i, j) &= \|\hat{\mathbf{m}}_{1_i}^j - f(\mathbf{k}_1, \mathbf{d}_1, \mathbf{R}_i, \mathbf{t}_i, \mathbf{M}_j)\|^2 \\ \delta_2(i, j) &= \|\hat{\mathbf{m}}_{2_i}^j - g(\mathbf{k}_2, \mathbf{d}_2, \mathbf{R}_i, \mathbf{t}_i, \mathbf{R}, \mathbf{t}, \mathbf{M}_j)\|^2 \end{aligned}$$

La fonction f est similaire à la fonction g à la différence près qu'elle détermine la matrice des paramètres extrinsèques $\mathbf{T}_{2i}$ en utilisant l'équation 1.19 avec $\mathbf{R}_i$, $\mathbf{t}_i$, $\mathbf{R}$ et $\mathbf{t}$. L'algorithme de *Levenberg Marquardt* est utilisé pour résoudre ce problème d'estimation non linéaire.

[Garcia, 2001] montre que la méthode globale améliore beaucoup la précision de reconstruction tridimensionnelle par rapport aux méthodes classiques dites linéaires.

calibration photogrammétrique avec mire de mouchetis

La calibration par méthode photogrammétrique sans mire, présentée dans [Cornille, 2005], est une approche dite *globale* de la calibration dont la mire de points est remplacée par une mire de mouchetis. La calibration sur un grand nombre de points (typiquement 3000 points au lieu de 64) permet alors de gagner un facteur 8 sur les erreurs de reprojection.

1.4 Stéréovision

La stéréovision met en correspondance des primitives entre les différentes vues des caméras. Par exemple, ces primitives peuvent être des segments, on parle alors de stéréovision éparse, ou alors des points, la stéréovision est dite "dense". Une fois les mises en correspondance effectuées, le

calcul de distance, appelée disparité, entre les deux appariements nous donne, par triangulation, l'information de profondeur du point ou région observé. Nous verrons dans le paragraphe suivant le détail d'une méthode de stéréovision dense s'appuyant sur la géométrie épipolaire du capteur vu précédemment.

1.4.1 Description algorithmique

Le but de la stéréovision dense est la mise en correspondance de points entre les deux images afin de calculer la position 3D du point perçu par les deux caméras. Nous avons vu dans la section précédente qu'il était possible, par estimation de la géométrie épipolaire, de calculer l'équation de la droite sur laquelle un point de l'image se trouve dans l'autre image. La recherche des stéréocorrespondants est alors accélérée puisque limitée aux droites épipolaires au lieu de rechercher dans l'image entière. Il est encore possible d'améliorer les performances en transformant les images de manière à rendre les épipolaires horizontales, c'est-à-dire confondues avec les lignes de l'image (cf. figure 1.9[haut]). [Lasserre et al., 1995] présentent une méthode de **rectification** permettant de minimiser les déformations entre images d'origine et images rectifiées.

L'algorithme général de la méthode de stéréovision par corrélation est présenté en figure 1.8, appliqué à une paire d'images droite et gauche. Cet algorithme comprend deux phases, l'une *en-ligne* et l'autre *hors-ligne*. Cette dernière représente la phase de calibration du système exposée dans la section précédente. Le résultat correspond à l'ensemble des paramètres de calibration utilisé pour la rectification des images et pour la triangulation. Nous verrons en chapitre 2 qu'il est possible d'intégrer cette phase aux calculs en-ligne de manière à tenir à jour les paramètres de géométrie du capteur.

La phase *en-ligne* requiert trois étapes :

1. **Prétraitements** : avant de rechercher les appariements entre pixels des deux images, plusieurs prétraitements sont réalisés séparément sur les deux images comme la réduction, le filtrage, la rectification et la correction des distorsions. En plus de ces prétraitements, des calculs propres au critère de corrélation utilisé peuvent être nécessaires comme le calcul des images *Census* ou le calcul des images des moyennes et des variances pour le score ZNCC (cf. paragraphes suivants).
2. **Calcul des scores de corrélation** : cette étape est la plus longue, elle peut être optimisée par différents traitements annexes.
3. **Traitements sur la disparité** : pour chacune des courbes de corrélation obtenues, la recherche de la disparité optimale permet de déterminer le point homologue. Afin de valider le choix du point, un test de corrélation droite-gauche est effectué. Un filtrage peut aussi être effectué sur la carte de disparité (image dont la valeur d'un pixel équivaut à la disparité trouvée) complète afin d'éliminer des points isolés par des opérateurs morphologiques simples.

Les lignes sont traitées successivement et pour chaque point de l'image de droite, son homologue est recherché dans l'image de gauche, sur la même ligne image et dans un intervalle $[d_{min}..d_{max}]$ appelé **intervalle de disparité**. Ce dernier est fonction de la configuration stéréo et de la plage de distance que l'on veut observer dans l'environnement. Généralement, $d_{min} = 0$ pour un point à l'infini et $d_{max} = f(dist_{min})$ avec :

$$f(dist_{min}) = \alpha_v \frac{base}{dist_{min}} \quad (1.48)$$

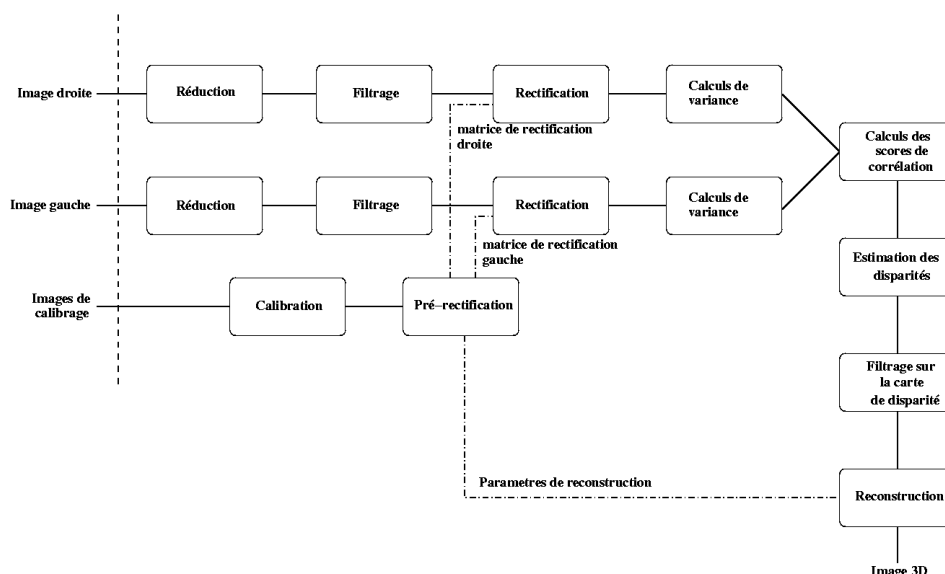


FIG. 1.8 – Algorithme général de la stéréovision

base étant la distance inter-caméras. L'écart est maximal pour une distance minimale des points appariés (cf. figure 1.10).

Pour savoir quel pixel, dans cet intervalle, est le correspondant du point étudié, on utilise un critère de ressemblance local sur les niveaux de gris dans un voisinage carré, centré sur le pixel analysé. Ce voisinage est appelé **fenêtre de corrélation** dont la taille est un paramètre important de l'algorithme et varie généralement entre 9×9 à 13×13 .

Différents critères numériques de corrélation ont été proposés comme critère de ressemblance entre deux fenêtres de corrélation, une prise autour du pixel $\mathbf{m}_1$ de l'image gauche et l'autre prise autour d'un pixel $\mathbf{m}_2$, candidat pris sur la même ligne de l'image droite :

- **SSD : Sum of Squared Differences** ou somme des différences quadratiques entre termes correspondants des deux fenêtres,
- **SAD : Sum of Absolute Differences** ou somme de la valeur absolue des différences entre termes correspondants des deux fenêtres,
- **ZSSD : Zero mean Sum of Squared Differences**, identique à SSD sauf que, au préalable, pour chaque fenêtre de corrélation, la moyenne des termes est d'abord calculée et soustraite à chaque terme. La différence est donc traitée entre les écarts à la moyenne des termes et non entre les termes eux-mêmes, ceci permettant d'obtenir un critère invariant aux translations uniformes des luminances dans une des images.
- **ZSAD : Zero mean Sum of Absolute Differences**, identique à SAD mais la différence est effectuée entre les écarts à la moyenne. Le critère est donc aussi invariant aux variations uniformes de luminance dans une des images.
- **NCC : Normalized Cross Correlation**, critère souvent utilisé mais aussi le plus long à calculer. C'est la somme des produits entre les termes correspondants des fenêtres, normalisée par le produit des moyennes quadratiques calculées pour chacune des fenêtres.
- **ZNCC : Zero mean Normalized Cross Correlation**, identique à NCC mais corrélation croisée entre les écarts à la moyenne. C'est aussi un critère invariant aux variations de

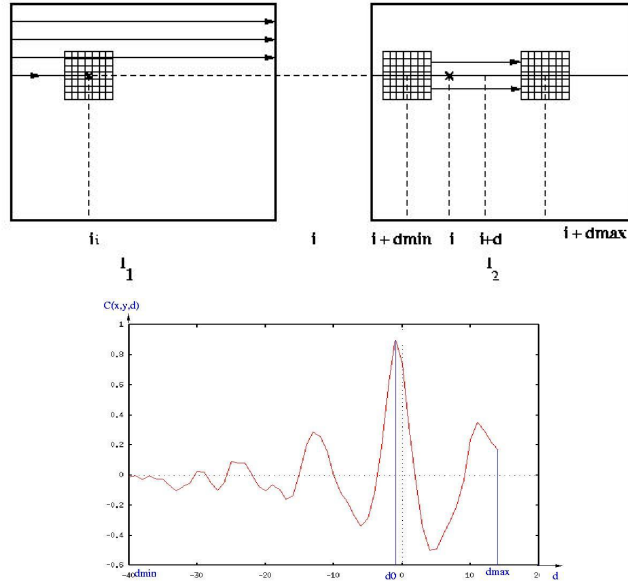


FIG. 1.9 – Recherche d'un pixel correspondant sur la ligne épipolaire d'une image rectifiée [haut] et courbe de corrélation correspondante [bas]

luminance. La formule est explicitée ci dessous :

$$C(x, y, d) = \frac{\sum_{(i,j) \in \mathcal{F}} \left[\left(I_g(x+i, y+j) - \overline{I_g(x, y)} \right) \cdot \left(I_d(x+i+d, y+j) - \overline{I_d(x+d, y)} \right) \right]}{\sqrt{\sum_{(i,j) \in \mathcal{F}} \left[I_g(x+i, y+j) - \overline{I_g(x, y)} \right]^2} \cdot \sqrt{\sum_{(i,j) \in \mathcal{F}} \left[I_d(x+i+d, y+j) - \overline{I_d(x+d, y)} \right]^2}}$$

avec :

- $\mathcal{F}$ fenêtre de corrélation,
- $I_g(x, y)$ et $I_d(x, y)$ les intensités des images gauche et droite,
- d le décalage horizontal sur x du pixel dans l'image droite appelé **disparité**. Les deux images étant rectifiées, il n'y a pas de décalage vertical.

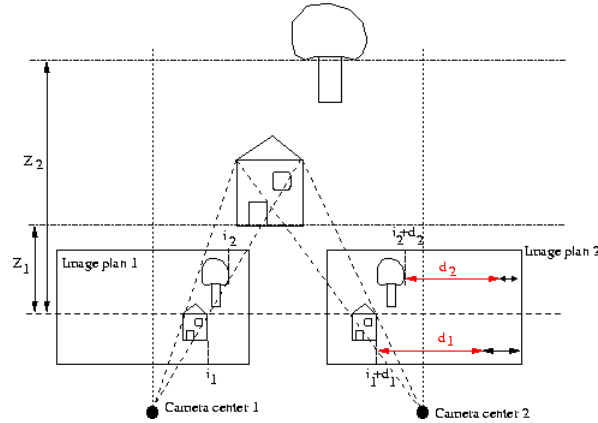


FIG. 1.10 – Illustration de l'influence de la *profondeur* sur la disparité

- $C(x, y, d)$ valeur du critère au pixel (x, y) pour une disparité d .

Ces critères numériques, en particulier ZNCC, sont longs à calculer. Les critères SAD ou ZSAD sont les plus simples, mais généralement, ils s'avèrent peu robustes. Pour des applications temps réel, [Zabih et al., 1994] ont proposé un critère non numérique, plus qualitatif, appelé **CENSUS**. Ce critère est plus robuste au bruit et applicable à des milieux peu texturés.

CENSUS

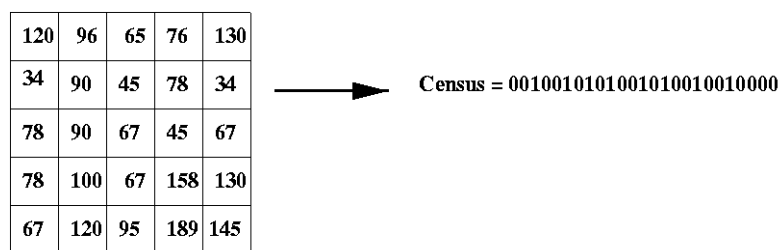


FIG. 1.11 – Transformation CENSUS sur un pixel, avec une fenêtre de corrélation de 5×5 .

Au préalable, une transformation est appliquée sur les images gauche et droite à comparer afin de remplacer la luminance en chaque pixel par une chaîne de bits, dépendante de la comparaison de la valeur de ce pixel avec les autres pixels de la fenêtre de corrélation. Par exemple, pour la fenêtre présentée en figure 1.11, la valeur 67 de luminance pour le pixel central est remplacée par la chaîne montrée à droite de la figure. Les bits de cette chaîne donnent les résultats des comparaisons de cette valeur 67 avec les valeurs de luminances des pixels de la fenêtre, parcourue ligne à ligne, de gauche à droite et de haut en bas. Le bit vaut 1 si 67 est plus grand ou égal à la valeur du pixel traité, 0 sinon. Par exemple,

- le premier bit est 0 car $120 > 67$,
- le second 0 car $96 > 67$,
- le troisième 1 car $65 < 67$, ...

Le résultat de cette transformation s'appelle **image census** issue de la **transformée census** des images originales.

Une fois que les transformations census sont appliquées sur les deux images, le score utilisé pour comparer la chaîne associée à un pixel m_1 de l'image gauche avec celles associées aux pixels de la même ligne de l'image droite est la distance de *Hamming* entre chaînes de bits, c'est à dire le nombre de bits identiques entre deux chaînes, ou le nombre de bits égal à 1 dans le *OU exclusif* entre deux chaînes.

Choix du critère de corrélation

En pratique, les critères ZNCC et *Census* donnent de bons résultats. [Gautama et al., 1999] montrent que le score *Census* peut se révéler meilleur que le score ZNCC pour certaines applications. Dans cette thèse, les deux scores sont utilisés en fonction des performances observées.

1.4.2 L'implémentation multi-résolution et ses avantages

La méthode décrite dans la section précédente peut être accélérée, en exploitant un schéma de traitement multi-résolution. Le but est de réduire l'intervalle de disparité $[d_{min}..d_{max}]$ en effectuant d'abord la recherche du correspondant à un niveau de résolution très faible (typiquement, sur une image 80×60), puis à propager les disparités pour les résolutions plus élevées. Si l'intervalle de disparité est de 80 pixels :

- l'algorithme séquentiel décrit dans la section précédente calculera 80 scores de corrélation à la résolution 640×480 , puis cherchera le maximum,
- l'algorithme exploitant une pyramide multi-résolution à 4 étages calculera 19 scores : 10 scores à la résolution 80×60 pour prédire une disparité et 3 scores aux résolutions 160×120 , 320×240 puis 640×480 .

Architecture générale

La multi-résolution s'appuie sur l'algorithme séquentiel de la corrélation *simple résolution* décrit en section précédente. Le principe peut se décomposer comme suit :

1. réduction de la paire originale avec différents facteurs (cf. figure 1.12),
2. corrélation de la paire d'images à la plus *faible* résolution,
3. corrélation successive en *remontant* la pyramide jusqu'au niveau de résolution original,
4. éventuellement, recherche d'une résolution sous-pixelique au niveau de résolution le plus élevé.

Il est possible d'effectuer les réductions avec différents algorithmes : réduction par échantillonnage en prenant 1 pixel sur n (n étant le facteur de réduction) ou réduction par moyennage : pour chaque pixel de l'image réduite, on calcule la moyenne des niveaux de gris sur les $n \times n$ pixels correspondants dans l'image initiale.

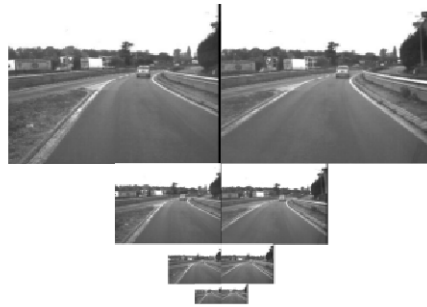


FIG. 1.12 – Pyramide de résolution sur une paire d'images

Description d'un niveau de la pyramide de résolution

La création de la pyramide de résolution est récursive et suit le schéma suivant :

1. Création du niveau : au lancement du premier niveau de la pyramide de résolution.
2. Création de la paire d'images à la résolution $\frac{A}{n^{N-1}} \times \frac{B}{n^{N-1}}$ avec
 - A : nombre de colonnes de l'image initiale

- B : nombre de lignes de l'image initiale
- n : facteur de réduction
- N : Numéro du niveau

3. Création du niveau suivant si nécessaire.

Corrélation avec estimée du niveau suivant

La corrélation s'effectue sur le niveau le plus petit de résolution avec l'algorithme de corrélation standard. Cette première étape nous donne une première carte de disparité sur laquelle on s'appuie pour la corrélation de niveau $N - 1$. Pour un pixel p donné au niveau N , on recherche la valeur de disparité du pixel correspondant dans l'image réduite de niveau $N + 1$. Cette valeur permet de rechercher le pixel homologue dans une zone centrée sur le pixel distant de $n \times v$ pixels de p . Cette étape évite de faire une recherche du pixel homologue sur la ligne entière. Au fur et à mesure que l'on remonte dans les niveaux de résolution, on améliore la précision des résultats.

Avantages de la multi-résolution

Plusieurs avantages justifient l'utilisation de la stéréocorrélation multi-résolution. Le fait de limiter l'intervalle de recherche optimise les performances globales du système tout en améliorant la carte de disparité en limitant le nombre de faux appariements. De plus, cela rend aussi plus robuste la corrélation vis-à-vis de la décalibration. En effet, réduire de manière importante l'image en échantillonnant tend à éliminer les problèmes de rectification qui pourraient apparaître : lorsqu'un point est à deux pixels de sa droite épipolaire, réduire l'image avec un facteur 2 réduit cette distance à un seul pixel. Nous verrons en chapitre 2 que cette propriété est importante pour obtenir des estimées des points 3D même quand le capteur est décalibré ou en cours de décalibration.

1.4.3 Modèle d'erreur

[Matthies, 1992] montre que les variations sur l'estimation de la disparité se répartissent selon une gaussienne dont l'écart-type croît lorsque la disparité diminue. Un autre aspect est la corrélation entre la forme du pic dans la courbe de corrélation et l'écart-type sur la disparité. En effet, ce dernier tend à augmenter lorsque le pic s'élargit. A l'aide de la méthode d'apprentissage statistique développée dans [Matthies, 1992], il est possible d'exprimer σ_d en fonction de δ (moyenne des différences de score entre le maximum et les deux disparités voisines) avec l'expression de la forme :

$$\sigma_d = \frac{n_1}{(\delta + n_2)} \quad (1.49)$$

avec n_1 et n_2 calculés expérimentalement.

Rappelons les équations de triangulation d'un point 3D $\mathbf{M}_c = (x_c, y_c, z_c)^t$ à partir de ces deux projections $\mathbf{m}_1 = (u_1, v_1)^t$ et $\mathbf{m}_2 = (u_2, v_2)^t$ dans les images (dans le repère $\mathcal{R}_1$ de la caméra

gauche) :

$$\begin{aligned} x_c &= f_3(d) \\ &= \frac{B\alpha_v(u_1 - u_0)}{d\alpha_u} \end{aligned} \quad (1.50)$$

$$\begin{aligned} y_c &= f_2(d) \\ &= \frac{-B(v_0 - v_1)}{d} \end{aligned} \quad (1.51)$$

$$\begin{aligned} z_c &= f_1(d) \\ &= \frac{B\alpha_v}{d} \end{aligned} \quad (1.52)$$

La jacobienne du vecteur $\mathbf{M}_c$ s'écrit :

$$\begin{aligned} \sigma_{\mathbf{M}}^2 &= \begin{pmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{xz} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{yz} \\ \sigma_{xz} & \sigma_{yz} & \sigma_z^2 \end{pmatrix} \\ &= \begin{pmatrix} \frac{\partial f_3}{\partial d} \\ \frac{\partial f_2}{\partial d} \\ \frac{\partial f_1}{\partial d} \end{pmatrix} \sigma_d^2 \begin{pmatrix} \frac{\partial f_3}{\partial d} & \frac{\partial f_2}{\partial d} & \frac{\partial f_1}{\partial d} \end{pmatrix} \end{aligned} \quad (1.53)$$

Par propagation de l'incertitude sur d vers le vecteur $(x_c, y_c, z_c)^t$ en utilisant sa jacobienne $\sigma_{\mathbf{M}}^2$, les expressions des incertitudes σ_x , σ_y et σ_z deviennent :

$$\sigma_x^2 = \gamma^2 \sigma_z^2 \quad (1.54)$$

$$\sigma_y^2 = \beta^2 \sigma_x^2 \quad (1.55)$$

$$\sigma_z^2 = \frac{B^2 \alpha_v^2}{d^4} \sigma_d^2 \quad (1.56)$$

et

$$\sigma_{xy} = \beta \gamma \sigma_x^2 \quad (1.57)$$

$$\sigma_{xz} = \gamma \sigma_x^2 \quad (1.58)$$

$$\sigma_{yz} = \beta \sigma_x^2 \quad (1.59)$$

avec $\beta = \frac{v_1 - v_0}{\alpha_v}$ et $\gamma = \frac{u_1 - u_0}{\alpha_u}$.

Ces expressions sont valables sous l'hypothèse qu'il n'y a pas d'erreur sur les coordonnées pixels (u_i, v_i) . Dans le cas contraire, il faut rajouter des coefficients supplémentaires sur les variances en y et z :

$$\sigma_x^2 = \gamma^2 \sigma_z^2 + \sigma_u^2 \frac{B^2 \alpha_v^2}{d^4 \sigma_u^2} \quad (1.60)$$

$$\sigma_y^2 = \beta^2 \sigma_z^2 + \sigma_v^2 \frac{B^2}{d^4} \quad (1.61)$$

Bibliographie du chapitre

- [Aubert et al., 2003] D. Aubert, L. Vabre, R. Chapuis, F. Nashashibi, M. Devy, G. Yahiaoui, J. Lelevé, S. Mousset, M. Lescure, et J. Lonnoy (2003). Spécification capteurs de la fonction prévenir les collisions. Rapport technique, ARCOS.
- [Blaszka, 1997] T. Blaszka (1997). *Approches par modèles en vision précoce*. Thèse de doctorat, Université de Nice Sophia-Antipolis, France.
- [Brand, 1995] P. Brand (1995). *Reconstruction tridimensionnelle à partir d'une caméra en mouvement : de l'influence de la précision*. Thèse de doctorat, Université Claude Bernard, Lyon I, France.
- [Brock et al., 1999] O. Brock et O. Khatib (1999). High-speed navigation using the global dynamic window approach. Dans *IEEE International Conference on Robotics and Automation (ICRA '99)*, pages 341–346, Detroit, Michigan.
- [Cernatès, 2005] J. A. Cernatès (2005). *Navigation visuelle d'un robot mobile dans un environnement d'extérieur semi-structuré*. Thèse de doctorat, Institut National Polytechnique de Toulouse.
- [Cornille, 2005] N. Cornille (2005). *Accurate 3D shape and displacement measurement using a scanning electron microscope*. Thèse de doctorat, University of South Carolina and Institut National des Sciences Appliquées de Toulouse.
- [Cornille et al., 2005] N. Cornille, D. Garcia, M. Sutton, S. McNeill, et J.-J. Orteu (2005). Calibrage d'imageurs avec prise en compte des distorsions. *Instrumentation, Mesure, Métrologie (I2M)*, 4.
- [Faugeras, 1993] O. Faugeras (1993). *Three-dimensional computer-vision : a geometric viewpoint*. MIT Press.
- [Faugeras et al., 1987] O. Faugeras et G. Toscani (1987). Camera calibration for 3d computer vision. Dans *International workshop on Machine Vision and Machine Intelligence*, pages 240–247, Tokyo (Japan).
- [Garcia, 2001] D. Garcia (2001). *Mesure de formes et de champs de déplacements tridimensionnels par stéréo-corrélation d'images*. Thèse de doctorat, INPT.
- [Gautama et al., 1999] S. Gautama, S. Lacroix, et M. Devy (1999). On the performance of stereo matching algorithms. Dans *4th Workshop on Vision, Modeling, and Visualization (VMV'99), Erlangen (Allemagne)*.
- [Hartley et al., 1997] R. Hartley et P. Sturm (1997). Triangulation. Dans *Computer Vision and Image Understanding*.
- [Horaud et al., 1995] R. Horaud et O. Monga (1995). *Vision par ordinateur : outils fondamentaux*.

- [Kulsrud, 1961] H. Kulsrud (1961). A practical technique for the determination of the optimum relaxation factor of the successive over-relaxation method. Dans *Communications of the ACM*, volume 4, pages 184–187.
- [Lacroix, 1995] S. Lacroix (1995). *Stratégies de perception et de déplacement pour la navigation d'un robot mobile autonome en environnement naturel*. Thèse de doctorat, Université Paul Sabatier, Toulouse III.
- [Lamiriaux et al., 2004] F. Lamiriaux, D. Bonnafoos, et O. Lefebvre (2004). Reactive path deformation for nonholonomic mobile robots. *IEEE Transactions on Robotics*, 20(6) :967–977.
- [Lasserre et al., 1995] P. Lasserre et P. Grandjean (1995). Stereo vision improvements. Dans *IEEE International Conference on Advanced Robotics*, Barcelone.
- [Laumond, 2001] J. Laumond (2001). *La robotique mobile*.
- [Lavest et al., 1999] J. Lavest, M. Viala, et M. Dhome (1999). Quelle précision pour une mire d'étalonnage. *Traitement du Signal*, 16.
- [Levenberg, 1944] K. Levenberg (1944). A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2 :164–168.
- [Lorusso et al., 1997] A. Lorusso, D. Eggert, et R. Fisher (1997). A comparison of four algorithms for estimating 3d rigid transformation. *Machine vision and applications*, pages 272–290.
- [Matthies, 1992] L. Matthies (1992). Stereo vision for planetary rovers : Stochastic modeling to near real-time implementation. *IJCV*, 8(1) :71–91.
- [Minguez et al., 2004] J. Minguez et L. Montano (2004). Nearness diagram (nd) navigation : collision avoidance in troublesome scenarios. *IEEE-TRA*, 20(1) :45–59.
- [Moravec et al., 1985] H. Moravec et A. Elfes (1985). High Resolution Maps from Wide Angle Sonar. Dans *Proc. IEEE International Conference on Robotics and Automation, St Louis (USA)*.
- [Nielsen, 1998] H. Nielsen (1998). Cubic splines. Lecture notes, Department of Mathematical Modeling, Technical University of Denmark.
- [Ostrowski, 1953] A. Ostrowski (1953). On over and under relaxation in the theory of the cyclic single step iteration. Dans *Mathematical Tables and Other Aids to Computation*, volume 7, pages 152–159.
- [Perez, 1994] J. Perez (1994). *Optique géométrique et ondulatoire*.
- [Peuchot, 1994] B. Peuchot (1994). Utilisation de détecteurs subpixels dans la modélisation d'une caméra. Dans *Actes du 9ème congrès AFCET (RFIA '94)*, pages 691–695, Paris.
- [Quinlan et al., 1993] S. Quinlan et O. Khatib (1993). Elastic bands : Connecting path planning and control. Dans *IEEE International Conference on Robotics and Automation (ICRA '93)*, pages 802–807.
- [Ravn et al., 1993] O. Ravn, N. Andersen, et A. Sorensen (1993). Auto-calibration in automation systems using vision. Dans *3rd International Symposium on Experimental Robotics (ISER '93)*, pages 206–218, Japon.
- [Smith et al., 1986] R. C. Smith et P. Cheeseman (1986). On the Representation of Spatial Uncertainty. *Int. Journal on Robotics Research*, 5(4).
- [Sola et al., 2005] J. Sola, A. Monin, M. Devy, et T. Lemaire (2005). Undelayed initialization in bearing only slam.

-
- [Toscani, 1987] G. Toscani (1987). *Système de calibration optique et perception du mouvement en vision artificielle*. Thèse de doctorat, Université de Paris-Sud, Centre d'Orsay (France).
- [Tournié, 2002] E. Tournié (2002). Compte rendu succinct. Rapport technique, ARCOS.
- [Tsai, 1987] R. Tsai (1987). A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation*.
- [Weng et al., 1992] J. Weng, P. Cohen, et M. Herniou (1992). camera calibration with distortion models and accuracy evaluation. Dans *IEEE Transaction on Pattern Analysis and Machine Intelligence (PAMI'92)*, pages 965–980.
- [Zabih et al., 1994] R. Zabih et J. Woodfill (1994). Non-parametric local transforms for computing visual correspondence. Dans *European Conference on Computer Vision (ECCV'94)*, pages 151–158, Stockholm, Suède.
- [Zeller, 1996] C. Zeller (1996). *Calibration projective, affine et euclidienne en vision par ordinateur et application à la perception tridimensionnelle*. Thèse de doctorat, Ecole Polytechnique.
- [Zhang, 1998] Z. Zhang (1998). Understanding the relationship between the optimization criteria in two-view motion analysis. Dans *IEEE International Conference on Computer Vision (ICCV'98)*, pages 231–236, Bombay.

Chapitre 2

Recalibration d'un capteur stéréoscopique

Sommaire

2.1	Etat de l'art	40
2.1.1	Estimation de la matrice fondamentale	40
	Algorithme de Longuet-Higgins	42
	Méthode non linéaire	42
2.1.2	Ajustement de faisceaux	43
2.1.3	Conclusion	44
2.2	Evaluation de la qualité de calibration	45
2.3	Principe de la réestimation	48
2.3.1	Algorithme	49
2.3.2	Description du modèle à réestimer	50
2.4	Autocalibration du modèle d'erreur	53
2.5	Application à l'odométrie optique	53
2.6	Résultats	55
2.6.1	calibration ponctuel	56
2.6.2	calibration avec redondance temporelle	58
2.7	Conclusion et discussions	60

Les motivations de ce travail se fondent sur l'impossibilité d'employer des mires pour la calibration périodique d'un système stéréoscopique sur un véhicule grand public. Les vibrations, chocs et déformations de la carrosserie supportés par un véhicule induisent des déplacements, même faibles, des deux caméras l'une par rapport à l'autre, déplacements très préjudiciables pour la qualité de la reconstruction 3D. S'il est tout à fait imaginable de réaliser une première calibration exacte en usine, il l'est beaucoup moins d'imposer des visites régulières de calibration chez le garagiste. Nous devons donc avoir la possibilité de recalibrer notre système sans l'utilisation d'objet de calibration de manière à rendre le système totalement indépendant du conducteur. Cette démarche se rapproche de l'autocalibration, très étudié par la communauté vision. Les hypothèses propres à notre application nous permettent de partir sur une calibration exact (cf. section 1.3.2) qui servira d'initialisation à la recalibration en partant du principe que le système n'est pas complètement décalibré.

En section 2.1 de ce chapitre, nous verrons une description des méthodes existantes d'autocalibration. L'évaluation de la calibration courant permettant de ne pas perdre l'hypothèse de la recalibration (et non de l'autocalibration) sera présentée en section 2.2. Nous verrons alors en section 2.3 la description du modèle à réestimer. La mise à jour du modèle d'erreur est présentée en section 2.4. Les méthodes utilisées pour la calibration étant très proche d'un système d'odométrie optique, nous en verrons la description en section 2.5. Enfin, la section 2.6 nous donnera des résultats sur des données de synthèse afin d'évaluer les performances du système.

2.1 Etat de l'art

Comme nous l'avons vu en section 1, la calibration d'un système stéréoscopique est, depuis longtemps, le sujet de nombreuses études. Les méthodes à base de mires, objets de géométrie connue, sont maintenant bien maîtrisées. L'utilisation de ces mires, quand elles sont nécessaires et de plus ou moins grande taille selon le banc utilisé, rend ces manipulations laborieuses. De plus, certaines applications telles que la stéréovision sur véhicules ne permettent pas leur mise en pratique. Nous verrons dans cette section un rapide état de l'art des méthodes d'autocalibration, méthodes qui ne nécessitent aucune connaissance euclidienne de l'espace observé afin de s'affranchir de tout objet de calibration.

En première partie de cette section, nous verrons les méthodes basées sur l'estimation de la matrice fondamentale dont l'importance pour un système stéréoscopique a été présentée en section 1.3. Ensuite, nous verrons qu'il est tout à fait possible d'appréhender un banc de stéréovision comme le mouvement d'une caméra et non la présence physique de deux caméras. A partir de cette approche, les techniques photogrammétriques comme l'ajustement de faisceaux peuvent être utilisées. Cela revient à estimer la transformation inter-caméra en même temps que la réestimation des points 3D du monde utilisés comme points de calibration. Nous verrons dans nos travaux qu'il est également possible de réestimer les paramètres intrinsèques de chacune des caméras.

2.1.1 Estimation de la matrice fondamentale

Comme nous l'avons vu en section 1.3, la matrice fondamentale contient l'information sur la géométrie épipolaire ainsi que les paramètres intrinsèques des deux caméras. Estimer cette matrice revient donc à identifier l'ensemble des paramètres nécessaires à la connaissance complète de la géométrie du capteur à un facteur d'échelle près.

Certaines méthodes d'évaluation de la matrice fondamentale considèrent la connaissance des intrinsèques ([Longuet-Higgins, 1981]) et utilisent la méthode linéaire des huit points appariés dans deux vues (méthode décrite plus bas). Les méthodes, sans hypothèses particulières sur les intrinsèques, utilisent le plus souvent les outils standards de la géométrie projective en appliquant des contraintes géométriques sur des primitives de l'image. La mire n'est plus utilisée au profit de recherches de correspondances de primitives entre les images. Ces correspondances peuvent être calculées automatiquement ou manuellement avec l'intervention d'un opérateur. Dans leur papier, [Liu et al., 1990], retrouvent les positions des caméras à partir de six points ou huit lignes mis en correspondance dans son algorithme linéaire ou de trois points ou trois lignes dans la version non-linéaire. Dans [Faugeras et al., 1995] et [Lourakis et al., 1999], les auteurs démontrent la possibilité de reconstruire une scène 3D à un facteur d'échelle près. La méthode décrite dans [Luong et al., 1993] calibre automatiquement deux ou trois caméras. Les paramètres intrinsèques sont estimés à l'aide de la détermination de la matrice fondamentale ; trois déplacements permettant de trouver une solution aux équations de Kruppa qui conduisent à l'évaluation des éléments de la matrice fondamentale. Les paramètres extrinsèques en sont alors déduits.

Les principaux inconvénients de ces techniques résident dans la trop forte sensibilité à la qualité des appariements des primitives. Les résultats de [Lourakis et al., 1999] montrent, par exemple, une erreur de vingt pixels sur l'estimation du point principal pour un bruit d'appariement de un pixel. L'utilisation de ces méthodes est donc rendue difficile et implique souvent l'intervention d'un tiers superviseur lors de la phase d'appariement. La calibration est alors dite *semi-automatique*.

Comme nous l'avons vu en section 1.3, l'équation principale liant la matrice fondamentale aux projections de points dans les images est la suivante :

$$\mathbf{m}_2^t \mathbf{F} \mathbf{m}_1 = 0 \quad (2.1)$$

La plupart des méthodes d'évaluation de la matrice fondamentale présentées dans la littérature comme [Luong et al., 1995] et [Boufama et al., 1995], se fondent sur cette dernière équation. D'après celle-ci, il est possible d'écrire les coefficients de $\mathbf{F}$ sous forme linéaire :

$$\mathbf{u}_i^t f = 0 \quad (2.2)$$

où $\mathbf{u}_i = (u_i u'_i, u_i v_i, u_i, v_i u'_i, v_i v'_i, v_i, u'_i, v'_i, 1)$ et

$$f = \begin{pmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{pmatrix}$$

Cette équation, vraie pour une paire de points appariés se généralise pour n-paires :

$$\mathbf{U}_n^t f = 0 \quad (2.3)$$

où $\mathbf{U}_n = [u_i, \dots, u_n]$

L'estimation de la matrice fondamentale $\mathbf{F} = \mathbf{A}_2^{-t} \mathbf{E} \mathbf{A}_1^{-1}$ permet d'obtenir l'information sur la géométrie épipolaire liée aux paramètres intrinsèques des deux caméras. Dans le cas d'une calibration de la transformation inter-caméra avec une connaissance préalable des intrinsèques, la matrice essentielle $\mathbf{E}$ se déduit simplement de l'expression de $\mathbf{F}$. Par contre, lorsque la calibration des caméras est inconnu il est nécessaire d'identifier les $\mathbf{A}_i$ pour retrouver $\mathbf{E}$. [Hartley, 1993] propose une méthode fondée sur l'algorithme de *Levenberg-Marquardt* pour retrouver la géométrie projective à un facteur d'échelle près. Un an plus tard, [Hartley, 1994] calcule les paramètres

intrinsèques d'une caméra fixe à l'aide de plusieurs images avec rotation de la caméra. L'auteur explique qu'il est possible d'estimer les intrinsèques avec seulement deux images à l'aide de contraintes telles que (1) l'orthogonalité du plan image par rapport à l'axe optique et (2) des pixels carrés ($\alpha_u = \alpha_v$). [Ma, 1996] utilise une méthode d'estimation des intrinsèques à l'aide de six translations pures et d'une optimisation linéaire. [Mohr et al., 1996] et [Hartley, 1997] introduisent alors le SVD¹⁵, technique de décomposition en valeurs singulières de la matrice fondamentale dans le but de simplifier les équations de *Kruppa*. [Lourakis et al., 1999] exploitent cette voie à l'aide de la contrainte de *Huang-Faugeras and Trivedi* pour trouver une solution plus robuste au bruit. Ils poursuivent dans [Lourakis et al., 2000] en proposant une technique d'estimation des paramètres intrinsèques lorsque ceux-ci sont variables.

Nous allons maintenant voir deux techniques d'évaluation de la matrice fondamentale basées sur l'équation 2.3.

Algorithme de Longuet-Higgins

Dans le cas de n paires de points mis en correspondance, la matrice $\mathbf{U}_n$ de l'équation 2.3 correspond à une matrice de taille $n \times 9$. Pour éviter la solution triviale $f = 0$ on ajoute la contrainte $F_{33} = 1$, contrainte autorisée grâce au rang égal à deux de la matrice. On peut aussi contraindre $\|f\| = 1$. De cette manière, 2.3 peut être résolue avec huit correspondances de primitives, d'où le nom donné par [Longuet-Higgins, 1981] *d'algorithme des huit points*. La résolution correspond à la minimisation sous contrainte suivante :

$$\begin{cases} \min_f \|\mathbf{U}_n f\|^2 \\ \|f\| = 1 \end{cases} \quad (2.4)$$

La solution de cette minimisation est donnée par le vecteur propre de la plus petite valeur propre de la matrice $\mathbf{U}_n^t \mathbf{U}_n$.

L'utilisation de cette méthode a comme avantages la rapidité d'estimation de la matrice fondamentale ainsi que le faible coût algorithmique (utilisation de seulement huit points) permettant l'utilisation d'approches robustes du type *RANSAC* afin d'éliminer les faux appariements dans les primitives 2D. Le principal défaut réside dans l'instabilité et la forte sensibilité au bruit rendant leurs utilisations difficiles. De plus, l'ajout de l'évaluation des distorsions liées aux caméras en brise la linéarité.

Méthode non linéaire

Il est possible d'évaluer la matrice fondamentale $\mathbf{F}$ par une méthode d'estimation non linéaire. La minimisation porte alors sur la somme des carrés des distances d'un point à la droite épipolaire lui correspondant. 2.1 signifie que $\mathbf{m}_2$ (point homologue de $\mathbf{m}_1 = (u, v, 1)^t$) se trouve sur la droite épipolaire l'_m de paramètres :

$$\begin{pmatrix} l'_1 \\ l'_2 \\ l'_3 \end{pmatrix} = \begin{pmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{pmatrix} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \quad (2.5)$$

¹⁵Singular Value Decomposition

La minimisation devra donc porter sur la distance suivante :

$$d(\mathbf{m}_2, l'_m) = \frac{|\mathbf{m}_2^t l'_m|}{\sqrt{l_1'^2 + l_2'^2}} \quad (2.6)$$

[Luong et al., 1995] considèrent simultanément la distance de $\mathbf{m}_2$ à la droite $\mathbf{F}\mathbf{m}_1$ ainsi que la distance de $\mathbf{m}_1$ à la droite $\mathbf{F}^t\mathbf{m}_2$ de manière à réduire l'écart entre les deux géométries épipolaires. De manière générale la minimisation pour n appariements s'écrit donc :

$$\sum_i (d^2(\mathbf{m}_{2i}, \mathbf{F}\mathbf{m}_{1i}) + d^2(\mathbf{m}_{1i}, \mathbf{F}^t\mathbf{m}_{2i})) \quad (2.7)$$

En utilisant 2.6

$$\min_{\mathbf{F}} \left[\sum_i \frac{(\mathbf{m}_{2i}^t \mathbf{F} \mathbf{m}_{1i})^2}{(\mathbf{F} \mathbf{m}_{1i})_1^2 + (\mathbf{F} \mathbf{m}_{1i})_2^2} + \sum_i \frac{(\mathbf{m}_{1i}^t \mathbf{F}^t \mathbf{m}_{2i})^2}{(\mathbf{F}^t \mathbf{m}_{2i})_1^2 + (\mathbf{F}^t \mathbf{m}_{2i})_2^2} \right] \quad (2.8)$$

Les techniques habituelles d'optimisation non linéaire telles que *Gauss-Newton* et *Levenberg-Marquardt* ([Levenberg, 1944]) peuvent alors être appliquées.

2.1.2 Ajustement de faisceaux

L'ajustement de faisceaux, aussi connu sous le nom de *Bundle Adjustment* en anglais, est une méthode fréquemment utilisée ([Pollefeys et al., 1998], [Fua, 2000], [Shum et al., 1999] ...). Le détail de la méthode est présenté en annexe A. Comme nous l'avons vu avec les méthodes précédentes d'estimation de la matrice fondamentale, les problèmes de robustesse aux mesures ont nécessité le développement de nouvelles techniques permettant de s'affranchir des erreurs sur les données d'entrées. C'est dans ce contexte que les travaux sur l'ajustement de faisceaux ont vu le jour avec des résultats de reconstruction très nettement supérieurs. Le principe est d'estimer les paramètres des caméras en même temps que les points 3D observés en minimisant la distance entre la projection de ces points et les points 2D identifiés dans l'image comme étant leurs projections. Cette minimisation non-linéaire permettra de travailler sur des données éparses avec la possibilité d'intégrer un ensemble de contraintes. Les principales difficultés sont la nécessité d'initialiser le système avec des estimées pertinentes ainsi qu'une dimension du problème qui peut rapidement exploser en nombre de paramètres. La minimisation s'écrit :

$$\min_{\mathbf{a}_j, \mathbf{b}_i} \sum_{i=1}^n \sum_{j=1}^m d(Q(\mathbf{a}_j, \mathbf{b}_i), x_{ij})^2 \quad (2.9)$$

avec :

- $\mathbf{a}_j$: vecteur de paramètres (intrinsèques et extrinsèques) de la caméra j (m caméras ou m positions d'une caméra),
- $\mathbf{b}_i$: vecteur du point 3D i (n points 3D) dans un repère de référence,
- $\mathbf{x}_{ij}$: projection du point 3D i dans l'image de la caméra j (point extrait dans l'image),
- $Q(\mathbf{a}_j, \mathbf{b}_i)$: prédiction de la projection du point 3D i dans la vue de la caméra j ,
- $d(x, y)$: distance euclidienne entre les deux points 2D x et y .

Cette définition montre l'indépendance de l'algorithme vis-à-vis des points manquants dans certaines vues. Notons, k et 3 les dimensions respectives de $\mathbf{a}_i$ et $\mathbf{b}_j$. Le nombre de paramètres à estimer est donc de $mk + 3n$ qui devient très élevé même pour des valeurs de m et n faibles.

L'ajustement de faisceaux s'écrit comme un problème d'estimation non linéaire de la manière suivante : $\mathbf{P} \in \mathbb{R}^M$ vecteur de paramètres composé de

$$\mathbf{P} = (\mathbf{a}_1^t, \dots, \mathbf{a}_m^t, \mathbf{b}_1^t, \dots, \mathbf{b}_n^t)^t$$

Le vecteur $\mathbf{X} \in \mathbb{R}^N$ est composé des mesures des n points dans les m images en supposant les appariements complets :

$$\mathbf{X} = (\underbrace{\mathbf{x}_{11}^t, \dots, \mathbf{x}_{1m}^t}_{\text{point 1}}, \mathbf{x}_{21}^t, \dots, \mathbf{x}_{2m}^t, \dots, \underbrace{\mathbf{x}_{n1}^t, \dots, \mathbf{x}_{nm}^t}_{\text{point } n})^t$$

Notons $\mathbf{P}_0$ le vecteur d'estimées initiales et $\sum_{\mathbf{X}}$ la matrice de covariance associée au vecteur de mesure $\mathbf{X}$. Pour chaque vecteur de paramètres, une prédiction du vecteur de mesures $\hat{\mathbf{X}}$ est générée : $\hat{\mathbf{X}} = f(\mathbf{P})$ défini par $\hat{\mathbf{X}} = (\hat{\mathbf{x}}_{11}^t, \dots, \hat{\mathbf{x}}_{1m}^t, \dots, \hat{\mathbf{x}}_{n1}^t, \dots, \hat{\mathbf{x}}_{nm}^t)^t$ avec $\hat{\mathbf{x}}_{ij}^t = Q(\mathbf{a}_j, \mathbf{b}_i)$. La minimisation porte sur la distance de Mahalanobis $\epsilon \sum_{\mathbf{X}}^{-1} \epsilon$, $\epsilon = \mathbf{X} - \hat{\mathbf{X}}$ sur $\mathbf{P}$. Cette minimisation est généralement effectuée à l'aide de l'algorithme de *Levenberg-Marquardt* en résolvant l'équation suivante :

$$\mathbf{J}^t \sum_{\mathbf{X}}^{-1} \mathbf{J} \delta = \mathbf{J}^t \sum_{\mathbf{X}}^{-1} \epsilon \quad (2.10)$$

où $\mathbf{J}$ est la jacobienne de f et δ la mise à jour du vecteur de paramètres $\mathbf{P}$. Sans rentrer plus dans les détails de l'algorithme, décrit en annexe A, il est intéressant de regarder la structure éparse de la jacobienne. Prenons le cas d'un système à trois points et trois caméras. La jacobienne sera de la forme suivante :

$$\mathbf{J} = \frac{\partial \mathbf{X}}{\partial \mathbf{P}} = \begin{pmatrix} \mathbf{A}_{11} & 0 & 0 & \mathbf{B}_{11} & 0 & 0 \\ 0 & \mathbf{A}_{12} & 0 & \mathbf{B}_{12} & 0 & 0 \\ 0 & 0 & \mathbf{A}_{13} & \mathbf{B}_{13} & 0 & 0 \\ \mathbf{A}_{21} & 0 & 0 & 0 & \mathbf{B}_{21} & 0 \\ 0 & \mathbf{A}_{22} & 0 & 0 & \mathbf{B}_{22} & 0 \\ 0 & 0 & \mathbf{A}_{23} & 0 & \mathbf{B}_{23} & 0 \\ \mathbf{A}_{31} & 0 & 0 & 0 & 0 & \mathbf{B}_{31} \\ 0 & \mathbf{A}_{32} & 0 & 0 & 0 & \mathbf{B}_{32} \\ 0 & 0 & \mathbf{A}_{33} & 0 & 0 & \mathbf{B}_{33} \end{pmatrix} \quad (2.11)$$

La prise en compte du nombre important de zéros lors de l'implémentation informatique permet un gain considérable en espace mémoire et en temps de calcul. L'implémentation donnée par [Lourakis et al., 2004] nous en donne un bon exemple. Grâce à ces matrices creuses nous pourrions employer des méthodes robustes pour éliminer les défaillances liées aux faux appariements. Nous verrons dans les sections suivantes la nature des jacobienes dans l'application qui nous intéresse. Un état de l'art détaillé est présenté dans [Triggs et al., 2000].

2.1.3 Conclusion

Dans cette section, nous avons vu les différentes méthodes de calibration sans l'utilisation d'un objet de calibration. La difficulté de l'estimation de la matrice fondamentale réside dans la précision nécessaire des mesures pour obtenir de bons résultats. La méthode d'ajustement de faisceaux permet, en même temps que l'estimation des paramètres du banc et des caméras, de réestimer les points de calibration. Cette méthode semble donc tout à fait adaptée à notre application pour

laquelle nous n'avons pas de mesures de grande qualité à cause des variations de la géométrie du capteur. L'ajustement de faisceaux nous permet donc d'utiliser des points 3D quelconques de la scène comme points de calibration. De plus, l'estimation des paramètres intrinsèques et extrinsèques est beaucoup plus stable lorsqu'ils sont estimés simultanément [Garcia, 2001].

Par contre, l'ajustement de faisceaux nécessite une initialisation pertinente du système. Il est donc important de lancer les recalibrations lorsque la géométrie du système n'a pas trop dérivé par rapport à sa calibration en cours. La fonction d'évaluation de l'état de la calibration est décrite en section suivante.

2.2 Evaluation de la qualité de calibration

Comme nous l'avons vu en section précédente dans l'état de l'art, l'utilisation d'une méthode photogrammétrique pour la calibration du banc stéréo impose une bonne estimée des paramètres de calibration. Pour notre application, ces estimées viennent d'une part des paramètres de calibration précédents et d'autre part des points 3D issus de la stéréocorrélation. Suite à une décalibration, les cartes de disparité se dégradent ainsi que les reconstructions 3D tributaires des triangulations sur les points appariés par la stéréocorrélation, triangulations dépendantes des paramètres de calibration.

L'évaluation de la qualité de la calibration doit donc pouvoir comparer les résultats donnés par l'algorithme de stéréovision avec une vérité terrain dont le calcul doit être indépendant de la géométrie épipolaire. Nous retrouvons donc deux fonctions dans notre évaluation :

1. appariement de points d'intérêt servant de vérité terrain,
2. stéréocorrélation sur les mêmes points que l'appariement précédent.

Sur une base de points de l'image de référence, nous allons comparer le résultat de ces deux méthodes d'appariement.

La figure 2.1 va nous servir de support à l'explication de l'algorithme. La paire d'images en haut de la figure correspond à la paire originale, sans aucun post-traitement, issue du capteur stéréoscopique. La paire située juste en dessous correspond à ces deux images (gauche et droite) rectifiées et corrigées des distorsions issues de la calibration en cours. La dernière image correspond à notre carte de disparité. Les six étapes encadrées sont décrites ci-dessous :

1. La première étape correspond à la phase de détection de notre vérité terrain. Son but est la création d'un jeu de points d'intérêt appariés entre les deux images. Cet appariement ne doit pas tenir compte de la géométrie épipolaire et doit rester totalement indépendant de la calibration du banc de stéréovision. Pour cela, nous avons utilisé la méthode proposée par [Jung et al., 2001] qui respecte ces contraintes. Le principe en est un appariement de points de *Harris*. C'est-à-dire que pour chaque point de *Harris* de l'image de gauche nous connaissons ses coordonnées dans l'image droite. Ce sont ces appariements qui vont servir de références.
2. Les étapes de 2 à 5 correspondent à la création du jeu d'informations de notre phase 2 qui prend en compte les informations sur les paramètres de calibration. Pour chaque point gauche de notre jeu de référence nous calculons les coordonnées dans l'image gauche rectifiée et corrigée des distorsions afin d'utiliser les paramètres de géométrie épipolaire de l'image gauche.

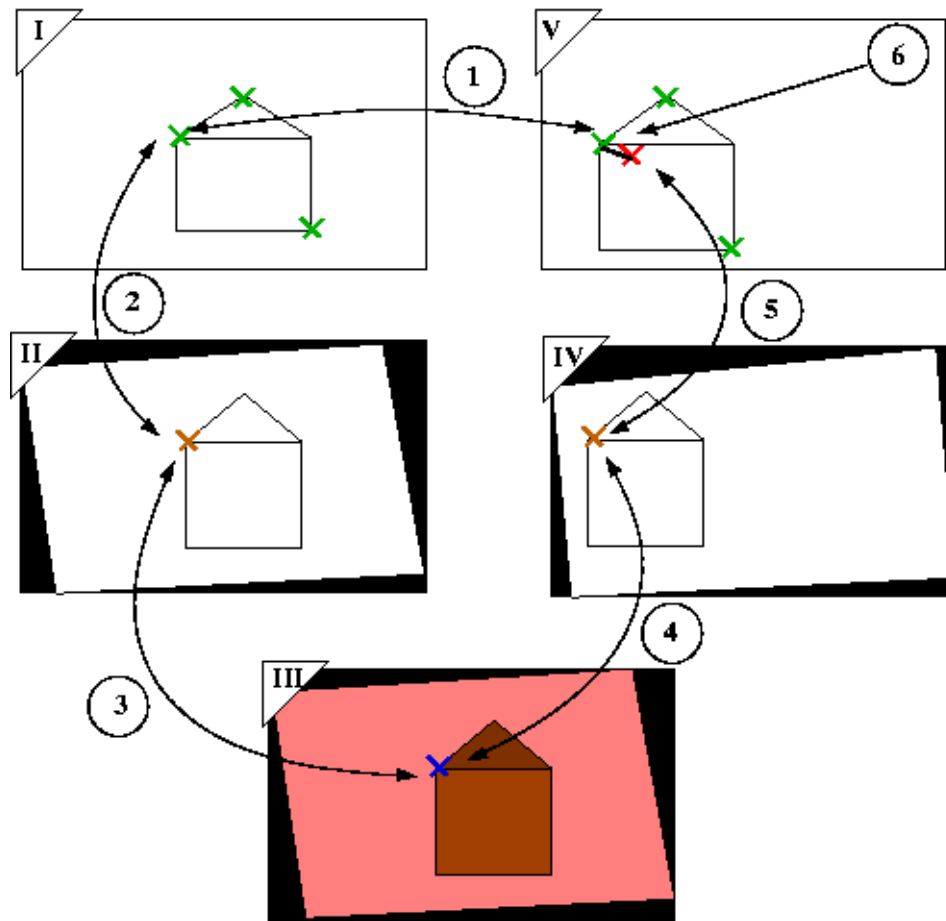


FIG. 2.1 – Evaluation de la qualité de calibration. Image I : image gauche originale ; Image II : image gauche rectifiée et corrigée des distorsions ; Image III : image de disparités issue de la stéréovision ; Image IV : Image droite rectifiée et corrigée des distorsions ; Image V : Image droite originale

3. L'ensemble de ces points est alors corrélé par les algorithmes de la section 1.4 pour nous donner une liste de stéréo-correspondants. Aucun filtre sur les résultats de disparités n'est appliqué de manière à garder l'ensemble des erreurs commises, objectif de cette phase de traitement. Dans ce cas, une erreur de rectification est facilement détectée.
4. Les valeurs de disparités sont directement en lien avec la reconstruction 3D des points avec l'utilisation de la transformation inter-caméras. L'erreur sur un point correctement corrélé à l'étape précédente malgré une décalibration s'intègre dans cette étape de reconstruction. A l'aide des paramètres de calibration en cours, il est possible de reprojecter ce point 3D dans l'image droite rectifiée et corrigée avec l'utilisation de la transformation inter-caméras pour le changement de repère caméra gauche/caméra droite et la matrice intrinsèque droite pour la projection dans l'image.
5. Toujours sur la base des paramètres de calibration en cours nous calculons les coordonnées du point trouvé dans l'image originale droite.
6. Toutes les étapes précédentes ont cumulé les informations liées aux paramètres de calibration (reconstruction, projection, géométrie épipolaire avec la rectification des images). Si

ces paramètres ne sont plus à jour, le point final à l'étape 5 dans l'image originale droite ne doit plus correspondre au point de Harris droit apparié lors de l'étape 1. Il est donc possible de calculer ces erreurs à travers la distance entre le point résultat de l'appariement de points d'intérêt et le point reprojeté issu de la stéréovision.

Le critère de qualité de calibration devient naturellement une erreur quadratique sur les distances entre points appariés par stéréovision et points d'intérêt appariés. Le critère s'écrit :

$$rms = \frac{1}{n} \sum_i d_i^2 \quad (2.12)$$

avec :

- i : numéro du point dans l'ensemble des points d'intérêt appariés
- n : nombre de couples de points d'intérêt, à la fois appariés et corrélés,
- d : distance euclidienne entre deux points 2D

La normalisation de ce score *rms* (*Root mean Square* en anglais) selon la largeur de l'image permet de comparer la qualité de calibration entre différents systèmes n'utilisant pas la même résolution d'image. L'utilisation de la reconstruction 3D est importante ici. L'espace de disparité est beaucoup moins bruité que l'espace 3D. Une bonne corrélation n'est pas la garantie d'une bonne qualité de reconstruction. Par exemple, avec une simple rotation de la caméra droite selon l'axe horizontal on peut observer une bande verticale très bien corrélée plus ou moins large selon l'angle de divergence des caméras. Cet effet est visible sur l'image 2.2. La reconstruction 3D du point avec cette valeur de disparité *faussée*, suivie de la reprojection dans la deuxième caméra permet de mettre en avant les erreurs sur ces points, erreurs non détectables en restant dans l'espace de disparité. D'éventuelles erreurs sur la phase d'appariement de points d'intérêt ne remettent pas en cause l'évaluation si leur nombre n'est pas trop important. Nous verrons leur influence dans la section 2.6 de résultats sur données de synthèse.

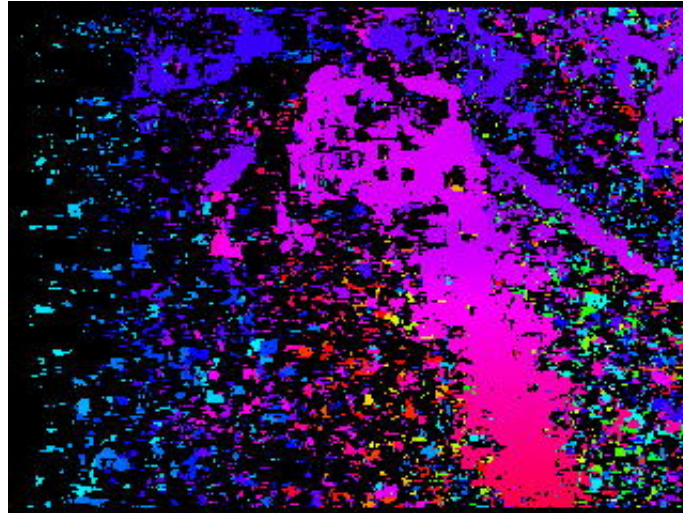


FIG. 2.2 – Effet sur la carte de disparité, d'une divergence de 4° des caméras sans variation de tilt suite à une décalibration. On remarque une bande encore nettement corrélée malgré le déficit de calibration.

Cette technique d'évaluation est tributaire de l'appariement des points d'intérêt. Il est donc impossible d'évaluer la qualité de la calibration dans des environnements uniformes. Les tech-

niques tirées de [Jung et al., 2001] sont relativement robustes malgré quelques défaillances sur des images présentant de fortes variations de contraste. Les résultats sont présentés et discutés en section 2.6.

2.3 Principe de la réestimation

Comme nous l'avons vu précédemment, nous pourrions utiliser une méthode d'estimation de la matrice fondamentale pour initialiser notre ajustement de faisceaux. Dans la section 2.2 précédente, nous avons vu la méthode d'évaluation de la calibration. Or, les méthodes utilisées et les résultats générés pour cette évaluation sont très proches de ceux dont nous avons besoin pour initialiser la réestimation des paramètres de calibration. Dans la suite de cette section nous verrons le détail de l'algorithme de recalibration puis nous poserons l'ensemble des équations utiles pour l'ajustement de faisceaux.

Pour simplifier les écritures, la description de l'algorithme qui va suivre se base sur la calibration d'une paire stéréo avec l'utilisation de deux seules images ; image gauche et image droite vue comme image 1 et image 2 d'une possible séquence d'images. Il est tout à fait possible, et simple, de généraliser cet algorithme pour travailler sur un nombre d'images plus important en suivant les points dans le temps (Cf figure 2.4). Ce suivi permet d'augmenter le nombre de droites dans le faisceau associé à un point et ainsi d'améliorer la précision des résultats au détriment du temps de calcul.

Par contre, l'utilisation de l'ajustement de faisceaux pose le problème du facteur d'échelle tout comme les méthodes d'évaluation de la matrice fondamentale. En effet, cette méthode produit une reconstruction 3D non métrique de l'environnement. L'introduction d'une métrique passe par l'ajout de contraintes sur le système :

- [Garcia, 2001] montre qu'il suffit de bloquer deux points et d'ajouter une contrainte sur la composante z d'un troisième pour déterminer le facteur d'échelle. La difficulté est de choisir trois points dont les estimées ne sont pas trop bruitées, sans quoi le facteur d'échelle ne sera pas correct. Nous verrons l'influence de ce bruit dans la section de résultats sur données de synthèse.
- l'utilisation d'une information odométrique avec un suivi temporel permet de fixer le facteur d'échelle (cf. figure 2.4) sur le même principe que [Garcia, 2001]. Le système ne revient donc plus à estimer deux jeux de paramètres en plus de la structure mais quatre jeux de paramètres. C'est-à-dire qu'en plus de fixer la position de la caméra gauche à l'instant t , nous fixons la position de cette caméra à l'instant $t+1$ (relativement à t) grâce à l'odométrie. Nous verrons l'influence du bruit d'odométrie dans la section résultats de ce chapitre. Il serait possible d'effectuer un suivi temporel sur plus de positions de caméra mais l'erreur bien connue de divergence de l'odométrie serait alors à prendre en compte.

Ces contraintes sur le processus de recalibration se doivent d'être plus précises que la décalibration attendu des caméras. Or, en dehors des cas d'accidents, la déformation de la carrosserie ne dépasse pas les quelques dixièmes de millimètre. Il est vrai que l'amélioration des techniques odométriques est importante surtout avec l'utilisation du *Mapmatching* [Bernstein et al., 1998] dont l'erreur relative est de l'ordre de 1 pour-cent, mais cela reste réservé à des véhicules haut de gamme et rend le processus de calibration dépendant d'un autre système.

D'autres possibilités sont envisageables dans notre cas spécifique. Par exemple, dans le contexte de la détection d'obstacles en milieu routier où le profil de la route est calculé, il devient alors possible de filtrer les points sur la surface de la route et d'ajouter une contrainte de coplanarité. Sinon, il est possible d'exploiter la connaissance liée à l'environnement, telles que la largeur des voies, la longueur des lignes blanches, ... pour ajouter des contraintes à la réestimation. Ces solutions sont à envisager au cas par cas mais ne peuvent être générales.

2.3.1 Algorithme

D'après la section 2.1.2 les estimées initiales nécessaires à l'ajustement de faisceaux correspondent aux vecteurs $\mathbf{X}_0 = (\mathbf{x}_{11}^t, \dots, \mathbf{x}_{n1}^t, \mathbf{x}_{12}^t, \dots, \mathbf{x}_{n2}^t)^t$ de projection des points dans les images. Ce vecteur correspond ici aux coordonnées 2D des points d'intérêt appariés. Le vecteur de paramètres $\mathbf{P}_0 = (\mathbf{a}_1^t, \mathbf{a}_2^t, \mathbf{b}_1^t, \dots, \mathbf{b}_n^t)^t$ est composé de $\mathbf{a}_j$, $1 \leq j \leq 2$, paramètres intrinsèques issus de la calibration précédente, et des coordonnées 3D $\mathbf{b}_i$, $1 \leq i \leq n$ issues de la triangulation des points d'intérêt appariés grâce à la matrice de transformation inter-caméras précédente. Notons :

- $\mathbf{R} = (\cos(\frac{\theta}{2}), u_1 \sin(\frac{\theta}{2}), u_2 \sin(\frac{\theta}{2}), u_3 \sin(\frac{\theta}{2}))$ la rotation d'angle θ autour du vecteur unitaire $\mathbf{u} = (u_1, u_2, u_3)^t$, $\|\mathbf{R}\| = 1$,
- $\mathbf{t}$ le vecteur translation,
- et k_j vecteur des paramètres de dimension k : paramètres intrinsèques α_u, α_v, u_0 et v_0 et paramètres de distorsion r_1, r_2 et r_3 (cf. section 1.2).

Les paramètres $\mathbf{a}_j$ et $\mathbf{b}_i$ de notre système deviennent donc :

$$\mathbf{a}_j = (\mathbf{R}_j, \mathbf{t}_j, \mathbf{k}_j)^t \quad (2.13)$$

$$\mathbf{b}_i = \mathbf{M}_i \quad (2.14)$$

avec i numéro du point 3D et j numéro de l'image. La fonction de projection correspond alors à :

$$Q(\mathbf{a}_j, \mathbf{b}_i) = \mathbf{K}_j(\mathbf{R}_j \mathbf{N}_i \mathbf{R}_j^{-1} + \mathbf{t}_j) + \Delta \quad (2.15)$$

avec :

- $\mathbf{K}_j$ matrice 3×3 des paramètres intrinsèques de la caméra j ,
- $\mathbf{N}_i = (0, \mathbf{M}_i^t)$ vecteur correspondant au point 3D $\mathbf{M}_i$,
- Δ vecteur d'aberration $\Delta = (r_1, r_2, r_3)^t$,
- et $\mathbf{R}_j \mathbf{N}_i \mathbf{R}_j^{-1}$ correspondant à la rotation du point 3D $\mathbf{M}_i$ d'angle θ_j autour du vecteur unitaire $\mathbf{u}_j$, rotation donnée par $\mathbf{R}_j$. $\|\mathbf{R}\| = 1$.

Le schéma des étapes de la recalibration est présenté en figure 2.3. La structure de l'algorithme est basée sur une approche robuste de RANSAC de manière à éliminer les faux appariements des points d'intérêt. Il suffit de reprendre ce jeu d'appariement calculé lors de l'évaluation de la calibration pour lancer la recalibration. Une sélection est effectuée dans ce jeu de points afin d'obtenir le nombre suffisant d'équations pour résoudre notre système. Ce nombre d'équations doit satisfaire les $3n + 2k$ inconnues (n points 3D de trois composantes (X, Y, Z) et k paramètres pour chacune des caméras). Cette nouvelle calibration est soumise aux points ne figurant pas dans l'échantillon sélectionné. Par comptage statistique des points valides en fonction du nouveau modèle, nous validons ou non cette estimation ; un point valide étant un point reprojeté à une distance $dist \leq \epsilon$ de ses coordonnées dans la liste des points d'intérêt appariés. Le pourcentage de points nécessaire à la validation du modèle est fonction de la performance de la méthode d'appariement des points d'intérêt. Dans notre cas nous avons fixé ce pourcentage à 90%. Si ce

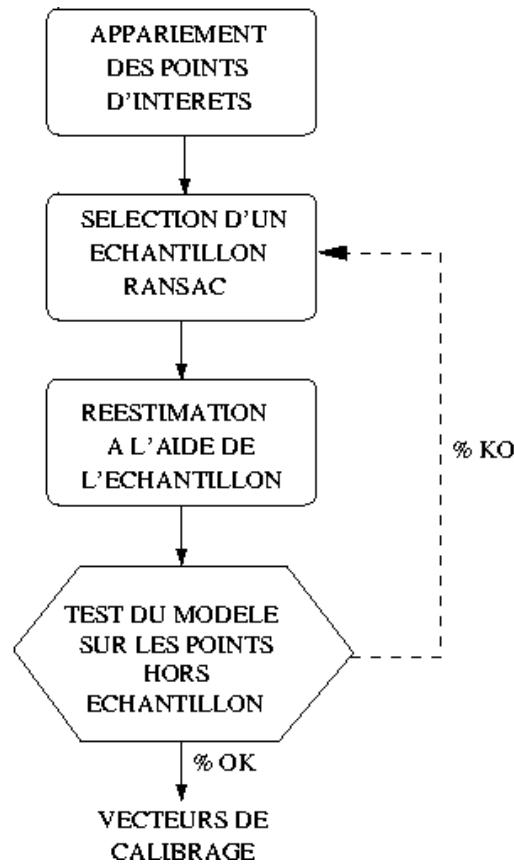


FIG. 2.3 – Algorithme de recalibration avec approche robuste RANSAC.

pourcentage n'est pas atteint, un autre échantillon est sélectionné jusqu'à ce qu'un jeu correct soit trouvé. Le nombre d'itérations maximum est fixé pour éviter de travailler sur des images obsolètes. La section 2.6 donne tous les détails sur la robustesse aux faux appariements, aux bruits sur les points 3D ainsi que sur la mesure des points d'intérêt et nous donnera des idées des temps d'exécution. A titre d'exemple, l'utilisation d'un algorithme robuste et l'exploitation des matrices creuses de l'ajustement de faisceaux nous permettent d'avoir des temps d'exécution de l'ordre de 20ms pour une itération.

2.3.2 Description du modèle à réestimer

Nous avons vu que le modèle complet d'une caméra, basé sur le modèle sténopé, présente quatre paramètres α_u, α_v, u_0 et v_0 . Si nous voulons tenir compte des distorsions, il est nécessaire d'ajouter trois paramètres de distorsion radiale r_1, r_2, r_3 , deux paramètres de distorsion prismatique p_1, p_2 et deux paramètres de distorsion de décentrage d_1, d_2 . A ces paramètres propres à la caméra il faut ajouter les paramètres de position et d'orientation soit, respectivement, trois paramètres t_1, t_2, t_3 et quatre paramètres q_1, q_2, q_3 et q_4 avec l'utilisation de quaternions. Le total des paramètres est alors de $4 + 3 + 2 + 2 + 3 + 4 = 18$ paramètres pour chaque caméra. Différentes hypothèses et/ou contraintes permettent de réduire ce nombre :

- Dans tous les cas, nous pouvons poser la contrainte sur le repère du monde en le fixant sur la caméra gauche. Les paramètres extrinsèques de la caméra gauche ne sont donc plus estimés mais fixés à $(1, 0, 0, 0)^t$ pour l'orientation et $(0, 0, 0)^t$ pour la position.

- L'industrialisation de caméras intégrées dans des véhicules de série imposera l'utilisation de caméra à focale fixe dont les paramètres intrinsèques ne risquent pas de varier au cours du temps. Dans un système final embarqué nous pourrions donc nous passer de l'estimation de $4 + 3 + 2 + 2$ paramètres pour chacune des caméras. Les 36 paramètres à estimer ne deviendront plus que 14 paramètres. De plus, si nous appliquons la contrainte précédente, cela revient à estimer seulement 7 paramètres. Cela produit systématiquement sept paramètres de moins à estimer.

Diminuer le nombre de paramètres à estimer est très important et joue considérablement sur la dimension du système et sur les points nécessaires pour lancer l'estimation. Il faut donc profiter de toutes les hypothèses possibles en fonction de l'application. Dans un souci de généralité nous ne poserons que la première contrainte de manière systématique et nous testerons avec et sans estimation des paramètres intrinsèques.

Nous allons maintenant détailler les équations de la jacobienne de l'ajustement de faisceaux nécessaires à l'équation 2.10. Dans tous les cas nous avons besoin des jacobienes du modèle par rapport aux paramètres extrinsèques des caméras (position et orientation) :

$$\frac{\partial u, v}{\partial t_1, t_2, t_3} = \frac{\partial u, v}{\partial x_c, y_c, z_c} \cdot \frac{\partial x_c, y_c, z_c}{\partial t_1, t_2, t_3} \quad (2.16)$$

$$\frac{\partial u, v}{\partial q_1, q_2, q_3, q_4} = \frac{\partial u, v}{\partial x_c, y_c, z_c} \cdot \frac{\partial x_c, y_c, z_c}{\partial q_1, q_2, q_3, q_4} \quad (2.17)$$

La jacobienne décrite par 2.16 et 2.17 impose la connaissance des jacobienes suivantes :

$$\frac{\partial u, v}{\partial x_c, y_c, z_c} = \frac{\partial u, v}{\partial x_d, y_d} \cdot \frac{\partial x_d, y_d}{\partial x_c, y_c, z_c} \quad (2.18)$$

$$\frac{\partial x_c, y_c, z_c}{\partial q_1, q_2, q_3, q_4} = \left(\frac{\partial R}{\partial q_1} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \middle| \frac{\partial \mathbf{R}}{\partial q_2} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \middle| \frac{\partial \mathbf{R}}{\partial q_3} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \middle| \frac{\partial \mathbf{R}}{\partial q_4} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \right) \quad (2.19)$$

$$\frac{\partial x_c, y_c, z_c}{\partial t_1, t_2, t_3} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (2.20)$$

où

$$\frac{\partial u, v}{\partial x_d, y_d} = \begin{pmatrix} \alpha_u & 0 \\ 0 & \alpha_v \end{pmatrix} \quad (2.21)$$

$$\frac{\partial x_d, y_d}{\partial x_c, y_c, z_c} = \frac{\partial x_d, y_d}{\partial x_r, y_r} \cdot \frac{\partial x_r, y_r}{\partial x_c, y_c, z_c} \quad (2.22)$$

$$\frac{\partial x_r, y_r}{\partial x_c, y_c, z_c} = \begin{pmatrix} \frac{1}{z_c} & 0 & -\frac{x_c}{z_c^2} \\ 0 & \frac{1}{z_c} & -\frac{y_c}{z_c^2} \end{pmatrix} \quad (2.23)$$

$$\mathbf{R} = \begin{pmatrix} q_1^2 + q_2^2 - q_3^2 + q_4^2 & 2(q_2q_3 + q_1q_4) & 2(q_2q_4 + q_1q_3) \\ 2(q_2q_3 - q_1q_4) & q_1^2 - q_2^2 + q_3^2 - q_4^2 & 2(q_3q_4 + q_1q_2) \\ 2(q_2q_4 - q_1q_3) & 2(q_3q_4 - q_1q_2) & q_1^2 - q_2^2 - q_3^2 + q_4^2 \end{pmatrix} \quad (2.24)$$

et

$$\begin{aligned}
 \frac{\partial x_d, y_d}{\partial x_r, y_r} &= \begin{pmatrix} \frac{\partial x_d}{\partial x_r} & \frac{\partial x_d}{\partial y_r} \\ \frac{\partial y_d}{\partial x_r} & \frac{\partial y_d}{\partial y_r} \end{pmatrix} \\
 \frac{\partial x_d}{\partial x_r} &= 1 + r_1\delta + r_2\delta^2 + \\
 &\quad r_3\delta^3 + 2x_r^2(r_1 + 2r_2\delta + 3r_3\delta^2) + 2x_r(p_1 + 3d_1) + 2d_2y_r \\
 \frac{\partial x_d}{\partial y_r} &= 2x_ry_r(r_1 + 2r_2\delta + 3r_3\delta^2) + 2y_r(p_1 + d_1) + 2d_2x_r \\
 \frac{\partial y_d}{\partial x_r} &= 2x_ry_r(r_1 + 2r_2\delta + 3r_3\delta^2) + 2x_r(p_2 + d_2) + 2d_1x_r \\
 \frac{\partial y_d}{\partial y_r} &= 1 + r_1\delta + r_2\delta^2 + \\
 &\quad r_3\delta^3 + 2y_r^2(r_1 + 2r_2\delta + 3r_3\delta^2) + 2y_r(p_2 + 3d_2) + 2d_1x_r \\
 \delta &= x_r^2 + y_r^2 \\
 \frac{\partial \mathbf{R}}{\partial q_1} &= \begin{pmatrix} 2q_1 & 2q_4 & 2q_3 \\ -2q_4 & 2q_1 & 2q_2 \\ -2q_3 & -2q_2 & 2q_1 \end{pmatrix} \\
 \frac{\partial \mathbf{R}}{\partial q_2} &= \begin{pmatrix} 2q_2 & 2q_3 & 2q_4 \\ 2q_3 & -2q_2 & 2q_1 \\ 2q_4 & -2q_1 & -2q_2 \end{pmatrix} \\
 \frac{\partial \mathbf{R}}{\partial q_3} &= \begin{pmatrix} -2q_3 & 2q_2 & -2q_1 \\ 2q_2 & 2q_3 & 2q_4 \\ -2q_1 & 2q_4 & -2q_3 \end{pmatrix} \\
 \frac{\partial \mathbf{R}}{\partial q_4} &= \begin{pmatrix} 2q_4 & 2q_1 & 2q_2 \\ -2q_1 & -2q_4 & 2q_3 \\ 2q_2 & 2q_3 & 2q_4 \end{pmatrix}
 \end{aligned} \tag{2.25}$$

Dans le cas où nous estimons les paramètres intrinsèques et les coefficients de distorsion des caméras, les jacobienues du modèle de caméra par rapport aux paramètres α_u, α_v, u_0 et v_0 et par rapport aux coefficients de distorsions $r_1, r_2, r_3, p_1, p_2, d_1$ et d_2 sont :

$$\frac{\partial u, v}{\partial \alpha_u, \alpha_v, u_0, v_0} = \begin{pmatrix} 1 & 0 & x_d & 0 \\ 0 & 1 & 0 & y_d \end{pmatrix} \tag{2.26}$$

$$\frac{\partial u, v}{\partial r_1, r_2, r_3, d_1, d_2, p_1, p_2} = \frac{\partial u, v}{\partial x_d, y_d} \cdot \frac{\partial x_d, y_d}{\partial r_1, r_2, r_3, d_1, d_2, p_1, p_2} \tag{2.27}$$

La jacobienne des coordonnées normalisées distordues en fonction de ses coordonnées de distorsion est donnée par :

$$\frac{\partial x_d, y_d}{\partial r_1, r_2, r_3, d_1, d_2, p_1, p_2} = \begin{pmatrix} x_r\delta & x_r\delta^2 & x_r\delta^3 & \delta & 0 & 3x_r^2 + y_r^2 & 2x_ry_r \\ y_r\delta & y_r\delta^2 & y_r\delta^3 & 0 & \delta & 2x_ry_r & x_r^2 + 3y_r^2 \end{pmatrix} \tag{2.28}$$

Schématiquement, la structure de la jacobienne dans le cas de deux caméras et de cinq points observés est du type :

		$\mathbf{k}_1$		$\mathbf{k}_2$		$\mathbf{T}_j$		$\mathbf{M}_i$				
		kp	kd	kp	kd	1	2	1	2	3	4	5
$\hat{\mathbf{m}}_i^1$	1	■	■			■		■				
	2								■			
	3									■		
	4										■	
	5											■
$\hat{\mathbf{m}}_i^2$	1			■	■		■	■				
	2			■	■		■		■			
	3			■	■					■		
	4			■	■						■	
	5			■	■							■

avec :

- les cases blanches correspondant à des valeurs nulles,
- $\mathbf{k}_j$ paramètres de la caméra j composés de $\mathbf{k}_p$, paramètres intrinsèques et $\mathbf{k}_d$ paramètres de distorsions,
- $\mathbf{T}_j$ paramètres extrinsèques de la caméra j (position et orientation),
- $\mathbf{M}_i$ coordonnées du point 3D i ,
- $\hat{\mathbf{m}}_i^j$ mesures des projections du point 3D i dans la caméra j .

Grâce à l'implémentation proposée en annexe A, la prise en compte des valeurs nulles simplifie considérablement la convergence du processus de réestimation.

2.4 Autocalibration du modèle d'erreur

L'autocalibration du modèle d'erreur basée sur 1.4.3 exploite, après chaque recalibration, une séquence d'images de disparité. Le modèle statistique permettant de calculer σ_d est mis à jour en fonction des points corrélés de la stéréovision.

2.5 Application à l'odométrie optique

L'odométrie mécanique pose souvent problème que ce soit en robotique ou dans les ITS. Elle est certainement la méthode de localisation la plus couramment utilisée pour les robots et engins mobiles disposant d'une structure de locomotion à roues. Le principe consiste à déduire une position de façon incrémentale, à partir de la vitesse et de la géométrie des roues. Bien que la mise en pratique soit très simple, elle peut poser problème pour des systèmes complexes. Le diamètre des roues doit bien évidemment être connu mais d'autres dimensions comme l'entraxe ou le point de contact avec le sol doivent également être pris en compte. Ces derniers paramètres peuvent poser problème, surtout pour un robot en environnement extérieur mais aussi dans les cas de glissements ou de patinages.

L'odométrie sur un véhicule peut aussi poser problème à des faibles vitesses. En effet, la plupart des odomètres sont fiables pour des vitesses supérieures à $5km/h$, vitesse largement supérieure à certaines manœuvres comme dans le cas de l'aide au parking où les vitesses peuvent descendre sous les $1km/h$. Il est à noter que certains équipementiers disposent d'odomètres fiables jusqu'à $0.3km/h$ mais leur disponibilité n'est certainement pas répandue à l'ensemble des véhicules de tourisme.

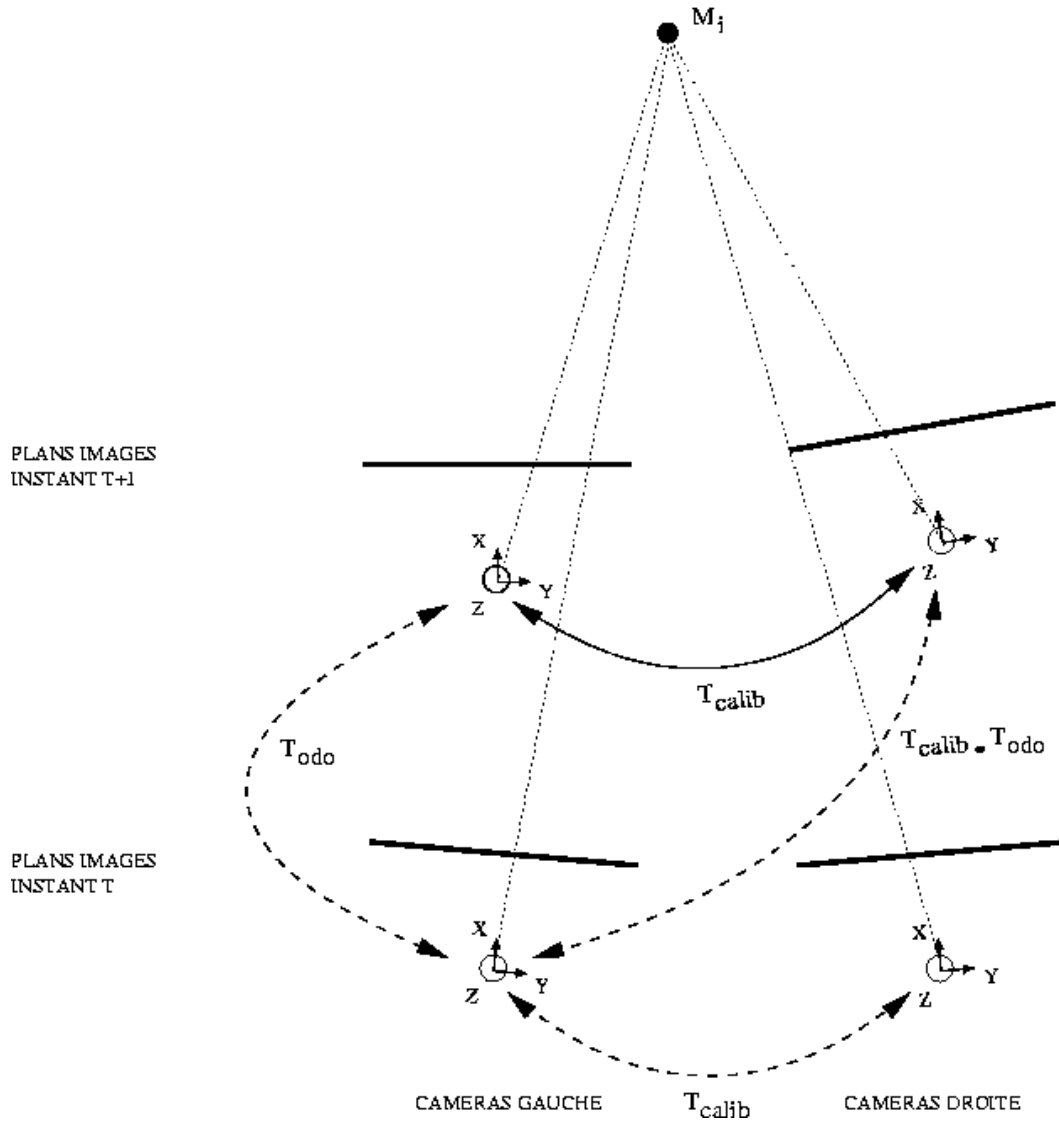


FIG. 2.4 – Evaluation des paramètres de calibration à l'aide de deux paires d'images successives dans le temps. Les paramètres extrinsèques de la caméra gauche à l'instant t sont fixés ainsi que la transformation T_{odo} entre deux positions successives de la caméra gauche en fonction des données odométriques. L'estimation porte alors sur la transformation T_{calib} et la transformation composée $T_{calib} \cdot T_{odo}$ en plus de la réestimation des positions des points M_i . Nous faisons l'hypothèse que la transformation T_{calib} ne varie pas entre les deux instants t et $t + 1$.

De plus, l'exploitation d'une information odométrique indépendante du système de vision implique la gestion d'un modèle d'erreur lié au caractère asynchrone des deux systèmes augmentant l'incertitude sur nos valeurs. Le fait de tirer partie des images servant à nos traitements de détection d'obstacles, par exemple pour obtenir l'information odométrique, permet de négliger ces erreurs.

La nécessité d'avoir une position locale fiable est pourtant incontournable. On l'a vu pour la recalibration mais c'est aussi le cas pour la fusion de données, elles-mêmes acquises à des instants différents sur une plate-forme mobile. Les méthodes mises en place pour la recalibration sont très proches d'une estimation de mouvement. En effet, c'est le but des méthodes photogrammétriques que d'estimer le déplacement entre deux positions d'une caméra mobile. Le modèle décrit précédemment est très facilement applicable à l'odométrie optique.

L'application de l'ajustement de faisceaux à l'odométrie optique revient à estimer non plus les paramètres intrinsèques, extrinsèques et la structure (les points $3D$) mais juste les paramètres extrinsèques, en faisant l'hypothèse que les intrinsèques ne bougent pas entre deux vues. Sur la base du formalisme utilisé jusqu'à présent, $\mathbf{X}_0$ représente le vecteur de coordonnées des points de Harris appariés d'une caméra à un instant t à la même caméra à l'instant $t + 1$. La distance parcourue entre les instants t et $t + 1$ représente la transformation rigide (translation et rotation) que l'on souhaite évaluer. Par exemple, dans le cas d'un système stéréoscopique, l'odométrie optique peut se baser sur l'appariement de points d'intérêt de deux images successives de la caméra gauche. Le vecteur de paramètres $\mathbf{P}_0$ est constitué des paramètres de la caméra mobile avec $\mathbf{P}_0 = (\mathbf{a}_t^t, \mathbf{a}_{t+1}^t, \mathbf{b}_1^t, \dots, \mathbf{b}_n^t)^t$ avec $\mathbf{a}$ vecteur de paramètres de la caméra constitué ici de la rotation $\mathbf{R} = (q_1, q_2, q_3, q_4)^t$ et la translation $\mathbf{t} = (t_x, t_y, t_z)^t$. On fixe ici $\mathbf{a}_t^t = (1, 0, 0, 0)^t$ pour le repère de la caméra à l'instant t et on estime $\mathbf{a}_{t+1}^t$.

Le facteur d'échelle n'est pas un problème puisque les degrés de liberté donnés par la réestimation de la structure des points $3D$ sont fixés. En effet, le système étant considéré comme calibré (ou recalibré), les points $3D$ ne sont pas réestimés. En pratique cela s'écrit grâce aux jacobiniennes où les termes relatifs aux points $3D$ sont égaux à zéro, $B_{ij} = 0$.

Les résultats sont rendus robustes par une approche RANSAC avec des temps d'exécution proche des $10Hz$. Son utilisation est donc tout à fait adaptée aux systèmes embarqués tels que les robots ou les véhicules.

Le lecteur intéressé pourra trouver dans [Mallet et al., 2000] des résultats obtenus au LAAS sur des séquences complexes, avec vérité terrain fournie par D-GPS.

2.6 Résultats

Comme nous l'avons vu précédemment, il existe deux méthodes différentes pour résoudre le facteur d'échelle. Cette section de résultats sera donc divisée en deux paragraphes. Le premier présente la calibration à partir d'une seule paire stéréoscopique. Le second présentera les résultats de calibration à partir d'un banc de stéréovision en mouvement, ce dernier étant connu par odométrie.

Les données de synthèse générées représentent un parallélépipède rectangle de $9 \times 12 \times 10$ mètres dont on va parcourir le volume par pas de 1 mètre (cf. figure 2.5). Les points $3D$ obtenus sont

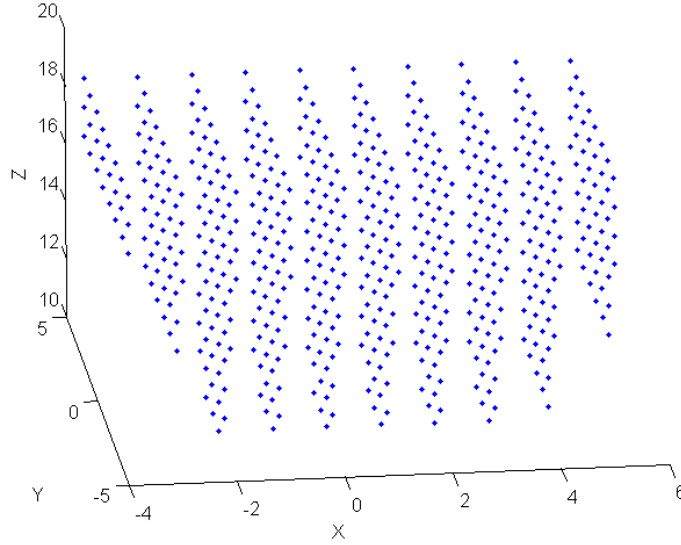


FIG. 2.5 – Objet de synthèse

alors projetés dans les images des deux caméras. Ce jeu de données représente nos points 3D et 2D que nous allons perturber dans les paragraphes suivants afin de tester la robustesse de nos algorithmes. Dans l'ensemble de cette section nous allons introduire 10% de faux appariements pour chaque test de manière à reproduire le pire comportement de notre appariement de points.

Il est évident que l'occupation volumique de cet objet de synthèse est parfaite et que les points rencontrés dans les environnements du véhicule ne seront jamais aussi bien répartis. On peut toutefois faire l'hypothèse que rares seront les cas où les points d'intérêts seront tous coplanaires (tous sur la route par exemple). Si c'est effectivement le cas, l'échec de calibration sera détecté au bout d'un certain nombre d'itérations et la calibration sera relancée sur une autre paire d'images en espérant que le cas ne se reproduise pas. Sauf le cas d'une route en plein désert cette hypothèse ne peut être considérée comme trop contraignante.

Le système de stéréovision est simulé de la manière suivante :

- focales 8mm,
- matrice $1/3''$ (4.8×3.6)mm,
- résolution des images 512×384 pixels,
- orientation de la caméra droite par rapport à la caméra gauche : $\mathbf{q}_1 = (1 \ 0 \ 0 \ 0)^t$,
- position de la caméra droite par rapport à la caméra gauche : $\mathbf{t}_1 = (0 \ 0.5 \ 0)^t$.

2.6.1 calibration ponctuel

Dans la description de la réestimation nous avons vu que les différents éléments pouvant influencer les résultats sont :

- les estimées initiales des points 3D,
- les mesures 2D,
- les estimées initiales des extrinsèques,
- les estimées initiales des intrinsèques,

- le pourcentage de faux appariements des mesures 2D,
- et le bruit sur les trois points choisis pour fixer le facteur d'échelle.

Une évaluation exhaustive devrait tenir compte de toutes les combinaisons possibles des variations de ces paramètres. Dans un but de lisibilité nous ne présenterons ici que les résultats les plus pertinents.

Ce paragraphe s'intéresse exclusivement à la calibration des extrinsèques, c'est-à-dire position et orientation de la caméra droite par rapport à la caméra gauche. Nous faisons l'hypothèse que les intrinsèques de chacune des caméras sont connues. Chacun de ces tests a été réalisé avec les estimées des extrinsèques suivantes représentant une décalibration de plus en plus prononcée :

- $\mathbf{q}_1 = (0.9 \ 0.03 \ 0.03 \ 0.03)^t$ et $\mathbf{t}_1 = (0.1 \ 0.4 \ 0.1)^t$
- $\mathbf{q}_1 = (0.8 \ 0.06 \ 0.06 \ 0.06)^t$ et $\mathbf{t}_1 = (0.1 \ 0.4 \ 0.1)^t$
- $\mathbf{q}_1 = (0.7 \ 0.1 \ 0.1 \ 0.1)^t$ et $\mathbf{t}_1 = (0.1 \ 0.4 \ 0.1)^t$

Les résultats étant identiques pour ces trois cas, nous n'en présenterons qu'un jeu.

Bruit sur les estimées 3D Ici, nous allons différencier deux groupes de points et appliquer différemment le bruit sur les points 3D. Les figures 2.6 et 2.7 présentent le cas de bruit sur les points 3D sauf pour les trois points utilisés pour contraindre le facteur d'échelle. Nous remarquons que dans un cas de bruit gaussien d'écart type $\sigma_{XYZ} = 1$ mètre comme dans l'autre $\sigma_{XYZ} = 10$ mètres, les résultats de la réestimation sont excellents avec une erreur de réestimation de l'ordre de 4.10^{-4} mètre par rapport à l'objet de synthèse. Par contre, dès lors que nous appliquons un bruit sur ses trois points de référence comme sur la figure 2.8, nous voyons apparaître le problème du facteur d'échelle déformant l'objet de synthèse.

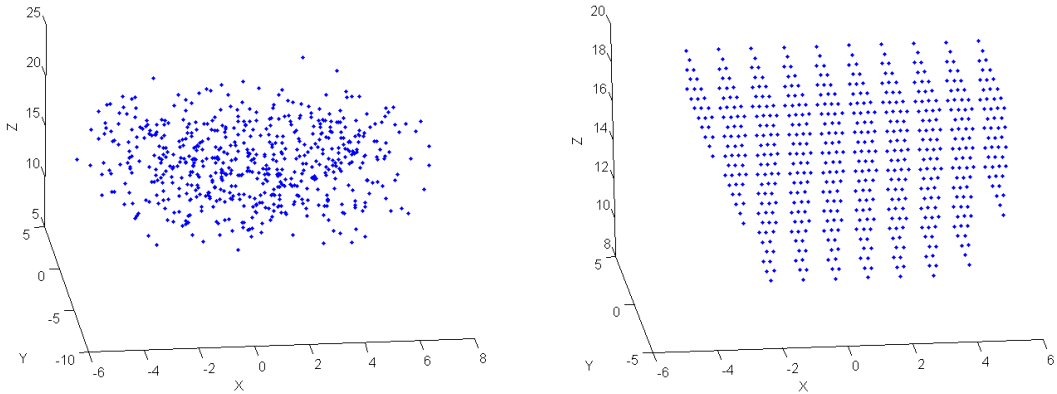


FIG. 2.6 – [gauche] Objet de synthèse avec bruit (gaussien $\sigma_{XYZ} = 1$ mètre), [droite] objet réestimé. Erreur moyenne des points réestimés : $\bar{e}_{XYZ} = 4.09.10^{-4}m$

Bruit sur les mesures 2D Le bruit sur les mesures 2D des projections des points dans les images dépend du système d'appariement des points d'intérêt utilisé. Les figures 2.9 et 2.10 présentent les résultats de réestimation pour des variances respectives de $\sigma_{uv} = 0.5$ pixel et $\sigma_{uv} = 2$ pixels. La différence de réestimation est très importante, entre $\bar{e}_{XYZ} = 0.52m$ et $\bar{e}_{XYZ} = 3.41m$. Il est donc très important de connaître les performances de l'appariement des points d'intérêt. Dans notre cas, [Jung et al., 2001] montrent une erreur moyenne autour de 0.5 pixel pour des images présentant des points de vue différents avec d'importants changements d'échelle.

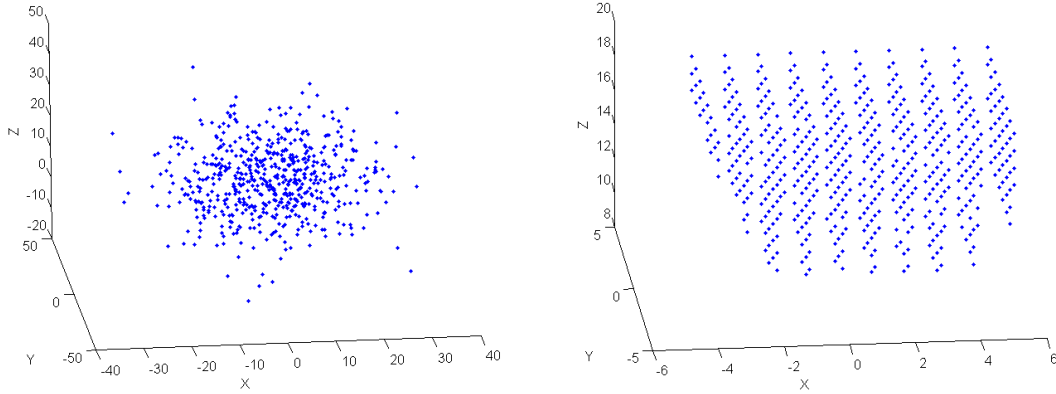


FIG. 2.7 – [gauche] Objet de synthèse avec bruit (gaussien $\sigma_{XYZ} = 10$ mètre), [droite] objet réestimé. Erreur moyenne des points réestimés : $\bar{e}_{XYZ} = 6.62 \cdot 10^{-4}m$

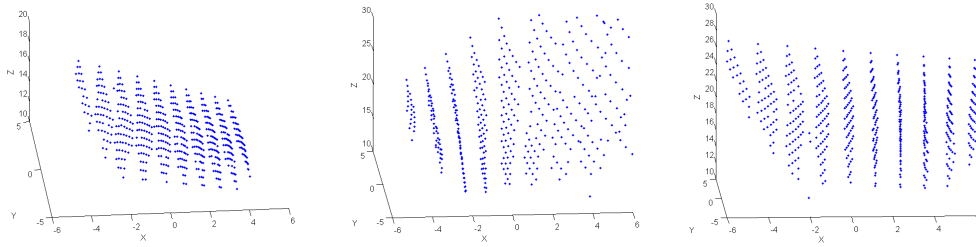


FIG. 2.8 – Objet de synthèse réestimé avec données initiales bruitées : bruit sur les estimées 3D $\sigma_{XYZ} = 10m$ et pour les trois points utilisés pour le facteur d'échelle : [gauche] bruit 3D $\sigma_{XYZ} = 0.1m$, [milieu] bruit 3D $\sigma_{XYZ} = 0.2m$ et [droite] bruit 3D $\sigma_{XYZ} = 0.3m$.

L'appariement des points d'intérêt est aisé pour notre application vu la faible distance entre les deux caméras permettant un grand champ de recouvrement des deux vues.

Bruit sur les mesures 2D et estimées 3D Les bruits 2D et 3D risquent d'apparaître simultanément sur un système réel. La figure 2.11 montre les résultats pour un bruit 3D gaussien $\sigma_{XYZ} = 10m$ et un bruit 2D gaussien $\sigma_{uv} = 0.5pixel$ pour rester dans un domaine de bruit cohérent avec notre application. L'erreur de réestimation est alors de $1.24m$. Il est important de voir ici le comportement du bruit d'un vrai système stéréo. L'erreur sur les distances estimées est quadratique avec la distance. C'est-à-dire que si nous choisissons des points estimés proches de la caméra nous pouvons espérer (sauf faux appariement filtré par l'utilisation d'algorithme robuste comme RANSAC) limiter l'erreur sur les trois points utilisés pour contraindre le facteur d'échelle. Nous verrons plus de détail en section 5.2 de résultats sur images réelles.

2.6.2 calibration avec redondance temporelle

La redondance temporelle des points appariés (cf. figure 2.4) multiplie le nombre de faisceaux pour un seul point contribuant à renforcer l'information sur celui-ci. Les résultats avec redondance temporelle des points sont totalement liés aux résultats précédents. De la même manière que le bruit appliqué sur les trois points se propage sur le facteur d'échelle estimé, le bruit appliqué sur l'odométrie influence directement ce dernier. Des figures identiques aux paragraphes précédents

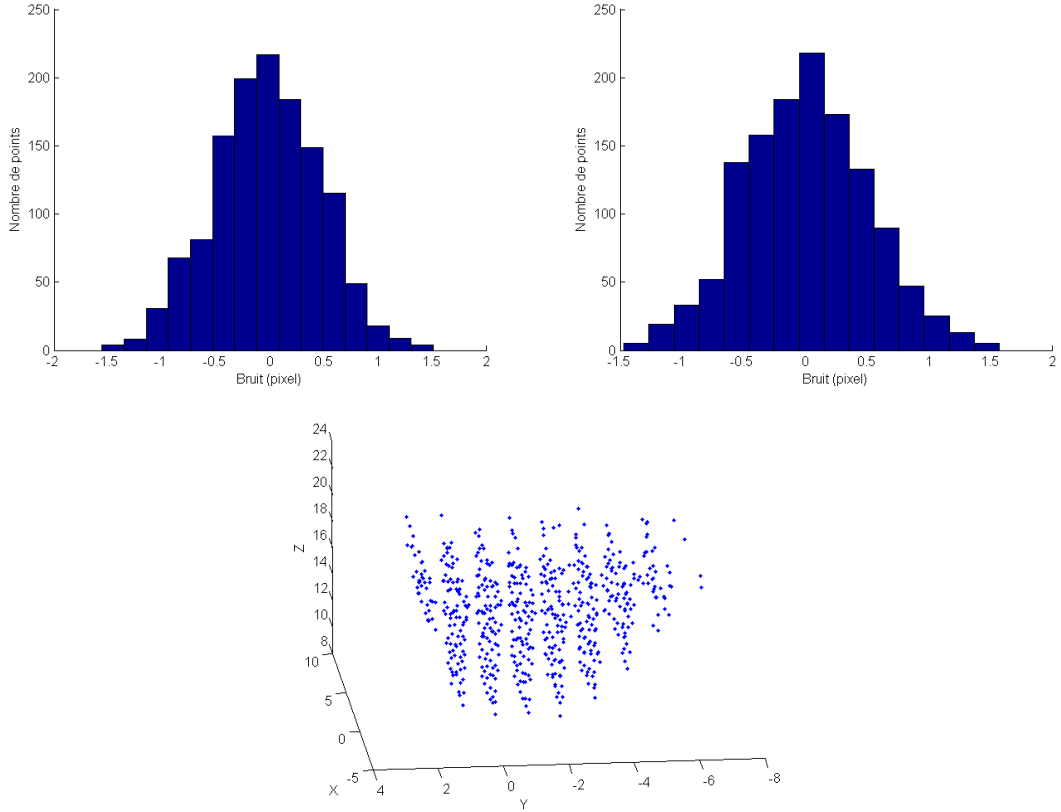


FIG. 2.9 – [haut] Bruit 2D ajouté aux mesures 2D u et v (bruit gaussien, $\sigma_{uv} = 0.5$ pixel), [bas] objet de synthèse réestimé. Erreur moyenne des points réestimés : $\bar{e}_{XYZ} = 0.52m$

pourraient être présentées ici avec les mêmes erreurs de réestimation. Dans ce paragraphe, nous allons nous intéresser à une séquence d'images de synthèse dont une paire stéréo est présentée en figure 2.12. Au début de la séquence le système est correctement calibré puis une divergence de la caméra droite est appliquée de manière à perturber la calibration. La figure 2.13 nous donne les images de disparité avant la décalibration [gauche], puis pendant [milieu] et après recalibration [droite]. Les erreurs des distances calculées par l'évaluation de la calibration sont visibles par des segments sur la figure 2.14. La figure 2.15 résume l'ensemble des résultats sur une séquence de synthèse. Sur cette dernière, les croix représentent le score de qualité de calibration si l'on joue la séquence sans recalibration en ligne. Ce score augmente lors de la décalibration du système (visible par la ligne en pointillé) et nous indique un problème dans la géométrie du capteur. Les points représentent le score de qualité de calibration après recalibration. Ce score est supérieur au seuil de recalibration fixé pour deux images :

- une pour la détection de la décalibration,
- une autre pour confirmer la décalibration et pour le recalibrage lui-même,

Nous remarquons que suite à la recalibration la qualité est améliorée même si elle semble un peu moins stable, effet visible sur la carte de disparité moins régulière qu'avant la décalibration. Cet effet peut-être limité en itérant à nouveau la calibration à partir du nouveau jeu de paramètres calculé. L'affinement de la calibration peut se déclencher jusqu'à ce que la variance sur le score d'évaluation de la calibration soit faible.

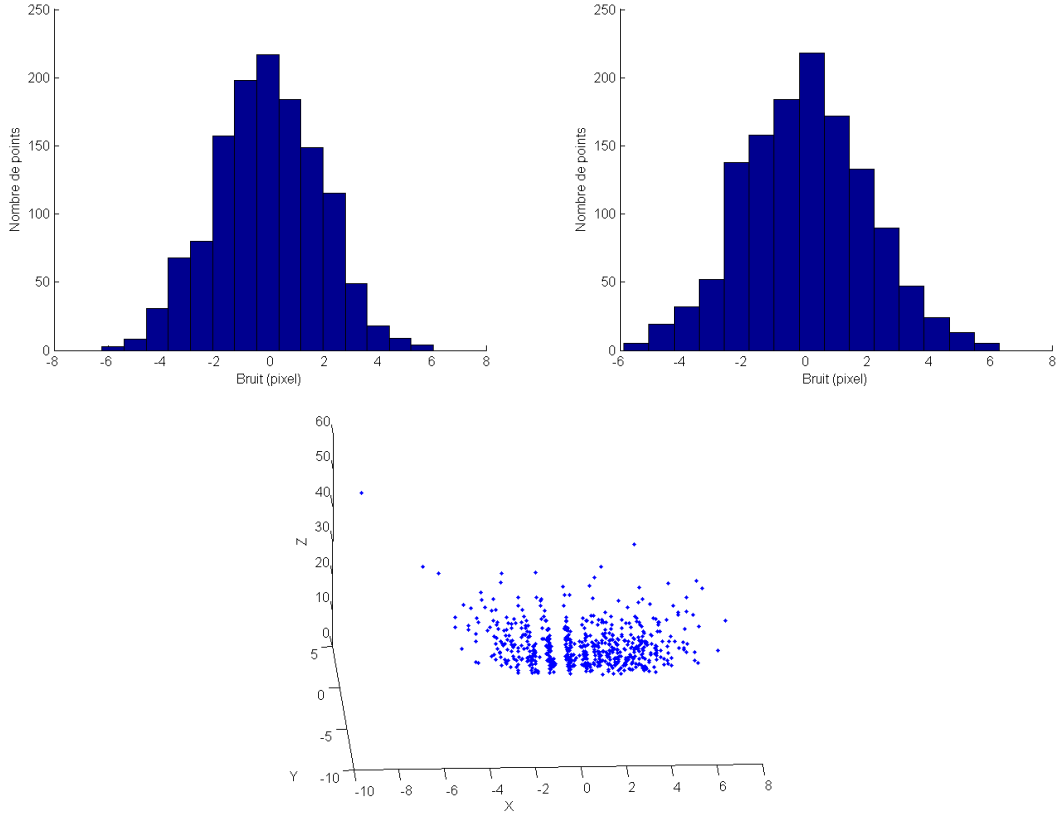


FIG. 2.10 – [haut] Bruit 2D ajouté aux mesures 2D u et v (gaussien, $\sigma_{uv} = 2$ pixels), [bas] objet de synthèse réestimé. Erreur moyenne des points réestimé : $\bar{e}_{XYZ} = 3.41m$

2.7 Conclusion et discussions

Nous avons vu dans ce chapitre une approche photogrammétrique pour la recalibration d'un système stéréoscopique basée sur les derniers paramètres évalués comme corrects. L'intérêt de cette approche permet de s'affranchir d'une autocalibration au prix d'une évaluation fine de la qualité de calibration.

Il peut être intéressant de voir dans quelle mesure le système décrit dans ce chapitre pourrait se transformer en autocalibration. Sans aller jusque là, on peut penser à une aide à la calibration en évitant l'intervention d'un opérateur. Pour faciliter la manipulation de calibration, on peut initialiser le système grâce à des mesures "à la main" de la transformation inter-caméras. En effet, dans le cas où les dimensions du banc sont mesurables (ni trop petit comme un microscope, ni trop grand comme les systèmes photogrammétriques satellitaires) et que la corrélation entre les images est robuste aux faiblesses de rectification (comme l'algorithme multi-résolution décrit en section 1.4), on peut obtenir une réestimation de l'ensemble des paramètres intrinsèques et extrinsèques du banc.

Au delà de ces considérations très applicatives, le résumé de notre méthode peut s'écrire ainsi : (1) le système est initialement calibré, (2) une tâche de fond exploite la géométrie épipolaire pour évaluer la qualité de calibration courant et (3) le système se recalibre si besoin. Le détail de cette dernière étape fait apparaître l'utilisation d'estimées initiales liées à la calibration *fort* précédent

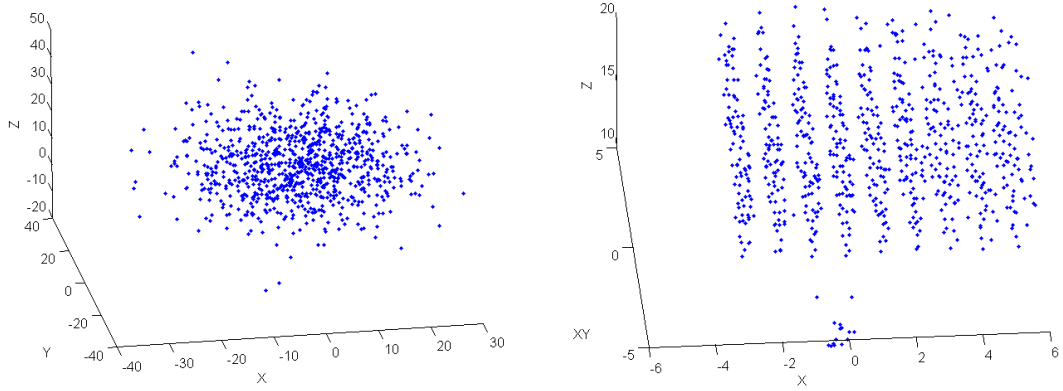


FIG. 2.11 – [gauche] Objet de synthèse avec bruit gaussien $\sigma_{XYZ} = 10$ mètres, bruit gaussien $\sigma_{uv} = 0.5$ pixel et 10% d'outliers), [droite] objet réestimé. Erreur moyenne des points réestimés : $\bar{e}_{XYZ} = 1.24m$



FIG. 2.12 – Paire d'images de synthèse

mais n'utilise pas la géométrie épipolaire puisque la triangulation est basée sur l'appariement des points d'intérêt. L'ensemble des paramètres de sorties de nos algorithmes se compose des paramètres intrinsèques (avec prise en compte des distorsions) et l'orientation des caméras l'une par rapport à l'autre. Cette méthode peut donc être qualifiée d'**autocalibration faible**. La dimension métrique peut être prise en compte en fonction de l'application et permet d'obtenir une **autocalibration forte**.

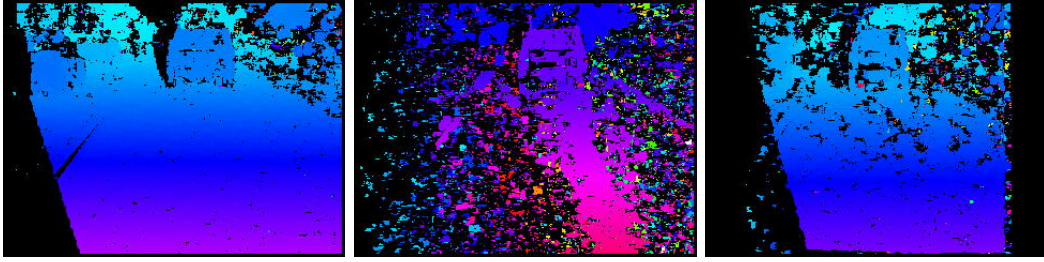


FIG. 2.13 – Résultats de cartes de disparité, [gauche] calibré, [milieu] décalibré, [droite] après recalibration



FIG. 2.14 – Affichage des distances entre appariement de points d'intérêt et appariement par stéréocorrespondance, [gauche] calibré, [milieu] décalibré, [droite] après recalibration

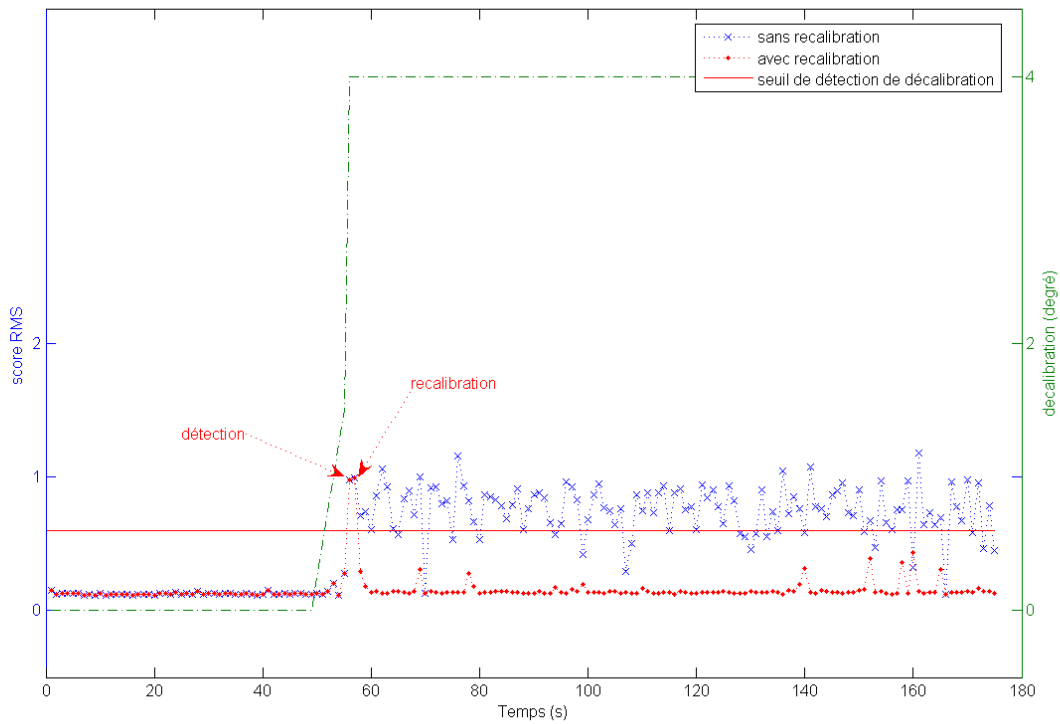


FIG. 2.15 – Résultats de recalibration sur images de synthèse

Bibliographie du chapitre

- [Bernstein et al., 1998] D. Bernstein et A. Kornhauser (1998). Map matching for personal navigation assistants. Dans *77th Annual meeting, The Transport Research Board*, Washington, D.C.
- [Boufama et al., 1995] B. Boufama et R. Mohr (1995). Epipole and fundamental matrix estimation using the virtual parallax property. Dans *International Conference of Computer Vision*, pages 1030–1036.
- [Faugeras et al., 1995] O. Faugeras, S. Laveau, L. Robert, G. Csurka, et C. Zeller (1995). 3d reconstruction of urban scenes from sequences of images. Rapport Technique 2572, INRIA, Sophia-Antipolis, France.
- [Fua, 2000] P. Fua (2000). Regularized bundle-adjustment to model heads from image sequences without calibration data. *International Journal of Computer Vision*, 38(2) :153–171.
- [Garcia, 2001] D. Garcia (2001). *Mesure de formes et de champs de déplacements tridimensionnels par stéréo-corrélation d'images*. Thèse de doctorat, INPT.
- [Hartley, 1993] H. Hartley (1993). Euclidean reconstruction from uncalibrated views. Dans *On the Second Europe-US Workshop on Invariance*, pages 187–202.
- [Hartley, 1994] H. Hartley (1994). Self-calibration from multiple views with a rotating camera. Dans *3rd European Conference on Computer Vision*, pages 471–478, Stockholm.
- [Hartley, 1997] H. Hartley (1997). Kruppa's equations derived from the fundamental matrix. Dans *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 19, pages 133–135.
- [Jung et al., 2001] I. Jung et S. Lacroix (2001). A robust interest point matching algorithm. Dans *International Conference on Computer Vision*.
- [Levenberg, 1944] K. Levenberg (1944). A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2 :164–168.
- [Liu et al., 1990] Y. Liu, T. Huang, et O. Faugeras (1990). Determination of camera location from 2d to 3d line and point correspondances. Dans *IEEE Trans. on Pattern Analysis machine Intelligence*, volume 12, pages 28–37.
- [Longuet-Higgins, 1981] H. Longuet-Higgins (1981). A computer algorithm for reconstructing a scene from two projections. *Nature*, 293(133-135).
- [Lourakis et al., 2004] M. Lourakis et A. Argyros (2004). The design and implementation of a generic sparse bundle adjustment software package based on the levenberg-marquardt algorithm. Rapport Technique 340, Institute of Computer Science - FORTH, Heraklion, Crete, Greece. Available from <http://www.ics.forth.gr/~lourakis/sba>.
- [Lourakis et al., 1999] M. Lourakis et R. Deriche (1999). camera self-calibration using the singular value decomposition of the fundamental matrix : From point correspondances to 3d measurements. Rapport Technique 3748, INRIA, Sophia-Antipolis, France.

- [Lourakis et al., 2000] M. Lourakis et R. Deriche (2000). Camera self-calibration using the singular value decomposition of the fundamental matrix : The case of varying intrinsic parameters. Rapport Technique 3911, INRIA, INRIA Sophia-Antipolis.
- [Luong et al., 1993] Q. Luong et O. Faugeras (1993). Self-calibration of a stereo rig from unknown camera motions and point correspondences. Rapport Technique 2014, INRIA, Sophia-Antipolis, France.
- [Luong et al., 1995] Q. Luong et O. Faugeras (1995). The fundamental matrix : Theory algorithms, and stability analysis. *International journal of Computer Vision*, 17 :43–75.
- [Ma, 1996] S. D. Ma (1996). A self-calibration technique for active vision systems. Dans *IEEE Transactions on Robotics and Automation*, volume 12, pages 114–120.
- [Mallet et al., 2000] A. Mallet, S. Lacroix, et L. Gallo (2000). Position estimation in outdoor environments using pixel tracking and stereovision. Dans *IEEE International Conference on Robotics and Automation, San Francisco, Ca (USA)*, pages 3519–3524.
- [Mohr et al., 1996] R. Mohr et B. Triggs (1996). Projective geometry for image analysis. Dans *ISPRS workshop tutorial*, Vienna, Austria.
- [Pollefeys et al., 1998] M. Pollefeys, R. Koch, et L. V. Gool (1998). Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. Dans *ICCV*, pages 90–95.
- [Shum et al., 1999] H. Shum, Q. Ke, et Z. Zhang (1999). Efficient bundle adjustment with virtual key frames : A hierarchical approach to multi-frame structure from motion. Dans *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR'99)*.
- [Triggs et al., 2000] B. Triggs, P. McLauchlan, R. Hartley, et A. Fitzgibbon (2000). Bundle adjustment – a modern synthesis. Dans B. Triggs, A. Zisserman, et R. Szeliski, editors, *Vision Algorithms : Theory and Practice*, volume 1883 of *Lecture Notes in Computer Science*, pages 298–372. Springer-Verlag.

Chapitre 3

Détection d'obstacles

Sommaire

3.1	Etat de l'art	66
3.1.1	Capteurs actifs	67
3.1.2	Capteurs passifs	68
3.1.3	Conclusion	69
3.2	Détection des obstacles sur la voie de circulation	70
3.2.1	Détection de la route	70
3.2.2	Modélisation du profil longitudinal de route	70
	Cas simple, sans dévers ni roulis	71
	Prise en compte du dévers	72
	Bilan de la création du profil longitudinal	74
3.2.3	Extraction des obstacles dans le profil longitudinal	74
3.2.4	Résultats	75
	Sans roulis ni dévers	75
	Variation de pente	77
	Roulis	80
	Cas général	83
3.3	Caractérisation des obstacles	86
3.3.1	Extraction des dimensions des obstacles	87
3.3.2	Résultats	88
3.4	Conclusion	88

Malgré la prévision de [Hancock, 1999], comme quoi la problématique de la détection d'obstacles ne sera jamais résolue, plusieurs méthodes ont été proposées depuis. Ce chapitre présente notre contribution à la réalisation d'une fonction de détection d'obstacles sur un véhicule intelligent. Ces travaux ont été réalisés dans le cadre du projet ARCOS¹⁶, action fédérative du PREDIT. Ce projet de recherche précompétitive avait pour but d'améliorer la sécurité routière avec une approche *véhicule-infrastructure-conducteur*. Pour cela, les quatre fonctions suivantes, essentielles à la sécurité routière, ont été choisies, sur la base des données accidentologiques afin de réduire de 30% les sinistres :

- gérer les interdistances entre véhicules,
- prévenir les collisions sur obstacles fixes, arrêtés ou lents,
- prévenir les sorties de route,
- alerter les véhicules en amont d'accidents/incidents.

Ces différentes fonctions concernent des accidents dont les causes sont très proches mais dont les effets peuvent différer (un objet au bord d'une route peut amener une collision ou une sortie de route). Afin de garder une cohérence sur le travail de ces fonctions malgré la diversité des approches (synergie entre sciences de l'ingénieur, sciences humaines et sciences sociales), une structure en dix thèmes a émergé :

1. techniques de perception,
2. autres techniques de mesure : visibilité et adhérence,
3. traitement de l'information et élaboration de l'action,
4. communication,
5. simulation / évaluation et accidentologie,
6. système homme-machine,
7. acceptabilité sociale individuelle,
8. autres aspects socio-techniques collectifs,
9. moyens expérimentaux,
10. pilotage fonctions, assistance technique ARCOS.

Plus particulièrement, le travail présenté dans ce chapitre de thèse a été réalisé dans le cadre du thème 1 de "Techniques de perception", pour la fonction de prévention des collisions. Cette problématique est très importante au vu des données accidentologiques qui montrent que 35% des accidents (3000 tués) se produisent sans tiers (donc dus à des sorties de route) et que 52% (environ 4000 tués) sont dus à des collisions entre deux véhicules.

Dans ce chapitre, nous allons tout d'abord décrire les différentes techniques existantes de détection d'obstacles en fonction des capteurs utilisés. La deuxième section présente nos algorithmes de détection d'obstacles sur une route de géométrie variable (variation de pente, dévers/roulis). A partir de cette détection, la section suivante caractérise les obstacles en objets géométriques. Nous terminerons ce chapitre par une conclusion et une discussion sur la fonction de détection d'obstacles mise en œuvre.

3.1 Etat de l'art

Un état de l'art détaillé a été présenté dans [ARCOS, 2002] dans le cadre d'un livrable du projet ARCOS. Cette section reprend les grandes lignes de ce document avec une mise à jour

¹⁶Action de Recherche pour une COnduite Sécurisée

de la bibliographie entre 2002 et 2005.

Avant de concevoir et réaliser une fonction de détection d'obstacles sur un système embarqué, il convient de choisir le ou les capteurs à utiliser. On peut distinguer deux catégories de capteur : (1) les capteurs actifs tels que les radars et les télémètres lasers, et (2) les capteurs passifs tels que les systèmes de vision.

3.1.1 Capteurs actifs

Les capteurs actifs auxquels sont incorporés un émetteur produisent leur propre énergie pour illuminer la scène dans la bande spectrale du récepteur. Les plus utilisés sont les radars et les télémètres lasers et beaucoup plus rarement les ultra-sons qui restent réservés aux obstacles proches pour un contexte parking, par exemple.

RADAR (*Radio Detection And Ranging*) : L'avantage des radars réside dans leur grande portée et leur indépendance aux conditions météorologiques comme la pluie ou la neige présentée dans [Jouannin, 1999, Kato et al., 2001, Kuroda et al., 1998, Langer, 1997]. Cette portée peut atteindre 200m avec une résolution de 0.1m en profondeur dans [Langer, 1997]. Il n'est donc pas étonnant de les retrouver dans les systèmes ACC¹⁷ qui permettent de réguler la vitesse du véhicule en fonction de son environnement. L'avantage certain de ce capteur est donc sa portée mais (1) il est très sensible aux problèmes des pentes et virages et (2) il est très difficile d'accès avec très peu de distributeurs en France. Il est donc très peu exploité par les laboratoires français. De plus, les radars sont tributaires de la surface de réflexion des obstacles, surface importante pour une voiture (10m² en moyenne) mais beaucoup plus réduite pour un piéton (environ 2m²). A cela, il faut ajouter une résolution angulaire encore très faible ne permettant pas de détecter les petits obstacles. La détection est donc beaucoup moins fiable et peut même être inexistante pour de petits obstacles. Un autre inconvénient pour notre système de détection d'obstacles est un angle d'ouverture limité à 10° horizontalement et 3° verticalement non compatible avec nos spécifications présentées en chapitre 1.

Télémètre laser ou LIDAR (*Light Detection And Ranging*) : La télémétrie laser est déjà largement utilisée par la communauté robotique pour la navigation de robots en terrain inconnu [Nashashibi et al., 1993]. Le principe de fonctionnement est une mesure du temps mis par la lumière réfléchiée par l'obstacle se trouvant dans l'axe de tir du laser. Le calcul de la distance en fonction de ce temps a été présenté par [Hancock, 1999] ainsi qu'un système complet de détection d'obstacles dans [Hancock, 1997]. Le balayage 2D du laser retourne une image de distance liée à une image de réflectance (l'énergie qui revient dans le capteur dépend de l'albedo c'est à dire du coefficient d'absorption de la surface détectée et de l'angle d'incidence) ; la détection d'obstacles travaillant sur les discontinuités de celles-ci. [Xie et al., 1994] utilisent un télémètre laser couplé à un capteur CCD, la détection d'obstacles étant réalisée dans l'image d'intensité et validée par la distance donnée par la télémétrie. Quant à [Trassoudaine, 1993], une segmentation de l'image de profondeur lui permet d'extraire les obstacles. Celui-ci, lors du projet ARCOS, exploite un laser 3D avec N lignes de balayage en site pour résoudre le problème des pentes. L'avantage direct est la résolution au détriment d'un encombrement important du système, d'une fragilité accrue et d'un prix élevé.

¹⁷ Adaptive Cruise Control

Capteur optique à temps de vol : Récemment, un nouveau type de capteur est apparu sur le marché par CSEM¹⁸ en particulier suite aux travaux de l'Université de Siegen. Des produits encore en cours de développement sont commercialisés par CSEM en Europe et CANESTA aux USA. La scène est illuminée par une matrice de LEDs, par une lumière diffuse modulée dans l'infra-rouge : la mesure est réalisée en chaque photosite. Le capteur à temps de vol (TOF¹⁹) permet d'acquérir une image de profondeur d'une résolution de 320×240 pixels à une fréquence de 24Hz . Des premiers systèmes commencent à sortir [Weingarten et al., 2004] mais il paraît encore difficile d'embarquer un tel système pour la détection d'obstacles lointains. La qualité des données $3D$ reste encore très dépendante du temps d'intégration des photosites et la portée du système reste très limitée (sept mètres pour le modèle utilisé au LAAS-CNRS). Le prix est très élevé, les versions existantes étant encore des prototypes, non calibrées. De nouveaux modèles prometteurs devraient voir le jour avec une plus grande robustesse vis-à-vis des fortes illuminations et avec des portées plus importantes.

En conclusion sur les capteurs actifs, les radars sont très robustes pour une détection d'obstacles proches et lointains mais ne permettent pas la détection de petits obstacles. La télémétrie laser fonctionne bien pour des obstacles lointains mais coûte encore beaucoup trop cher pour une intégration sur des véhicules de tourisme, surtout si on intègre un balayage $3D$ pour milieu routier.

3.1.2 Capteurs passifs

Un capteur passif est un capteur qui reçoit une énergie émise sans que lui-même n'irradie la scène. Ces capteurs, dans le monde de la robotique, sont majoritairement des caméras. Deux approches sont alors possibles : l'utilisation d'une seule caméra ou l'exploitation de plusieurs points de vue avec plusieurs caméras reliées de manière rigide sur le véhicule. L'intérêt des caméras par rapport aux capteurs actifs tient surtout dans le champ de perception accessible et par la capacité d'interprétation et d'analyse des scènes perçues. En effet, là où un radar est limité à 12° de champ horizontal et 3° de champ vertical, une caméra peut facilement atteindre respectivement les 50° et 30° . De plus, le coût d'une caméra (environ 5 \$) reste largement inférieur à un radar ou un LIDAR. Par contre, la difficulté de la vision artificielle est le coût possible des algorithmes puisque l'image en elle-même ne donne pas de mesures de distance comme un capteur actif. Il faut donc passer par des algorithmes de traitement d'image. Les résultats ne sont donc plus issus d'une mesure physique mais sont le fruit d'une suite combinatoire de calculs. De plus, une caméra, par rapport à un télémètre laser, n'a pas les mêmes capacités de détection pour des obstacles lointains. Il faut donc spécifier précisément les besoins de l'application afin de définir les paramètres de la caméra à utiliser (focale et taille du capteur entres autres).

Approche monocaméra : Généralement, l'approche monocaméra consiste en une interprétation de l'image avec une connaissance *a priori* de ce qu'est un obstacle (typiquement, un véhicule ou un piéton) et non en une reconstruction $3D$ de l'environnement. Cette interprétation se compose de deux étapes pour la détection d'obstacles. Tout d'abord, une extraction de primitives spécifiques aux obstacles permet de trouver l'information pertinente dans les images. Ces primitives peuvent être des informations de texture [Kalinke et al., 1998], de couleur [Aufreere et al., 2000], [Chapuis et al., 2000], des ombres portées [Steux, 2001], [Leeuwen et al., 2001], des segments [Betke et al., 1998], des régions en mouvement données par le flux optique [Batavia et al., 2001]

¹⁸<http://www.csem.ch/fs/vision.htm>

¹⁹Time Of Light

et bien d'autres encore. Suite à cette extraction de primitives, une analyse de celles-ci cherche à les mettre en correspondance à partir d'un modèle d'objet [Murphey et al., 2000], [Betke et al., 1998], [Hu et al., 2000]. Couplée à cette analyse, s'ajoute souvent une détection des zones d'intérêt dans les images [Bertozzi et al., 1997] ainsi qu'un suivi de l'objet détecté avec une analyse temporelle [Betke et al., 1998]. [Bertozzi et al., 1997], [Steux, 2001], [Matthews et al., 1996] et [Papageorgiou et al., 1998] présentent des systèmes complets de détection d'obstacles par monocaméra.

D'autres systèmes plus récents ont une approche hybride de détection de lignes et détection de contours comme cela est proposé dans [Jiang et al., 2002] et une approche sur le flux optique pour [Demonceaux et al., 2004].

Approche multicaméras :

– Appariements :

Les méthodes par appariements se basent sur la carte de disparité issue de l'appariement de primitives entre les différentes vues du capteur. Certains ont choisi de travailler sur des segments ou des contours [Franke et al., 2000], [Koller et al., 1994], [Labayrade et al., 2002] et d'autres sur des pixels [Williamson, 1998]. Il en résulte une carte dense pour la primitive *pixel* et une carte éparse pour les autres. Dans [Labayrade, 2004], les auteurs se donnent la possibilité de remonter à une carte de disparité dense à partir de sa première estimation éparse.

Généralement, le but de l'appariement de primitives permet une reconstruction 3D de l'environnement perçu par les caméras. C'est à partir de ce nuage de points 3D que les obstacles sont extraits. L'auteur de [Labayrade et al., 2002] travaille, quant à lui, directement sur la carte de disparité et se ramène, par le biais d'histogrammes sur les disparités des lignes de l'image, à un espace appelé *v-disparité*. L'extraction des obstacles, du profil de route et du point de fuite de l'image sont directement déduits de cette représentation.

– Rectification homographique :

Cette méthode exploite la rectification homographique sur une image de manière à ce que le plan de route soit identique dans les deux images. Une simple différence entre les images permet alors de discriminer les pixels qui ne correspondent pas à ce plan. L'homographie est appliquée *a priori* à l'aide d'une calibration préalable de la route. Cette méthode suppose donc que la route est plane. Une fois les objets au-dessus de la route mis en évidence, ils sont caractérisés par une analyse de connexité et de similarité.

3.1.3 Conclusion

Nous avons vu dans ce rapide état de l'art qu'un grand nombre de capteurs peuvent répondre au cahier des charges d'une fonction de détection d'obstacles. Ici, pour des raisons de coûts, nous avons écarté les capteurs actifs. De plus, il est raisonnable de penser que l'utilisation des caméras va se répandre de plus en plus sur les véhicules et qu'il sera donc possible d'utiliser une caméra pour plusieurs fonctions d'aide à la conduite, ce qui réduit les coûts de manière importante.

Le choix entre l'approche monocaméra et multicaméras a été naturel vu le contexte des recherches du groupe de Robotique et d'Intelligence Artificielle du LAAS-CNRS. Les travaux sur des environnements extérieurs non structurés ont amené, dans le groupe, une expertise sur l'utilisation de la stéréovision dense. L'extraction des obstacles étant une étude des points 3D

reconstruits à partir de l'image de disparité. Désormais, l'ajustement de faisceaux pourrait nous permettre, à partir d'une seule caméra en mouvement, de reconstruire une telle carte de disparité au détriment du temps de calcul, important dans la fonction de détection d'obstacles. Nous avons donc opté pour le banc de stéréovision dont les problèmes de calibration sont maintenant négligeables grâce aux algorithmes d'autocalibration ou de recalibration.

Depuis 2002, la contribution importante de [Labayrade, 2004] pour la détection d'obstacles a fourni une méthode robuste et rapide. Il n'en reste pas moins que l'extraction du profil de route reste compliquée surtout en présence de roulis et/ou dévers. La technique que nous allons présenter dans les sections suivantes reprend l'idée de la robustification grâce aux cumuls des histogrammes sur les lignes de l'image mais travaille dans l'espace tridimensionnel et non dans la *v-disparité*. Le fait de reconstruire systématiquement tous les points en $3D$ va nous permettre de corriger ligne par ligne l'effet du dévers et/ou du roulis. De plus, la caractérisation des obstacles par des boîtes englobantes sera directement extraite des histogrammes des points $3D$ d'où une robustesse des dimensions des obstacles détectés. Tout au long de ce chapitre, nous ferons des comparaisons entre notre approche travaillant dans l'espace $3D$ avec l'approche *v-disparité*.

3.2 Détection des obstacles sur la voie de circulation

La méthode décrite dans cette section se décompose en deux étapes dont les calculs intermédiaires sont conservés afin d'optimiser l'ensemble du temps de traitement. La première consiste en la création du profil longitudinal de la route, pendant dans l'espace $3D$ de la *v-disparité*. Cette étape est indispensable pour prendre en compte les variations de pente de la route et un éventuel dévers de la route et/ou roulis du véhicule. La prise en compte de ces paramètres nous permet d'éviter les fausses détections liées à un seuil de hauteur arbitraire d'identification des points au-dessus de la route. Cette section va donc tout d'abord présenter la création du profil longitudinal de la route utilisé pour la détection des obstacles. Une fois les obstacles détectés, nous verrons leur caractérisation par des attributs simples comme la largeur, la hauteur et la profondeur.

3.2.1 Détection de la route

Pour les applications sur véhicule intelligent, il est fondamental d'éviter les fausses détections, en particulier dues aux obstacles détectés en bord de route (panneaux, arbres, glissières ...). Pour ce faire, il faut limiter la détection des obstacles à la surface de la route, ou encore, à la voie de circulation. De nombreuses approches ont déjà été proposées pour détecter cette voie. Celle que nous avons développée et intégrée dans notre système est une approche image proposée par [Chapuis et al., 2000], qui exploite un modèle de la route dans l'image afin de regrouper les indices préalablement détectés et de remettre à jour le modèle dans l'image. La méthode est très rapide et robuste aux occultations. Quelques résultats sont présentés en figure 3.1 et une présentation plus détaillée est donnée en Annexe B.

3.2.2 Modélisation du profil longitudinal de route

La modélisation du profil longitudinal doit pouvoir prendre en compte toutes les variations possibles de géométrie d'une route, en incluant les obstacles. Dans ce contexte, chacune des lignes de l'image traitées conduira à un point du profil longitudinal. Quelle que soit la position



FIG. 3.1 – Robustesse aux occultations de la détection de voie (haut), aux virages (milieu) et aux sorties d'autoroute (bas).

du système stéréoscopique, l'axe Y liant les deux caméras est supposé parallèle à la route sous le véhicule. De cette manière, les paramètres de la route (roulis, dévers et hauteur du sol) sont calculés localement pour chacune des lignes. Cette méthode permet de modéliser n'importe quel type de route. La différence entre le roulis et le dévers se traduira par un angle α de roulis constant dans toutes les lignes i étudiées alors que le dévers de la route ajoutera un angle β_i variable selon les lignes : $\alpha + \beta_i = \text{angle apparent}$, $i \in \text{lignes image}$. C'est une différence par rapport à la *v-disparité* qui calcule un angle unique de dévers.

Cas simple, sans dévers ni roulis

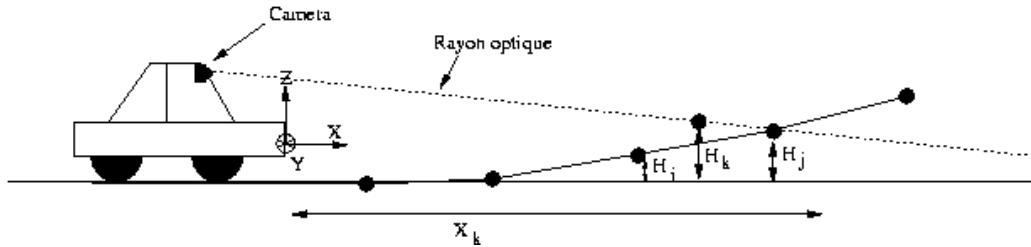


FIG. 3.2 – Profil longitudinal de route en ligne droite

Sans tenir compte du roulis/dévers, l'objectif est de construire un profil longitudinal de la voie de circulation comme montré en figure 3.2. La base de l'algorithme réside dans l'étude des lignes de l'image, une à une et limitée à la voie de circulation. En absence d'obstacle, de dévers et de roulis, tous les points 3D notés P_i^{3D} issus de la reconstruction de cette ligne i (suite à la sté-

réocorrélation) correspondent à des points de la surface de la route, tous à une même profondeur X et hauteur Z par rapport au repère véhicule. Une visualisation de la reconstruction de ces points, dans un cas théorique, est présentée en figure 3.3. Deux histogrammes construits à partir des coordonnées de ces points dans le repère véhicule, permettent la construction du profil longitudinal de route : (1) un premier histogramme selon la hauteur des points fait ressortir les objets, que ce soit la route elle-même et/ou un obstacle et (2) pour chacun de ces pics, un deuxième histogramme est effectué selon la profondeur de manière à associer un couple $(X_k, Z_k)_i$ que nous appellerons point d'intérêt p_k de notre ligne image i . Ainsi, les coordonnées de ce point d'intérêt sont reportées dans le profil longitudinal de route. Pour chacun de ces histogrammes, la taille de discrétisation (dimension des *bins*) est basée sur l'écart type de la profondeur (histogramme en X) et de la hauteur (histogramme en Z) donnée par le modèle d'erreur lié au banc stéréoscopique. C'est à dire que pour chacune des lignes étudiées, le nombre de *bins* varie en fonction des données 3D présentes sur la ligne : une ligne proche du véhicule aura un nombre de *bins* plus grand qu'une ligne éloignée dont l'écart type sera important. L'erreur de la stéréocorrélation est donc gérée grâce à ces dimensions variables, permettant de s'affranchir du problème du bruit pour des points fortement éloignés du véhicule.

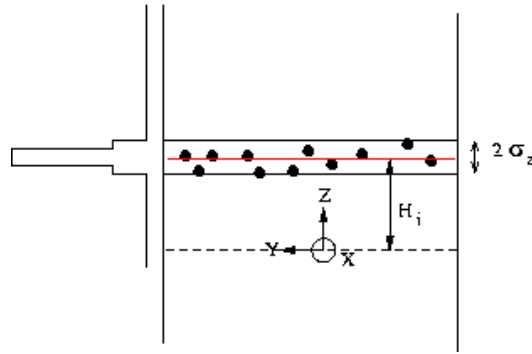


FIG. 3.3 – Etude d'une ligne image de la voie de circulation (ligne conduisant au point i de la figure 3.2) : reconstruction 3D et histogramme selon la hauteur des points. Les points 3D de la route horizontale (par rapport au système stéréo) se projettent sur l'axe Z selon une gaussienne de moyenne la hauteur de la route et d'écart type égal au σ_Z du modèle d'erreur du banc stéréoscopique (σ_Z constant si pas de dévers).

Le cas théorique d'une ligne en présence d'un obstacle est montré en figure 3.4. L'histogramme peut présenter plusieurs points d'intérêt dans le cas où la surface de la route et l'obstacle sont suffisamment visibles sur la ligne. Dans ce cas, la génération des deux points d'intérêt liés aux deux pics des histogrammes conduisent à deux points dans le profil longitudinal de la route. Dans ce cas, la dimension des *bins* utilisés pour la discrétisation des histogrammes ne doit pas être constante vue l'hétérogénéité des distances et hauteurs liées aux hauteurs et distances différentes. Cette dimension variable permet de garder une cohérence des histogrammes vis-à-vis des écarts types du modèle d'erreur.

Prise en compte du dévers

Le cas d'une ligne présentant du dévers et/ou roulis est présenté en figure 3.5. Dans la présentation de la *v-disparité* de [Labayrade, 2004], il est montré que le dévers et le roulis ne sont pas observables indépendamment l'un de l'autre. Dans un abus de langage nous utiliserons donc par

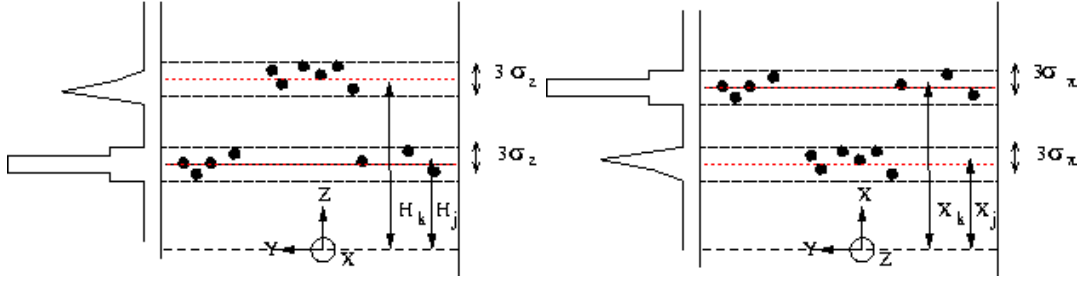


FIG. 3.4 – Etude d’une ligne image de la voie de circulation (ligne conduisant aux points j et k de la figure 3.2) en présence d’un obstacle : reconstruction 3D et histogramme selon la hauteur Z [gauche] et la distance X [droite] des points. Deux pics sont présents : l’un pour l’obstacle et l’autre pour la route.

la suite le terme de *dévers* pour parler de l’effet du dévers et/ou du roulis. L’histogramme d’une route avec un angle de dévers non nul ne présente plus un pic dont la variance se rapproche de celle intrinsèque à notre système stéréo σ_Z mais se rapproche d’une distribution uniforme de variance σ_u . En effet, la route ne se projetant plus en un unique point de l’axe Z ($\pm\sigma_Z$), celle-ci apparaît avec une épaisseur apparente. La détection du dévers peut donc se baser sur la comparaison de ces deux variances σ_Z et σ_u . Par interpolation linéaire des points 3D projetés dans le plan YZ , il est possible de calculer l’angle α d’inclinaison du dévers. Par correction des points P_i^{3D} selon cet angle, nous retrouvons le cas précédent sans dévers rendant possible l’extraction des points d’intérêt de la ligne étudiée. Cette méthode de correction de dévers permet donc la prise en compte systématique du dévers ligne par ligne et quel que soit ce dévers. C’est un avantage certain sur la *v-disparité* qui extrait un paramètre de dévers par profil longitudinal de route. Il devient donc possible de corriger n’importe quelle géométrie de route.

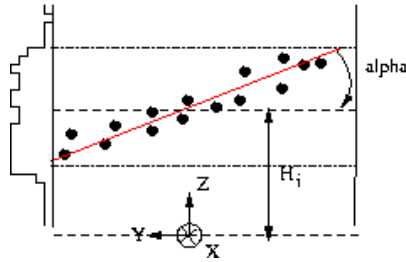


FIG. 3.5 – Etude d’une ligne image de la voie de circulation présentant un dévers et/ou du roulis : reconstruction 3D et histogramme selon la hauteur des points. La projection des points de la route avec dévers ne se projète plus en un unique point sur l’axe $Z \pm \sigma_Z$. L’épaisseur "apparente" de la route est directement liée à l’inclinaison entre la route et l’axe reliant les deux caméras.

La prise en compte des pics dans l’histogramme en Z est très importante ici. En effet, une seule interpolation linéaire sur des points correspondant à la route et à un obstacle (comme en figure 3.7) n’est pas satisfaisante puisque l’information à interpoler est hétérogène. Par contre, par séparation des points correspondant à des pics distincts avec l’utilisation de méthodes morphomathématiques, nous pouvons calculer des interpolations différentes, sources de plusieurs points d’intérêt du profil de route.

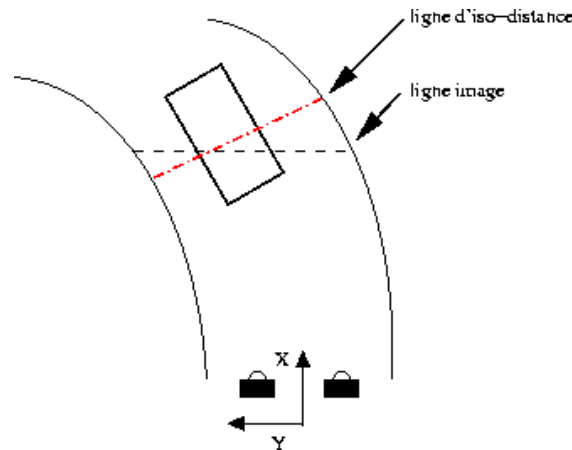


FIG. 3.6 – Une ligne de l'image correspond à une même distance Z de la route seulement quand celle-ci est en ligne droite.

Cette approche de correction de dévers permet aussi d'obtenir les lignes d'iso-distances par calcul de l'angle de lacet de la reconstruction 3D de la ligne image de la route. Les interpolations linéaires ne seraient donc plus dans le plan YZ mais dans XY . Après regroupement des points ayant la même distance, nous pouvons alors corriger l'effet selon lequel une ligne image ne correspond pas forcément à une ligne iso-distante, par exemple dans un virage prononcé présenté en figure 3.6. L'effet serait une amélioration du profil de route comme pour la correction du dévers. Malheureusement, la quantité de mémoire nécessaire et le temps de calcul supplémentaire ne vaut pas le peu de précision gagné.

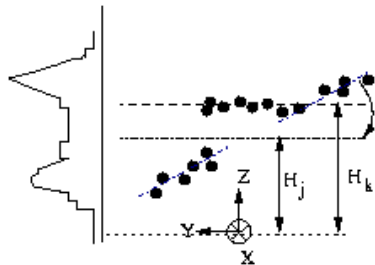


FIG. 3.7 – Etude d'une ligne image de la voie de circulation présentant un dévers et un obstacle : reconstruction 3D et histogramme selon la hauteur des points. Un obstacle visible par un plan vertical apparaît comme parallèle au repère caméra vu que la *coupe* de l'obstacle est réalisée selon Y , axe liant les deux caméras.

Bilan de la création du profil longitudinal

La modélisation du profil de route ainsi décrite permet de généraliser un maximum de situations possibles. La correction ligne à ligne du dévers/roulis permet d'obtenir le même profil si le roulis est constant sur chaque ligne, ou si le dévers est variable (angle nul pour une ligne proche du véhicule si Y parallèle à la route et angle de plus en plus élevé si le virage est de plus en plus prononcé). Les méthodes d'extraction des obstacles peuvent donc s'affranchir des paramètres de la route pour détecter les obstacles au sein du profil.

3.2.3 Extraction des obstacles dans le profil longitudinal

Un profil vertical d'autoroute habituel dépasse rarement les 15%, soit une inclinaison de 8.5° par rapport à l'horizontal. En approximant, dans un premier temps, la route par des morceaux de pente, il est donc possible de trouver les zones correspondant à un profil de route *anormal*. Le calcul, par morceau (*Line Fitting*), de la pente de la route sert à détecter une rupture anormale de pente attribuée à un obstacle (Cf. figure 3.8 [gauche]). Rappelons que chacun des points du profil de route est issu des pics obtenus sur des lignes différentes de l'image et donc aussi de l'obstacle. L'algorithme de détection d'obstacles revient donc en une suite d'interpolations linéaires, rendues robustes par une approche RANSAC, sur des intervalles successifs de profondeur. Lorsqu'une pente est trop importante, l'intervalle en question est libellé comme *intervalle obstacle*. La longueur en X des interpolations linéaires est dépendante de l'écart type sur la distance d'un point du modèle d'erreur de la stéréo. De cette manière, la route proche du véhicule sera interpolée par une droite dont la dimension sera plus faible qu'une portion de route plus éloignée. Ceci afin de prendre en compte le fait que la stéréovision donnera des informations beaucoup plus denses et précises à proximité du véhicule que pour une longue distance (par exemple, 30 m).

A ce stade de l'extraction des obstacles, la méthode est équivalente à la *v-disparité* qui calcule des *cassures* artificielles de la route à l'aide d'une transformée de *Hough* adaptée. Ces cassures ne sont pas représentatives de vraies informations géométriques de la route et pourraient gêner la généralité de la méthode. A partir d'une détection d'un *intervalle obstacle* de notre méthode, un lissage par une quadrique (Cf. figure 3.8 [droite]) est appliqué sur les intervalles précédents considérés comme étant *intervalle route*. Le calcul de la surface de la route tend donc à se rapprocher d'une géométrie plus naturelle d'une route. Cette quadrique donne la valeur de la hauteur du sol pour cet intervalle obstacle. Un seuil de hauteur égal à $3\sigma_Z$ du modèle d'erreur permet alors de discriminer les points obstacles de cet intervalle des points de la route. Une fois les points obstacles détectés, la moyenne des distances de ces points est attribuée à la distance de l'obstacle. L'utilisation de la quadrique se justifie par le fait que l'on rencontre difficilement plus de deux ruptures de pente au sein d'une même image.

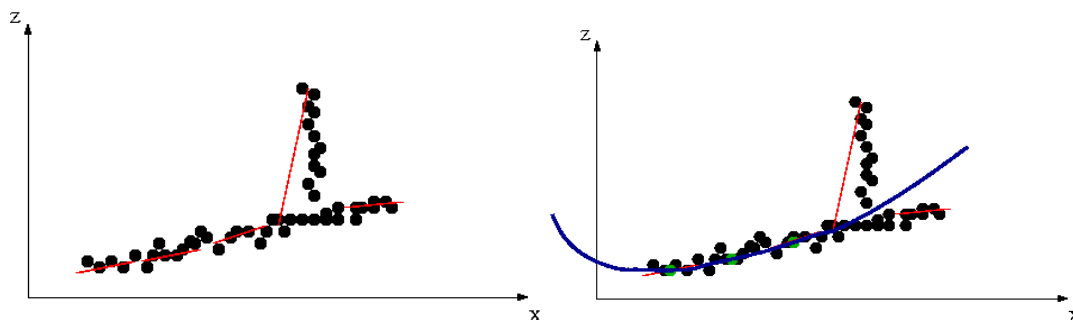


FIG. 3.8 – Interpolation linéaire par morceau du profil longitudinal de la route[gauche] et lissage de la surface de la route par une quadrique[droite].

3.2.4 Résultats

Sans roulis ni dévers

La figure 3.9 représente les histogrammes sur les profondeurs [gauche] et sur la hauteur [droite] d'une ligne image de la route de la figure 3.11. Lorsque aucun paramètre de dévers n'est

présent dans l'image, les histogrammes sont extrêmement discriminants sur les distances et sur les hauteurs des points d'intérêt à reporter dans le profil de route. La figure 3.10 présente ce dernier après calcul des histogrammes de toutes les lignes images. La détection de l'obstacle est très simple et est visible en figure 3.11 par la ligne rouge en bas du véhicule.

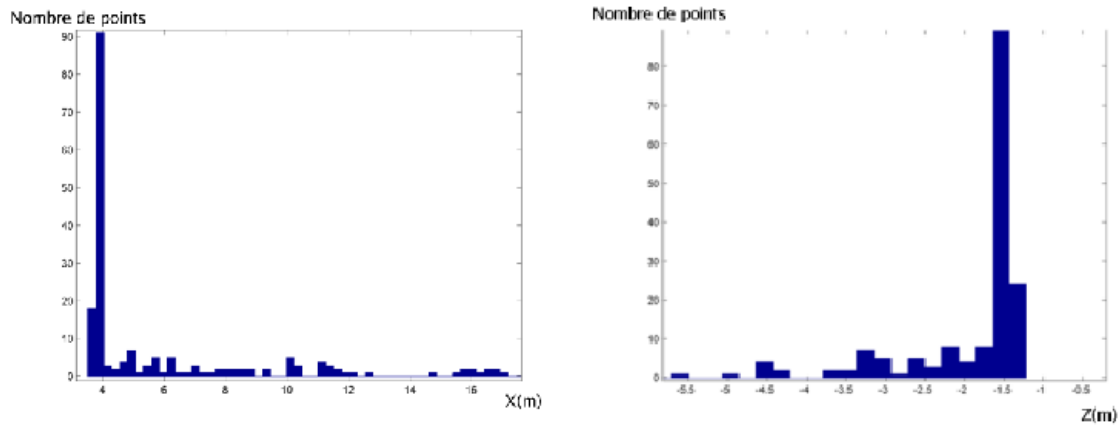


FIG. 3.9 – Histogrammes selon la profondeur et la hauteur d'une ligne image de la voie de circulation. La ligne traitée correspond à une distance de $4m$ à l'avant du véhicule et à une hauteur de $1.5m$, hauteur des caméras par rapport au sol.

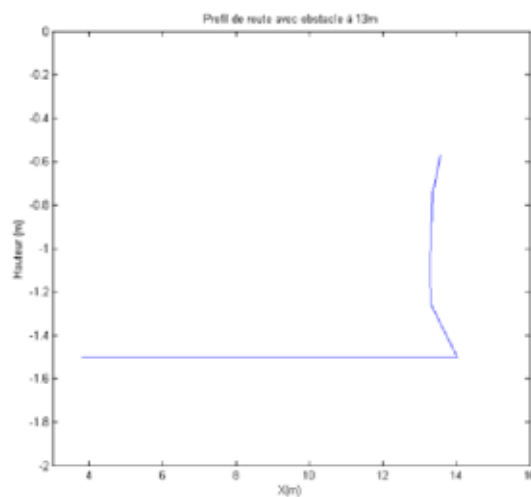


FIG. 3.10 – Résultat du profil longitudinal de route sur image de synthèse. La route est détectée à $1.5m$ en dessous des caméras. L'obstacle est présent à $13m$.

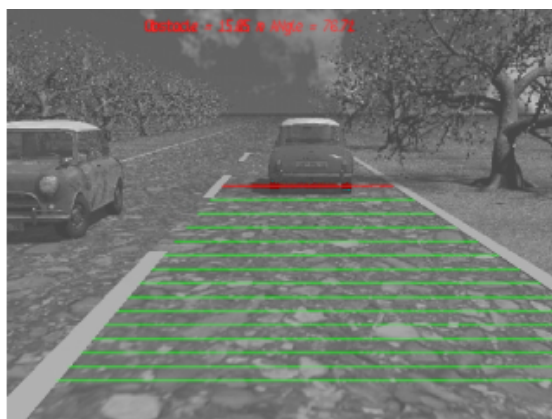


FIG. 3.11 – Résultat de la détection d'obstacles sur image de synthèse. Les lignes vertes correspondent à des lignes interprétées comme étant sur la route. La ligne rouge est détectée comme étant la base de l'obstacle.

Variation de pente

Contrairement au cas précédent, la paire d'images stéréoscopiques de la figure 3.12 montre une rupture de pente importante à l'avant du véhicule. Il est donc important de prendre en compte cette (ou ces) inclinaison(s) de la route par rapport au capteur afin de ne pas déclencher de fausses détections. La figure 3.13 nous donne la reconstruction 3D d'une ligne image de la route. Son histogramme en Z correspondant en figure 3.14 ne laisse aucun doute quant à la hauteur de cette ligne. L'ensemble des informations données par les lignes est fusionné dans le profil en figure 3.15. Les lignes rouges représentent les estimations linéaires locales du plan de la route. L'approximation par une quadrique permet une plus grande stabilité de l'extraction des obstacles avec un seuillage plus fin de leur hauteur.

Le traitement est totalement équivalent à la détection d'obstacles sur route plane. Aucun prétraitements n'est requis, la méthode prenant en compte intrinsèquement le problème des variations de pente. Le profil longitudinal ainsi créé est équivalent à la représentation *v-disparité* donnée en figure 3.16. L'extraction des plans du sol est réalisée par des transformées de Hough successives générant des cassures dans le profil de route. Dans ce cas, deux droites sont extraites pour chacune des portions de route d'inclinaison différente. L'intérêt de la *v-disparité*, qui n'existe pas dans notre méthode, est la détection de la ligne d'horizon donnée par l'intersection de la droite de la route avec l'axe des lignes de l'image.

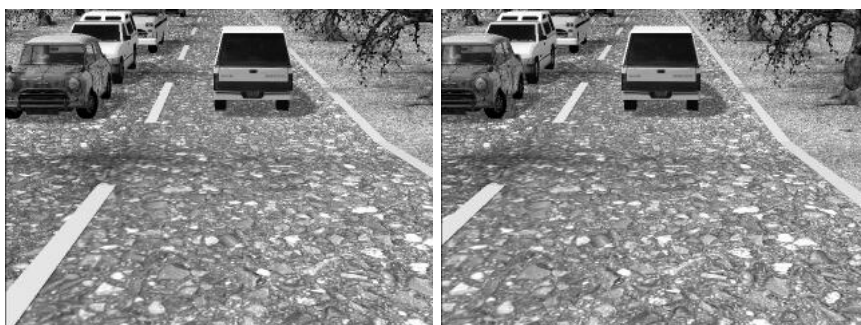


FIG. 3.12 – Paire stéréo sur route avec une variation de pente à l'avant du véhicule

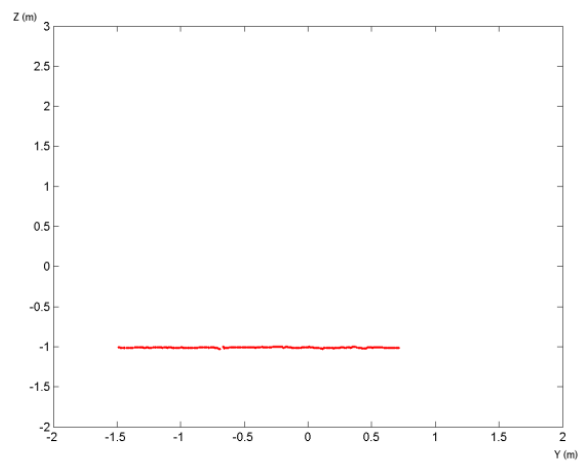


FIG. 3.13 – Reconstruction d'une ligne image de la route. La hauteur de la route est détectée à $1m$ en dessous des caméras.

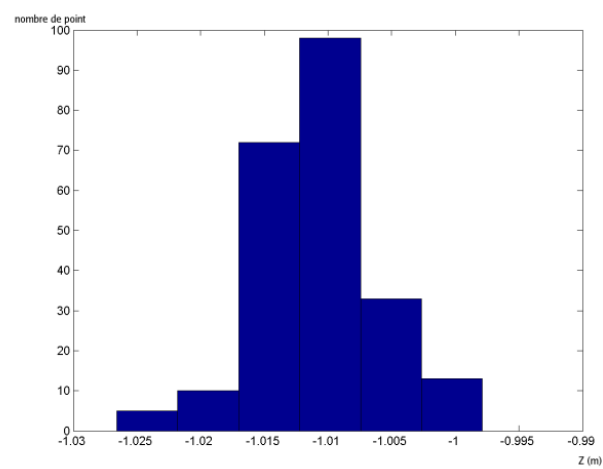


FIG. 3.14 – Histogramme en Z d'une reconstruction d'une ligne image de la route. Cet histogramme représente une ligne à une hauteur de $1m$ en dessous des caméras.

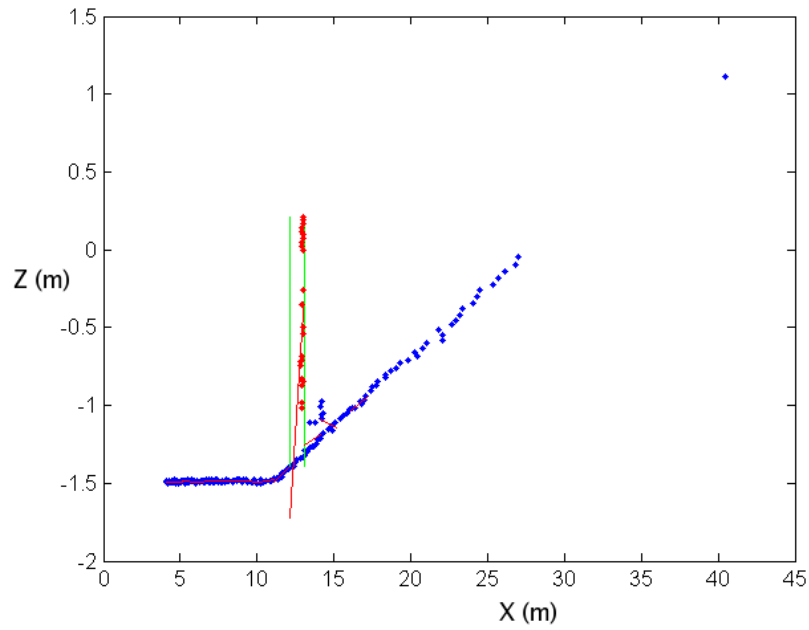


FIG. 3.15 – Profil longitudinal de la route. Une cassure de pente est visible à 10m. L'obstacle est détecté à 12.90m sans fausse détection liée à la variation de pente.

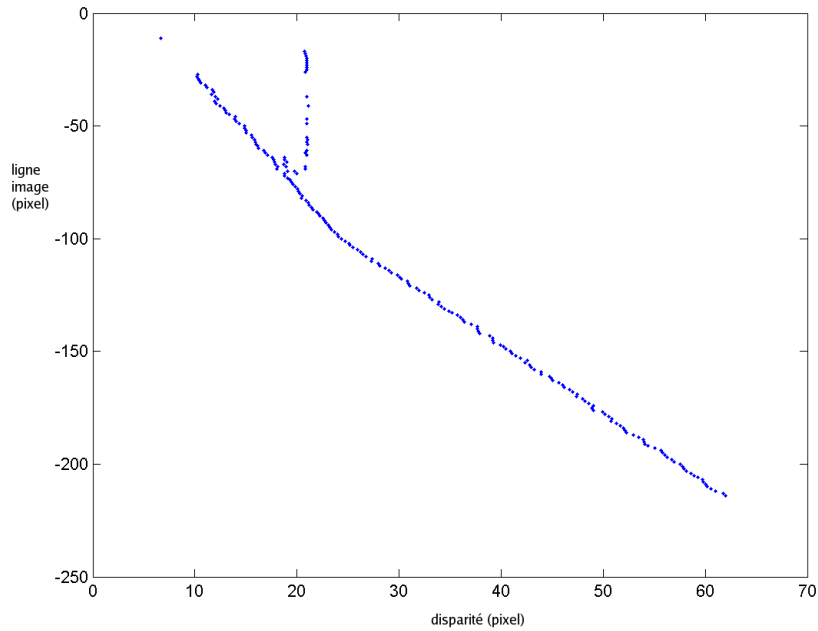


FIG. 3.16 – Représentation v -disparité. Une cassure de pente est visible au niveau de la ligne 100. L'obstacle est apparent par les points alignés verticalement.

Roulis

Nous allons voir dans cet exemple l'effet de roulis constant de 2° présent dans la paire stéréo de la figure 3.17. Son influence est visible dans la reconstruction $3D$ d'une ligne image de la route (cf. figure 3.18[gauche]) et l'est encore plus dans l'histogramme de hauteur présenté en figure 3.19[gauche]. Ce dévers n'est pas directement problématique pour la détection d'obstacles mais, comme nous l'apercevons sur le profil de route de la figure 3.20[haut], perturbe les estimations locales des pentes de la route vu l'épaisseur apparente de celle-ci.

En corrigeant l'angle de dévers directement au niveau de la reconstruction $3D$ (cf. figure 3.18[droite]), l'histogramme sur Z gagne un facteur $\lambda = 3$ sur la variance du pic (cf. figure 3.19[droite]), effet notable dans le profil de route (cf. figure 3.20). Remarquons le gain apporté par la correction du roulis sur la ligne avec l'intervalle sur Z allant de $[-1.58..-1.42]$ sans correction à $[-1.54..-1.50]$ avec la correction. Sans la correction du roulis dans le profil de route, les interpolations linéaires seraient fortement bruitées et généreraient des fausses détections d'intervalle obstacle.

A partir de ce profil de route corrigé du dévers et prenant en compte, par construction, les variations de pente, il est facile d'extraire l'obstacle et les points obstacles comme le montre la figure 3.21.

La figure 3.22 donne la représentation du profil de route dans l'espace de v -disparité. L'extraction du dévers est réalisée en calculant "l'épaisseur" de la route à travers une transformée de *Hough* adaptée. Cette extraction semble moins facile à réaliser que la correction des points $3D$ avant même le calcul du profil. Surtout, la méthode semble inadaptée si un angle de dévers non constant dans toutes les lignes de l'image apparaît et s'ajoute à cet angle de roulis.

La simulation du dévers est trop difficile à générer vu la variation progressive de l'angle α . Seule une étude sur images réelles permet d'évaluer le système dans le cas de dévers.



FIG. 3.17 – Paire stéréo sur route en dévers

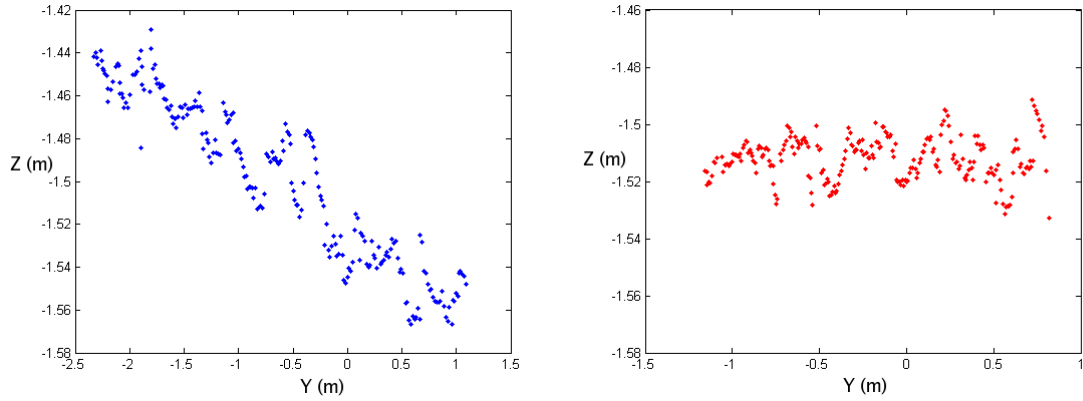


FIG. 3.18 – Reconstruction d’une ligne image de la route sans correction de dévers [gauche], et avec correction [droite]. La ligne non corrigée présente une inclinaison de 1.81° , angle non visible après correction. Remarquons l’intervalle sur Z qui passe de $[-1.58.. -1.42]$ à $[-1.54.. -1.50]$ relevant le gain apporté par la correction du roulis.

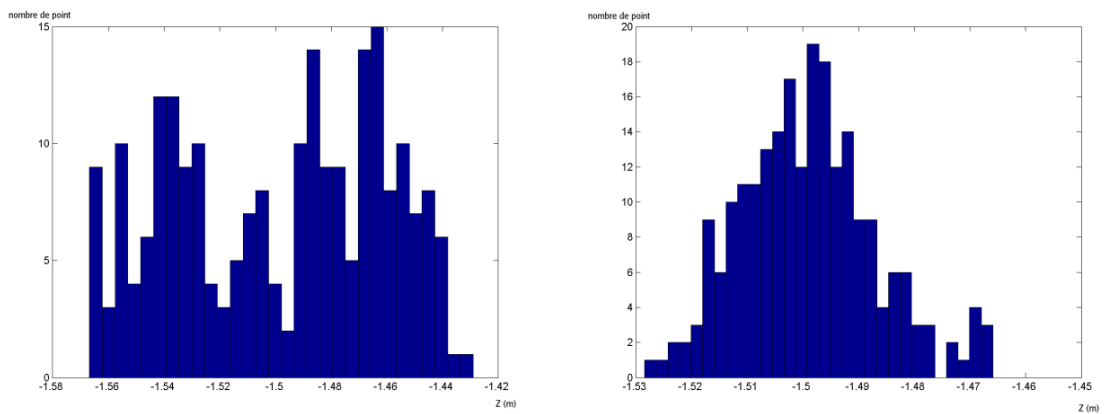


FIG. 3.19 – Histogramme en Z d’une reconstruction d’une ligne image de la route sans correction du dévers ($\sigma_Z = 0.037\text{m}$)[gauche], et avec correction ($\sigma_Z = 0.012\text{m}$)[droite]. La correction du dévers rend l’extraction de l’information de hauteur beaucoup plus pertinente (beaucoup plus de points dans les *bins* de l’histogramme) que sans correction.

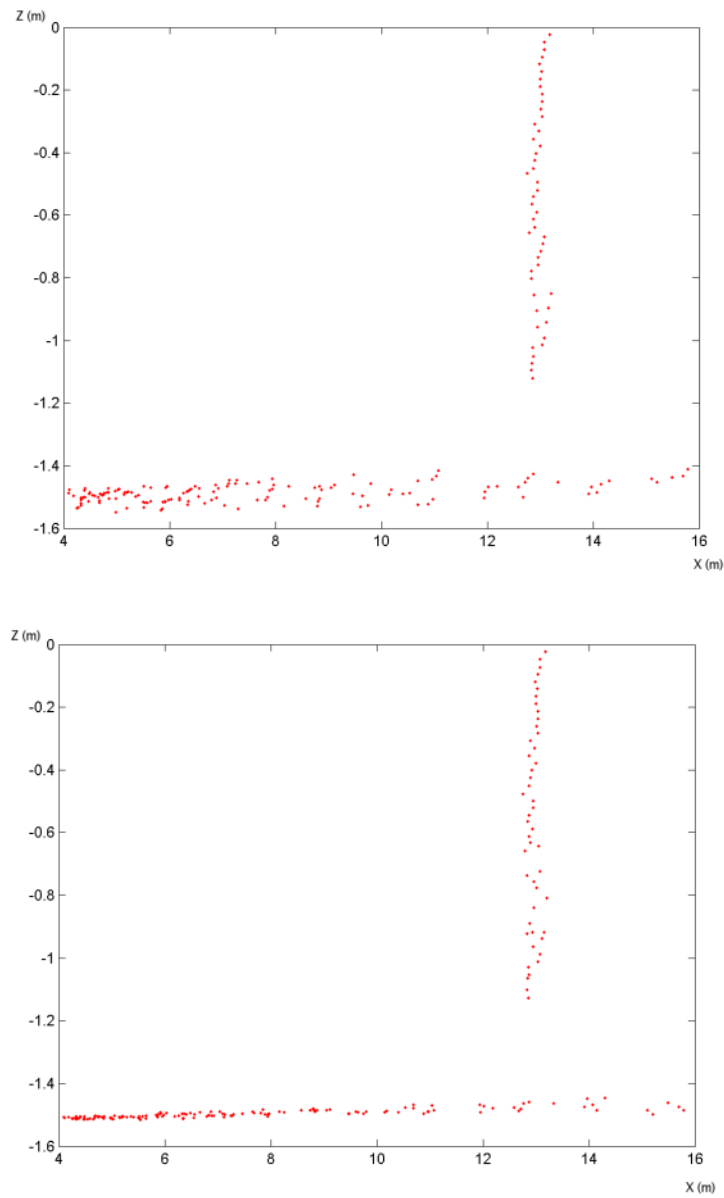


FIG. 3.20 – Profil longitudinal de la route sans correction du dévers [haut], et avec correction [bas]

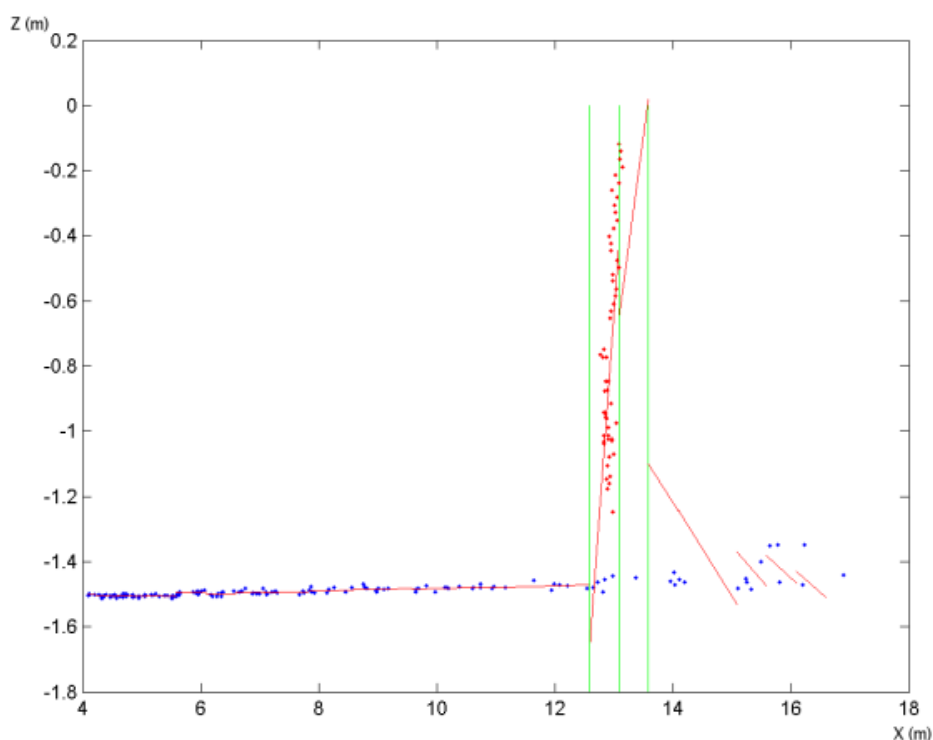


FIG. 3.21 – Estimation de la route et extraction des obstacles. L'obstacle est détecté à $12.95m$ du véhicule.

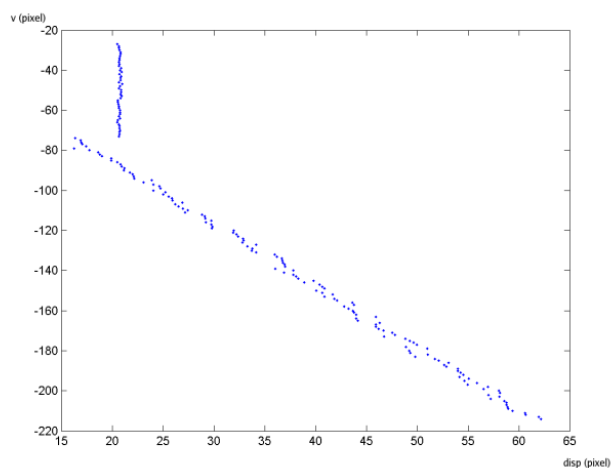


FIG. 3.22 – Représentation *v-disparité*

Cas général

Ce paragraphe présente un cas de dévers et de variation de pente non nuls (cf. figure 3.23[haut]) afin de tester la robustesse de notre démarche. Sur les 7.5 premiers mètres à l'avant du véhicule, un dévers de 2° est présent. A partir de $7.5m$, un dévers de -2° est présent ainsi

qu'une pente de 5° . La figure 3.23[milieu] donne une reconstruction 3D d'une ligne non corrigée [gauche] et corrigée [droite] du dévers. La figure 3.24 montre l'intérêt de la correction du dévers pour avoir une extraction pertinente du profil de route et des obstacles. Les figures 3.25 et 3.26 montrent les profils de route dans l'espace de *v-disparité* et dans l'espace 3D. Ce dernier nous permet une extraction de l'obstacle sans présence de fausses détections, le dévers étant corrigé (surface nette de la route) et les pentes de la route étant extraites.

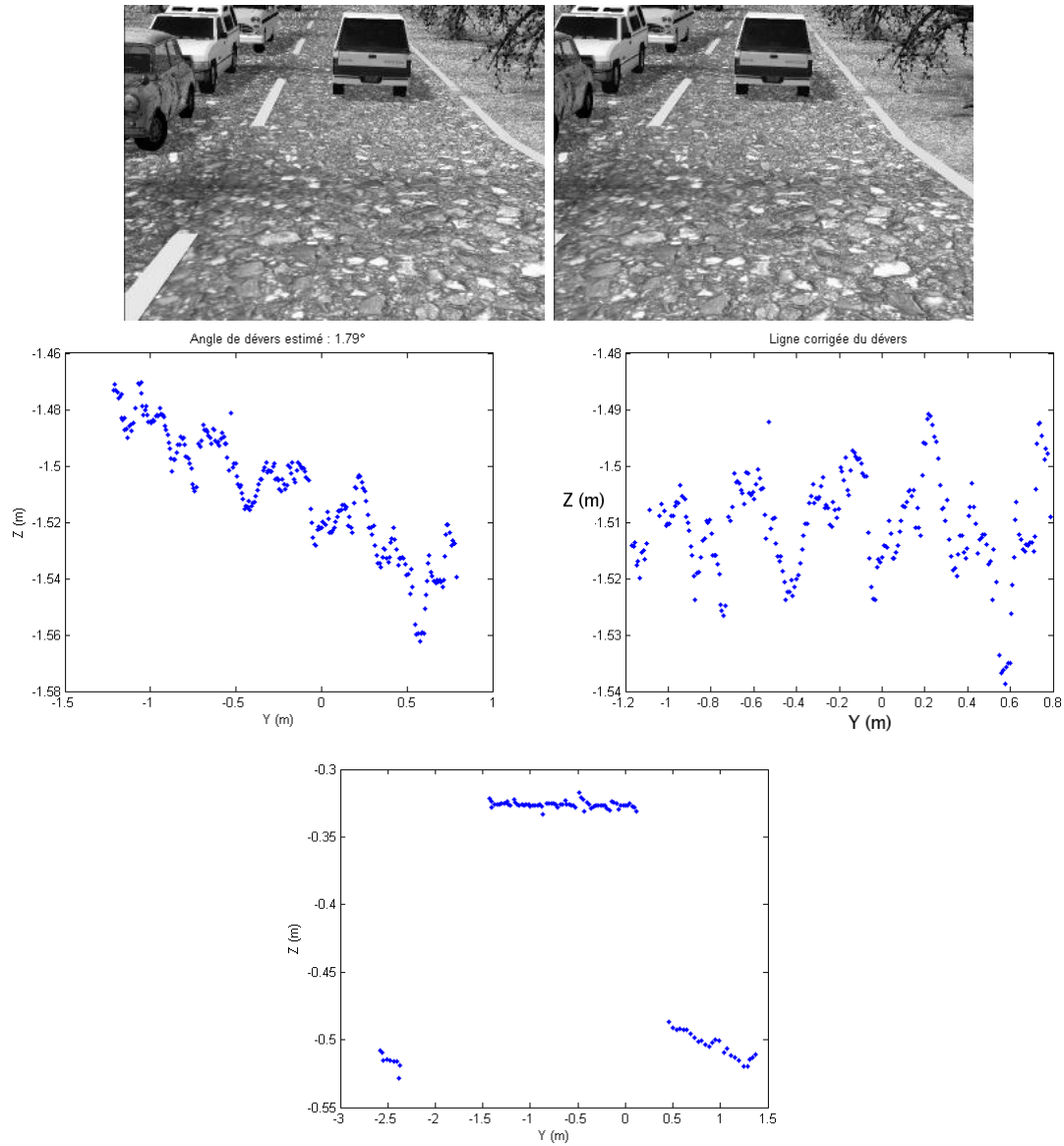


FIG. 3.23 – Paire stéréo sur route en dévers avec variation de pente [haut], reconstruction d'une ligne image de la route sans correction de dévers $\sigma_Z = 0.0208m$ [milieu-gauche], et avec correction $\sigma_Z = 0.0088m$ [milieu-droite], reconstruction d'une ligne image avec présence d'un obstacle [bas].

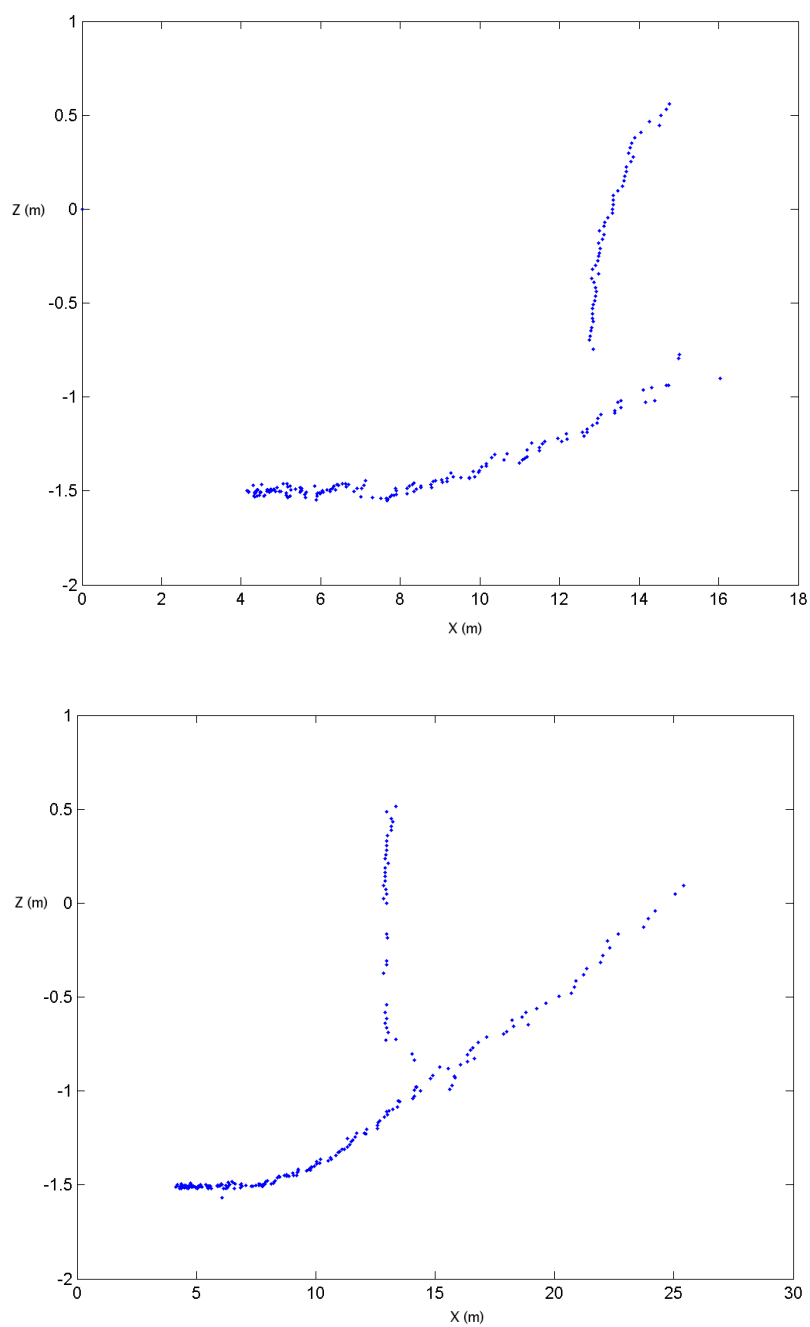


FIG. 3.24 – Profil de route sans correction [haut] et profil de route corrigé des dévers [bas]

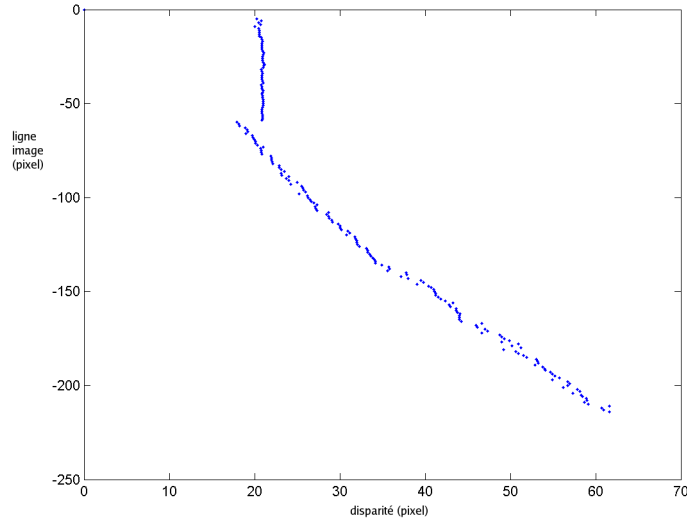


FIG. 3.25 – Représentation v -disparité

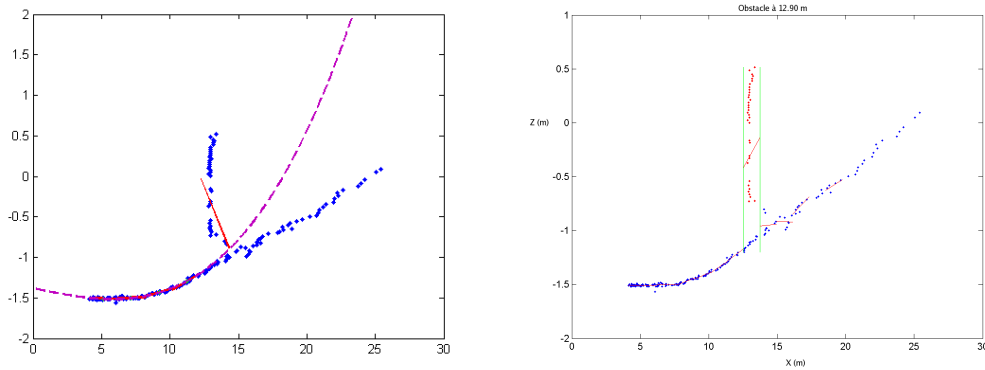


FIG. 3.26 – Lissage de la surface de la route par une quadrique [gauche] et estimation de la route avec extraction des obstacles [droite]

3.3 Caractérisation des obstacles

Une fois les obstacles détectés, nous devons les caractériser par quelques attributs simples comme la largeur, la hauteur, la profondeur (quand celle-ci est disponible) et surtout sa distance au véhicule porteur du système de détection. Dans cette étude, seuls les paramètres statiques sont étudiés. Nous pouvons représenter les obstacles par des boîtes englobantes. Sur une route plane sans variation de pente, les obstacles ne sont observables que par leur arrière, ce qui rend la mesure de la profondeur impossible. Celle-ci n'est disponible que dans le cas d'approche d'une montée ou d'un virage prononcé.

Notre méthode de caractérisation est basée sur des projections successives dans les plans XZ , YZ et XY . Nous décrivons les différentes étapes de caractérisation à partir de l'exemple virtuel donné en figure 3.27 avec l'hypothèse que nous travaillons sur l'ensemble de la route et non

plus seulement sur la voie de circulation pour faciliter la visualisation sur les images. Un autre exemple pourrait être deux motocyclettes côte à côte sur notre voie. Ici, trois obstacles sont à détecter : deux voitures et un camion lointain en travers de la route. La méthode de détection d'obstacles décrite précédemment traite l'image ligne par ligne. Les résultats de ces traitements sont fusionnés de manière à séparer les obstacles les uns des autres et à en extraire leurs limites.

La caractérisation des obstacles apporte donc une seule étape supplémentaire à la détection d'obstacle précédemment décrite, chaque point du profil de route étant étudié dans le plan YZ pour extraire la largeur et la hauteur des obstacles.

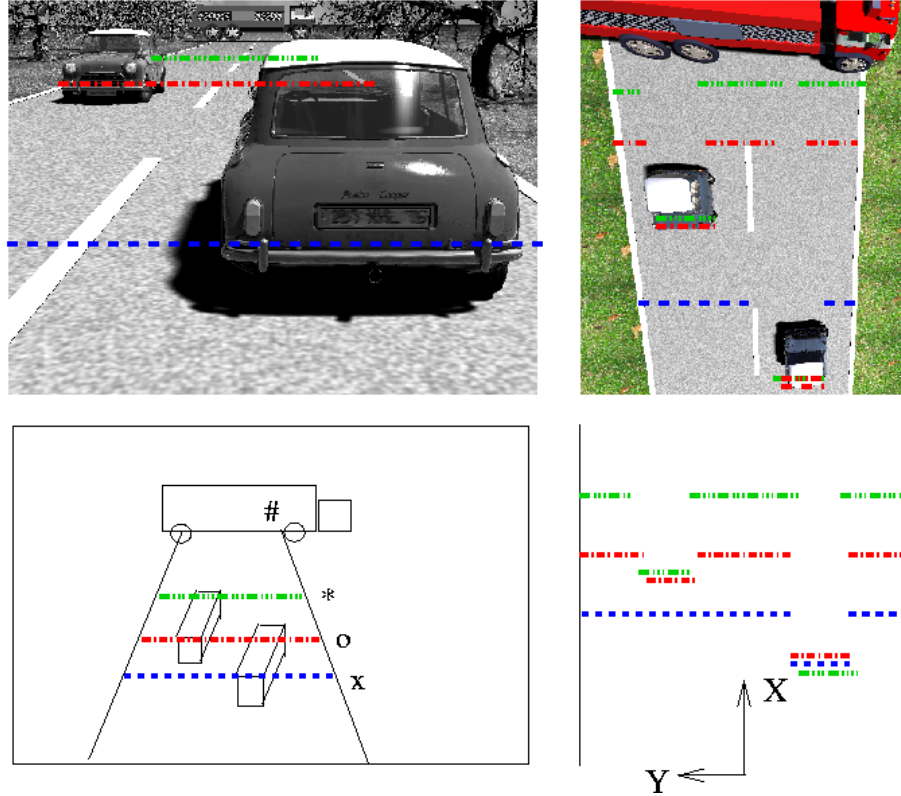


FIG. 3.27 – Images de synthèse (haut) et projection des points 3D dans le plan XY (bas) représentant une configuration possible de l'environnement.

3.3.1 Extraction des dimensions des obstacles

La figure 3.27[droite] montre comment les pixels sur les quatre lignes notés x , o , $*$ et $\#$ sont projetés sur le sol (ou, pour être précis, sur le plan supportant les quatre roues de la voiture portant le système). Quand un pixel d'une ligne est sur un obstacle, il se trouve à une distance différente d'un pixel de la même ligne représentant la route. Ce phénomène explique la répartition des points dans le profil de route présenté en figure 3.28[gauche]. Ces histogrammes permettent (1) de séparer les obstacles qui semblent être à une même distance et (2) d'estimer la profondeur de ces obstacles. Pour terminer la caractérisation, la figure 3.28[droite] montre comment toutes les informations amenées par les différentes lignes sont fusionnées sur le plan YZ . Cette dernière projection permet de (1) séparer les obstacles qui sont vraiment à une même distance de la caméra et (2) d'estimer leur largeur.

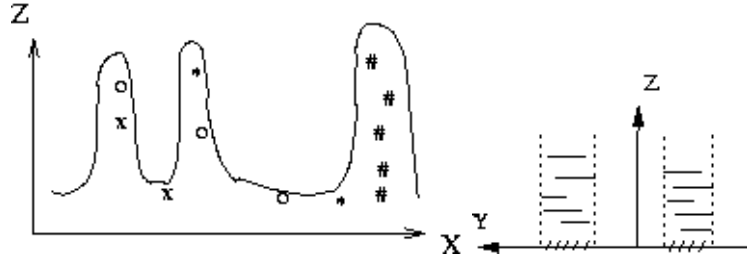


FIG. 3.28 – Profil de route [gauche], projection des points P_i^{3D} sur le plan YZ [droite].

3.3.2 Résultats

Prenons un exemple simple comme le profil de route du cas général en figure 3.26. L'extraction des obstacles a étiqueté les points obstacles (points rouge sur la figure) dans l'intervalle appelé *intervalle obstacle*. La hauteur de l'obstacle est simplement calculée avec le minimum et le maximum des hauteurs des points d'intérêt obstacles. Par exemple, ici la hauteur de l'obstacle est de 1.30 cm. De plus, pour chacun des points d'intérêt, sont conservées les coordonnées Y_{min} et Y_{max} des points 3D ayant servi à sa construction (points issus d'un pic de l'histogramme en Z). De cette manière, la largeur des obstacles est déterminée par superposition de ces coordonnées (cf. figure 3.29 [droite]). Ici la largeur trouvée est de 1.54m pour une largeur effective de 1.5m. La boîte englobante image est le fruit d'un stockage des coordonnées des pixels lors des créations des profils de route. Ici, les roues du véhicule ne sont pas détectées dans le 3D (trou dans le profil entre l'obstacle et la route) ce qui explique que la boîte ne descend pas jusqu'à la route. Il est facile de pallier ce problème en calculant l'intersection entre le plan de l'arrière du véhicule avec le plan de la route.

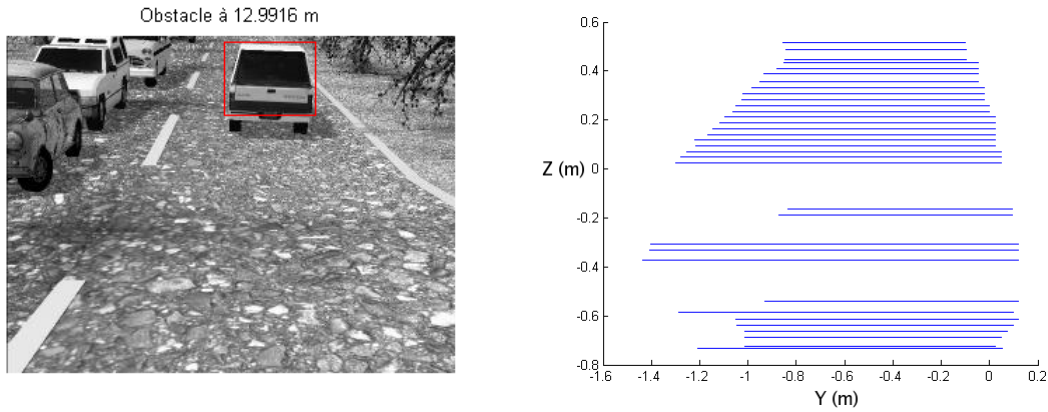


FIG. 3.29 – Boîte englobante image de l'obstacle détecté à 12.99m pour une distance réelle de 13m [gauche] et calcul de la largeur de l'obstacle [droite]

3.4 Conclusion

Nous avons présenté dans ce chapitre une approche générique de détection d'obstacles qui prend en compte les paramètres de dévers et/ou roulis et de variation de pente de la chaussée. Ces paramètres étant évalués ligne par ligne, l'approche est totalement générique à tout type de géométrie de route. Nous avons vu que même si le roulis et le dévers ne sont pas observables

séparemment, l'effet d'un dévers variable dans l'image n'est pas négligeable. Après étude entière de la voie de circulation, il serait alors possible d'extraire le paramètre unique de roulis afin de suivre dans le temps les paramètres du dévers par un filtre de Kalman Etendu. La fusion temporelle ainsi mise en place pourrait robustifier la création du profil longitudinal de route.

La détection de la limite de la voie de circulation est un inconvénient vis-à-vis de la *v-disparité* qui travaille sur l'ensemble de l'image et qui n'est donc pas dépendante d'une autre fonction de détection de voie. Cette limitation est amenée par le souci de conserver des performances proches du temps réel (15Hz pour le système actuel sur un *PIV 1.7GHz* sur des images 320×240 avec la stéréocorrélation limitée à la voie de circulation). Notre approche se base sur une carte de disparité dense qu'il serait difficile d'obtenir sur l'ensemble de l'image à une telle fréquence par un traitement logiciel. Le projet **PICASSO** d'implémentation FPGA d'une stéréo à 100Hz lèvera cette limitation.

Bibliographie du chapitre

- [ARCOS, 2002] ARCOS (2002). Etat de l'art en détection d'obstacles par perception. Rapport Technique R13, PREDIT.
- [Aufrere et al., 2000] R. Aufrere, F. Marmoiton, R. Chapuis, F. Collange, et J. Derutin (2000). Road detection and vehicles tracking by vision for acc. *Traitement du signal*.
- [Batavia et al., 2001] P. Batavia et S. Singh (2001). Obstacle detection using adaptive color segmentation and color stereo homography. Dans *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE.
- [Bertozzi et al., 1997] M. Bertozzi et A. Broggi (1997). GOLD : a parallel real-time stereo vision system for generic obstacle and lane detection. *IEEE Transactions on Image Processing*. In press.
- [Betke et al., 1998] M. Betke et H. Nguyen (1998). Highway scene analysis from a moving vehicle under reduced visibility conditions. Dans *IEEE Intelligent Vehicle*.
- [Chapuis et al., 2000] R. Chapuis, F. Marmoiton, et R. Aufrere (2000). Road detection and vehicles tracking by vision for an on-board acc system in the velac vehicle. Dans *3rd International Conference on Information Fusion*, volume 2.
- [Demonceaux et al., 2004] C. Demonceaux et D. Kachi-Akkouche (2004). Robust obstacle detection with monocular vision based on motion analysis. Dans *Intelligent Vehicles Symposium*.
- [Franke et al., 2000] U. Franke et A. Joos (2000). Real-time stereo vision for urban traffic scene understanding. Dans *IEEE Intelligent Vehicle Symposium, Dearborn, USA*.
- [Hancock, 1997] J. Hancock (1997). High-speed obstacle detection for automated highway applications.
- [Hancock, 1999] J. Hancock (1999). *Laser Intensity-Based Obstacle Detection and Tracking*. Thèse de doctorat, Carnegie Mellon university, The Robotics Institute, Pittsburg Pennsylvania.
- [Hu et al., 2000] Z. Hu et K. Uchimura (2000). Tracking cycle : A new concept for simultaneously tracking of multiple moving objects in a typical traffic scene. Dans *IEEE Intelligent Vehicles Symposium*.
- [Jiang et al., 2002] G. Jiang, M. Yu, S. HE, S. Shi, et X. LIU (2002). A new method for obstacle detection in its. Dans *ICSP*.
- [Jouannin, 1999] S. Jouannin (1999). *Association et fusion de données : application au suivi et à la localisation d'obstacles par radar à bord d'un véhicule routier intelligent*. Thèse de doctorat, Université Blaise Pascal, Clermont Ferrand.
- [Kalinke et al., 1998] T. Kalinke, C. Tzomakas, et W. Seelen (1998). A texture-based object detection and an adaptive model-based classification. *Mustererkennung*, pages 627–634.

- [Kato et al., 2001] T. Kato, T. Tanizaki, T. Ishhi, H. Tanaka, et Y. Takimoto (2001). 76 ghz hight performance radar sensor featuring fine step scanning machanism utilizing nrd technology. Dans *IEEE Intelligent Vehicle*.
- [Koller et al., 1994] D. Koller, T. Luong, et J. Malik (1994). Binocular stereopsis and lane marker flow for vehicle navigation :lateral and longitudinal control. Rapport Technique UCB/CSD-94-804, LAAS.
- [Kuroda et al., 1998] H. Kuroda, S. Kuragaki, T. Minowa, et K. Nakamura (1998). An adaptative cruise control system using a millimeter wave radar. Dans *IEEE Intelligent Vehicle*.
- [Labayrade, 2004] R. Labayrade (2004). *Détection générique, robuste et rapide d'obstacles routiers par stéréovision embarquée*. Thèse de doctorat, Université Pierre et Marie Curie Paris VI Jussieu.
- [Labayrade et al., 2002] R. Labayrade, D. Aubert, et J. Tarel (2002). Real time obstacle detection in stereo vision on non flatroad geometry through v-disparity representation. Dans *IV2002*.
- [Langer, 1997] D. Langer (1997). *An integrated MMW radar system for outdoor navigation*. Thèse de doctorat, Canegie Mellon University.
- [Leeuwen et al., 2001] M. V. Leeuwen et F. Groen (2001). A platform for robust real-time vehicle tracking using decomposed templates. Rapport technique, University of Amsterdam.
- [Matthews et al., 1996] N. D. Matthews, P. E. An, et C. J. Harris (1996). Vehicle detection and recognition for autonomous intelligent cruise control. Rapport technique, University of Southampton.
- [Murphey et al., 2000] Y. Murphey, J. Chen, J. Crossman, J. Zhang, P. Richardson, et L. Sieh (2000). Depthfinder, a real-time depth detection system for aided driving. Dans *IEEE Intelligent Vehicles Symposium*.
- [Nashashibi et al., 1993] F. Nashashibi et M. Devy (1993). 3d incremental modeling and robot localization in structured environment using a laser range finder. Dans *International Conference on Robotics and Automation*, pages 20–27, Atlanta. IEEE.
- [Papageorgiou et al., 1998] C. P. Papageorgiou, M. Oren, et T. Poggio (1998). A general framework for object detection. Dans *ICCV '98 : Proceedings of the Sixth International Conference on Computer Vision*, page 555, Washington, DC, USA. IEEE Computer Society.
- [Steux, 2001] B. Steux (2001). *RT maps, un environnement logiciel dédié à la conception d'applications embarqués temps-réel. Utilisation pour la détection automatique de véhicules par fusion radar/vision*. Thèse de doctorat, Ecole des mines de Paris.
- [Trassoudaine, 1993] L. Trassoudaine (1993). *Solution multisensorielle temps réel pour la détection d'obstacles sur route*. Thèse de doctorat, Université Blaise Pascal, Clermont Ferrand.
- [Weingarten et al., 2004] J. Weingarten, G. Gruener, et R. Siegwart (2004). A state-of-the-art 3d sensor for robot. Dans *IEEE International Conference on Intelligent Robots and Systems (IROS'2004)*.
- [Williamson, 1998] T. Williamson (1998). *A high-performance stereo vision system for obstacle detection*. Thèse de doctorat, Carnegie Mellon University, The Robotics Institute, Pittsburg, Pennsylvania.
- [Xie et al., 1994] M. Xie, L. Trassoudaine, J. Alizon, et J. Gallice (1994). Road obstacle detection and tracking by an active and intelligent strategy. *Machine and vision application*, pages 165–177.

Chapitre 4

Modélisation de la place de parking

Sommaire

4.1	Description du problème	94
4.1.1	Repères et définitions	94
4.1.2	Spécifications	95
4.2	Etat de l'art	96
4.3	Evaluation des techniques issues de la Robotique	97
4.3.1	Extraction du plan du sol	98
4.3.2	Recalage des données entre plusieurs scènes	100
	ICP (Iterative Closest Point)	101
4.3.3	Conclusion	101
4.4	Détection des composantes de l'environnement	104
4.4.1	Extraction des obstacles	104
4.4.2	Détection de trottoirs	105
4.5	Modélisation de la place de parking	107
4.5.1	Modèle local	107
	Calcul de la boîte englobante d'un obstacle	108
	Calcul de la boîte englobante totale d'un obstacle	109
4.5.2	Modèle global	110
	Recalage des données issues de différentes perceptions	111
	Gestion de la fenêtre glissante	112
	Construction du modèle fusionné	113
	Affinement de la boîte englobante décrivant un obstacle	115
4.5.3	Mesure de la place libre	118
4.6	Résultats	118
4.6.1	Capteur <i>idéal</i> (selon le constructeur du système)	119
	Présentation des caractéristiques du banc de synthèse	119
	Résultats de modélisation de la voie	120
4.6.2	Capteur <i>idéal</i> (selon le développeur)	124
	Présentation des caractéristiques du banc de synthèse	124
	Résultats de modélisation de la voie	124
4.7	Conclusion	125

Ce chapitre traite de la modélisation de l'environnement en milieu urbain afin d'aider les manœuvres de parking. Contrairement au chapitre précédent dans lequel nous avons traité uniquement de la détection d'obstacles sans suivi temporel, la modélisation de la place de parking devra prendre en compte l'aspect temporel. Le véhicule, en phase de recherche d'une place de parking, perçoit son environnement de manière incrémentale. La fonction de modélisation de l'environnement doit donc fusionner l'ensemble de ces vues afin de construire un modèle complet comprenant les obstacles et l'espace libre, modèle pouvant servir à détecter une place libre adaptée au véhicule, puis à un calcul automatique de trajectoire. Par contre, le cas des objets dynamiques n'a pas été considéré dans cette étude.

Après une mise en contexte en section 4.1 et un rapide état de l'art dans la section 4.2, nous verrons en section 4.3 les premières approches envisagées pour extraire les principales caractéristiques de la scène. Ensuite, en section 4.4, nous présenterons les méthodes d'étiquetage des points 3D selon les trois catégories : *sol*, *obstacle* et *trottoir*. La section 4.5 donnera les méthodes de segmentation des obstacles par une représentation de type **Boîte Englobante**. Cette section présentera la modélisation d'une scène locale dans un premier temps, puis donnera les algorithmes de fusion de scène dans un deuxième temps pour terminer par le calcul de l'espace libre, qui caractérise une place de parking. En fin de chapitre, nous verrons des résultats sur images de synthèse avant de conclure.

4.1 Description du problème

Comparée à une fonction de détection d'obstacles dans l'environnement relativement structuré qu'est un milieu routier, la modélisation d'une place de parking en milieu urbain doit pouvoir s'adapter à une très grande variété de configurations possibles. Tout d'abord, on distingue trois catégories principales de place : créneau (parallèle à la route), en bataille (perpendiculaire à la route) et en épis (en oblique). De plus, la difficulté principale consiste à délimiter les places libres, car ces limites peuvent être aussi bien des voitures, des murs, des buissons, des lignes blanches ou même des plots. La figure 4.1 donne quelques exemples en image de synthèse. Bien que l'environnement soit structuré il est possible que différents composants habituels du milieu urbain, ne soient pas présents ; par exemple, le trottoir n'est pas toujours présent ou pas toujours visible. Il n'est donc pas possible de baser la modélisation sur des hypothèses dures sur la présence dans tous les cas, d'éléments dans les images.

4.1.1 Repères et définitions

Dans la suite de ce chapitre, les différents repères suivants seront utilisés :

- **Repère voiture** $R_{voiture}$: il est situé sur le milieu de l'essieu arrière, X vers l'avant du véhicule, Y sur sa gauche et Z vers le haut,
- **Repère caméra** R_{cam} : situé sur la caméra gauche, X pointant en avant, Y sur la gauche et Z vers le haut. Le calibrage du repère caméra par rapport au repère voiture est calculé *a priori* grâce aux plans du véhicule et une vérification *a priori* sur une reconstruction 3D sur sol plan et sans obstacle. Plusieurs configurations sont possibles pour l'emplacement du capteur. Si l'on souhaite observer les deux côtés d'une voie avec un seul capteur, celui-ci doit être positionné au milieu, soit à l'avant, soit à l'arrière du véhicule. Par contre, si on souhaite dédier le système à un côté de l'environnement (typiquement, à droite dans nos contrées), il est préférable de positionner le capteur sur une aile avant ou arrière du véhicule, ou de l'intégrer au rétroviseur extérieur.

- **Repère monde R_W** : il correspond à $R_{voiture}$ lors de la marche avant pour la recherche d'une place de parking. A ce repère monde est associé une zone de détection, appelée **fenêtre glissante**, zone mémoire de notre système sur laquelle est appliqué l'ensemble des traitements. Tout élément plus loin que la taille de la fenêtre glissante est "oublié" pour ne pas surcharger les traitements avec des informations obsolètes.

A chaque itération de l'algorithme deux modèles $3D$ sont créés ou mis à jour :

- **modèle local** : modèle de la scène courante construit à partir d'une seule paire stéréoscopique,
- **modèle global** : modèle issu de la fusion des modèles locaux construits dans la fenêtre glissante.

Plusieurs grilles de discrétisation sont créées dont les facteurs de discrétisation tiennent compte de :

- σ_X : résolution en profondeur,
- σ_Y : résolution latérale,
- σ_Z : résolution en hauteur,

tous trois liés au modèle d'erreur du banc stéréoscopique utilisé.

Donnons les quelques définitions suivantes :

- **Sol** : plan, sans discontinuité en profondeur, sur lequel le véhicule est supposé pouvoir circuler,
- **Obstacle** : tout élément situé sur le sol et dont la hauteur est supérieure à celle franchissable par le véhicule,
- **Trottoir** : plan parallèle au sol démarqué par une discontinuité de hauteur,
- **Limite sol/trottoir** : limite entre le sol et le trottoir. Dans cette étude, non considérons que cette limite est non curviligne. Une place libre est recherchée typiquement dans une rue en ligne droite, mais notre méthode pourra facilement s'adapter pour la recherche dans des courbes de grand rayon.

4.1.2 Spécifications

Un véhicule qui cherche une place de parking dépasse rarement la vitesse de $20km/h$ soit $5.55m/s$. Si nous souhaitons avoir une acquisition tous les dix centimètres pour obtenir un modèle particulièrement dense, cela revient à une fréquence de traitement de $52Hz$. Raisonnablement, vu la densité des points apportée par le capteur stéréoscopique, nous pouvons limiter l'acquisition à 40 ou 50 centimètres, soit des fréquences de traitement entre $10.4Hz$ et $13Hz$. Dans ce chapitre, nous considérerons comme temps réel une application dont le temps de traitement ne dépasse pas les $100ms$, soit une acquisition tous les $52cm$.

Les besoins en résolution des mesures $3D$ sont dictés par les dimensions des objets de l'environnement du véhicule. Par exemple, une boule d'attelage (ISO 50) présente un déport de $22cm$ qu'il sera utile de détecter lors de la manœuvre. Les hauteurs des trottoirs varient entre $7cm$ pour les plus bas et $15cm$ pour les plus hauts. Il est donc important d'avoir une résolution dans cet ordre de grandeur afin de le détecter. Dans cette étude, les objectifs en terme de résolution des mesures sont fixés à $10cm$.

4.2 Etat de l'art

La complexité croissante des environnements urbains ouvre le champ à de nouvelles applications d'aide au conducteur telle que l'aide au parking. Cette fonction se compose d'une boucle *perception-planification-exécution* nécessitant une large palette de techniques. La planification d'un robot mobile non-holonyme a longuement été étudiée [Kondak et al., 2001], [Walsh et al., 1994] ; des planificateurs temps réel sont même disponibles [Simeon et al., 2001]. [Driss et al., 1994], [Xu et al., 2000] et [Wada et al., 2003] proposent des outils complets de la perception à la planification. Malgré ces études existantes, on peut regretter une grande faiblesse des outils de perception proposés au vu de la difficulté que pose l'aide au parking.



FIG. 4.1 – Variété des configurations possibles de parking : [haut] milieu structuré créneau et bataille ; [bas] milieu non structuré

De nombreux systèmes commerciaux sont disponibles à des prix allant de 50 à 150 € (voir le site www.parking-sensor.co.uk, *Saab 9-5*, *Mercedes* ...). L'ensemble de ces systèmes reposent sur des ultrasons dont le manque de robustesse et la limite de portée sont bien connus de la communauté robotique. D'autres systèmes commerciaux tels que celui monté sur la *Toyota Prius*, propose un système complet d'aide au parking (boucle complète de la perception à l'exécution de la manœuvre). Les démonstrations ont montré une lenteur très importante de la manoeuvre et une faible généralité du système vis-à-vis des configurations de parking. Le système proposé par Toyota ne mesure pas la place, mais comme nous le voyons sur la figure 4.2, le conducteur a la charge de placer un rectangle dans l'image de retrovision à l'endroit où il veut se garer. Le système calcule alors automatiquement la manœuvre à réaliser sur cette seule information. La production de ce système, même s'il était limité au Japon, a été annulée.

De nouvelles approches à base de réseaux de neurones ont été proposées avec l'utilisation d'une caméra. Les systèmes proposés fonctionnent sur données de synthèse [Driss et al., 1994] ou travaillent sur la détection des lignes blanches [Xu et al., 2000], méthode utile uniquement en pré-



FIG. 4.2 – Interface graphique du système de parking automatique de la Toyota Prius

sence de marquages. Les Japonais, dont le problème de congestion des routes pose des difficultés au parking, sont très actifs dans le domaine. [Wada et al., 2003], dans le cadre du projet **iCAN**, propose une gestion centralisée des parkings dans lequel un ordinateur principal relié à des caméras de surveillance, informe les véhicules des places disponibles. [Ono et al., 2002] proposent un système à base de télémètre $2D$ laser. Non seulement ces méthodes fondées sur la vision monoculaire ou sur des capteurs laser ou ultra-sons, génèrent uniquement des modèles $2D$, insuffisants pour garantir la sécurité de la manœuvre, mais de plus, les performances sont très dépendantes de la couleur des véhicules ou des conditions de temps et d'illumination.

En raison des coûts ou des limitations de ces différents capteurs et voulant rester totalement indépendant de l'infrastructure, nous avons souhaité utiliser une approche différente. Vu la complexité de l'environnement, il nous paraît important d'utiliser un capteur $3D$ pour ne pas être limité par un simple modèle $2D$. Pour toutes ces raisons, nous avons décidé de choisir la stéréovision, compromis entre performances et coût. Concernant l'aide au parking à partir de données stéréo, nous ne connaissons que les travaux de [Kaempchen et al., 2002] qui proposent un système dont le principe est l'extraction de plans dans un nuage $3D$ issu de la stéréovision. Le principe de détection des obstacles repose sur le *L-fitting* [Suski, 2001] : un modèle en forme de L parcourt toute l'image $3D$ projetée sur le sol et compte, pour chaque position et par accumulation, le nombre de points en accord avec le modèle. Ce nombre correspond à la probabilité de la détection du véhicule. Une base de quatre modèles de places de parking est apprise et comparée à la perception courante. Un algorithme *ICP* (Iterative Closest Point) flou [Krebs, 2000] est utilisé pour fusionner les différentes scènes acquises dans le temps. Chaque point est d'abord assigné à un modèle d'objet afin de faciliter l'appariement entre l'image $3D$ et le modèle de place de parking. Le système résultant est très éloigné du temps réel et son manque de généralité limite trop les configurations acceptables.

La section suivante présente nos premiers travaux réalisés lors de stages ingénieurs dans [Lemondé, 2002] puis [Nguyen, 2003], afin d'évaluer pour cette application, des techniques robustes issues de la robotique pour la modélisation et la localisation d'un robot mobile.

4.3 Evaluation des techniques issues de la Robotique

Plusieurs techniques existent déjà dans le domaine de la robotique pour détecter les obstacles, extraire des plans dans une image $3D$ et fusionner des informations issues de différentes

acquisitions. Nous allons dans cette section, présenter deux techniques classiques en Robotique : la segmentation plane et le recalage de données par ICP. Nous les avons étudiées pour essayer de les adapter à notre application.

4.3.1 Extraction du plan du sol

Les premiers travaux sur la perception de la place de parking ont été consacrés à l'extraction du sol à partir de l'image 3D. Une connaissance *a priori* forte sur l'environnement d'une place de parking nous permet de considérer le sol comme étant l'élément prédominant dans l'ensemble des plans composant la scène et dont les normales ont des directions équivalentes et verticales. Pour exploiter cette hypothèse, nous avons repris les travaux de [Betge-Brezetz, 1996] de segmentation d'une image 3D afin d'en extraire les points du sol. D'après la définition donnée du sol, nous devons chercher dans la scène une grande surface d'orientation uniforme. Celle-ci correspond alors à un ensemble de points ayant tous des directions de normales équivalentes. Il faut donc estimer les normales en chaque point. Considérons pour cela l'image tridimensionnelle comme une surface définie par une équation paramétrique. Cette équation correspond à la forme générique d'une image 3D où i et j sont considérés comme les paramètres. Il est possible d'estimer la normale $\vec{N}_{ij}$ au point P_{ij} de la manière suivante :

$$\vec{N}_{ij} = \frac{\partial P_{ij}/\partial i \wedge \partial P_{ij}/\partial j}{\|\partial P_{ij}/\partial i \wedge \partial P_{ij}/\partial j\|} \quad (4.1)$$

avec :

$$\frac{\partial P_{ij}}{\partial i} = \begin{bmatrix} \frac{\partial x_{ij}}{\partial i} \\ \frac{\partial y_{ij}}{\partial i} \\ \frac{\partial z_{ij}}{\partial i} \end{bmatrix} \quad \text{et} \quad \frac{\partial P_{ij}}{\partial j} = \begin{bmatrix} \frac{\partial x_{ij}}{\partial j} \\ \frac{\partial y_{ij}}{\partial j} \\ \frac{\partial z_{ij}}{\partial j} \end{bmatrix} \quad (4.2)$$

D'une manière plus générale, $\vec{N}_P = [N_x \ N_y \ N_z]^T$ est la normale en un point P . Les dérivées en x , y et z sont estimées par des opérateurs locaux (filtre de Sobel par exemple). La direction de chacune d'elles dans un système de coordonnées sphériques est $\vec{N}_P \equiv (\alpha, \beta) \in [-\pi; \pi] \times [-\frac{\pi}{2}; \frac{\pi}{2}]$.

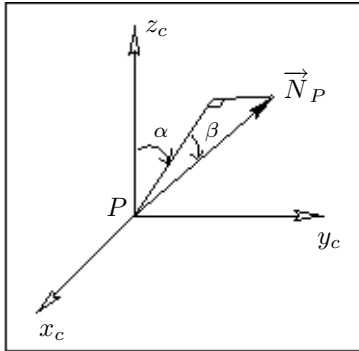


FIG. 4.3 – Système de coordonnées sphériques utilisé pour extraire le sol :

$$\begin{aligned} \alpha &= \text{atan2}(N_y, N_z) \\ \beta &= \arcsin N_x \end{aligned} \quad (4.3)$$

Le système de coordonnées sphériques utilisé est montré en figure 4.3. Il est tel que les coordonnées $(\alpha, \beta) = (0, 0)$ correspondent à des normales dirigées suivant l'axe z_c du repère voiture.

Le sol correspond à des points ayant des normales équivalentes, ici centrées en $(0,0)$ vu l'hypothèse de sol plat. L'histogramme bi-variable H des normales représente le nombre de points ayant une direction de normale donnée :

$$H(\alpha, \beta) = \text{Card} \{ P \mid \vec{N}_P \equiv (\alpha, \beta) \} \quad (4.4)$$

Cet histogramme des normales est construit sur l'ensemble des points $P_{ij}(i=1, N_i; j=1, N_c)$ de l'image. Pour cela, il faut discrétiser chacun des intervalles angulaires $[-\pi; \pi]$ et $[-\frac{\pi}{2}; \frac{\pi}{2}]$ respectivement en n_α et n_β sous-intervalles.

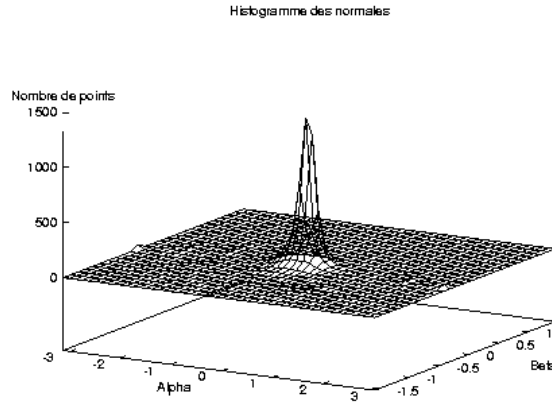


FIG. 4.4 – Histogramme bi-variable H des normales. Le sol étant majoritaire et uniforme, il apparaît comme un pic dans l'histogramme.

Pour une scène composée d'un sol uniforme, la prédominance de ce dernier apparaît comme un pic dans l'histogramme H des normales.

Une procédure récursive partant du sommet $H_{\max}$ de l'histogramme et descendant progressivement va sélectionner les points 3D correspondant à ce pic. Cette procédure sera ensuite rappelée pour d'autres pics. Le nombre n_α et n_β de sous intervalles dans H , ainsi que le voisinage pris autour de $H_{\max}$ pour déterminer quels points 3D appartiennent au sol, dépendent directement de l'erreur d'estimation des normales, erreurs obtenues par propagation des erreurs sur les points 3D eux-mêmes.

Tous les points sélectionnés dans un pic de l'histogramme H précédent ont des directions de normale équivalentes. La majeure partie d'entre eux appartient au sol, mais certains ont pu être pris sur des surfaces externes à ce sol avec une même orientation tel que le sommet des objets par exemple. Pour sélectionner seulement les points appartenant réellement au sol, le calcul de l'équation d'un plan $\mathcal{P}$ passant par l'origine du capteur et perpendiculaire à la direction vectorielle correspondant au pic permet la construction d'un histogramme mono-variable H' des distances au sol (cf figure 4.5). Il représente le nombre de points sélectionnés précédemment et situés à une distance d du plan $\mathcal{P}$:

$$H'(d) = \text{Card} \{ P \in H_{\max} \text{ de } H(d) \mid \text{dist}(P, \mathcal{P}) = d \}$$

Dans le cas de la figure 4.4 pour un pic $H_{\max}$ en $(0,0)$, le plan P n'est autre que le plan Oxy . L'examen des pics de cet histogramme H' permet de segmenter plan par plan les points qui ont des normales de même orientation, par exemple points sur la chaussée et points sur le trottoir.

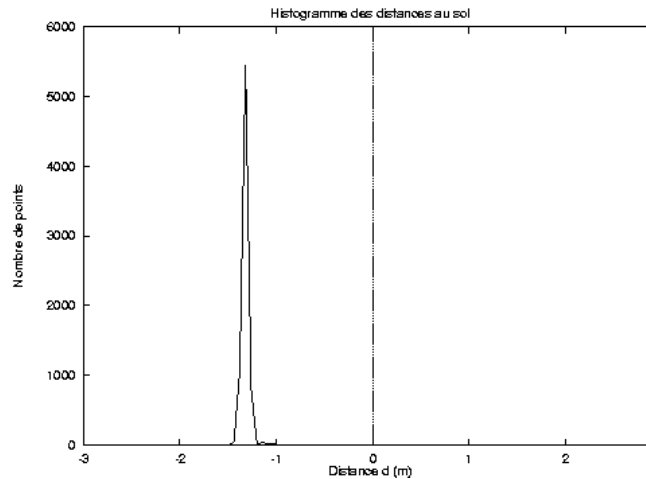


FIG. 4.5 – Histogramme mono-variable H' . Sélection des points de normales équivalentes, appartenant au même plan.

La figure 4.6 donne des résultats sur une image réelle. Malgré tout, cette approche séduisante de segmentation plane nous pose plusieurs problèmes : (1) les normales aux points sont souvent trop bruitées pour en déduire le plan du sol et (2) le temps de calcul nécessaire pour le calcul des normales et des histogrammes n'est pas acceptable vis-à-vis de notre application. Nous n'avons donc pas pu retenir cette approche pour notre système.

4.3.2 Recalage des données entre plusieurs scènes

Du fait des limitations des champs de vue de notre capteur, la modélisation de la place de parking doit être un processus incrémental. Lors de ce processus, le recalage est la fonction qui permet de positionner successivement les données acquises depuis un point de vue, par rapport au modèle construit jusque-là. Cela revient à estimer la transformation rigide entre la position courante et la position précédente du capteur. La solution la plus simple consiste à utiliser les informations odométriques disponibles sur le bus *CAN* du véhicule. Ces informations nous donnent la translation et la variation d'angle de lacet du véhicule entre deux positions sans tenir compte d'éventuels glissements lors des virages. Cette solution peut alors être retenue pour le recalage des données, surtout en faisant l'hypothèse que la recherche de la place de parking s'effectue en ligne droite. Par contre, la qualité de ces informations se dégrade lorsque la vitesse est inférieure à quelques kilomètres par heure (environ $3km$ en fonction des systèmes) et la synchronisation entre ces données et le système de vision n'est pas aisée.

Il est beaucoup plus pertinent d'extraire cette information de déplacement directement dans les images. Une première solution, l'odométrie optique, est celle qui a été mise en place dans le cadre de la recalibration du système stéréoscopique et permet, à partir de points d'intérêts extraits dans deux images successives, d'estimer le déplacement du capteur entre deux acquisitions. La description plus détaillée du système est donnée en chapitre 2. La seconde solution décrite dans le paragraphe suivant extrait la transformation recherchée à partir d'appariements denses de deux images de points *3D*.

ICP (Iterative Closest Point)

Comme dans la méthode de [Kaempchen et al., 2002], nous avons évalué l'algorithme ICP (*Iterative Closest Point*). L'algorithme se base sur un prétraitement appliqué aux données 3D pour sélectionner des points pertinents sur lequel *ICP* sera appliqué (typiquement, points de discontinuité 3D extraits des deux images successives).

L'algorithme ICP permet de calculer itérativement la transformation 3D T_t qui permet de superposer au mieux deux nuages de points 3D, ici pour superposer les points issus du modèle local m_t , acquis à l'instant t , avec le modèle global MG_{t-1} issu de la fusion des modèles locaux $ML_0 \dots ML_{t-1}$. Il faut disposer d'une estimée initiale T_t^0 ; à l'itération k , la position du point de vue courant est estimée par une transformation T_t^k . Un ensemble de points de contrôle ($p_t^i = 1 \dots$) a été sélectionné sur le modèle local ML_t . On leur applique la transformation T_t^k afin de les exprimer dans le repère du modèle global; pour chacun d'eux, on estime le point le plus proche sur l'ensemble des points de contrôle fusionnés dans le modèle MG_{t-1} . La transformation T_t^k peut alors être réestimée à partir de ces appariements, pour obtenir T_t^{k+1} . Ces opérations sont répétées itérativement jusqu'à superposition parfaite des modèles.

Malheureusement, les temps de traitement (environ 1 seconde par fusion) sont très largement supérieurs au temps alloué à la fusion de modèle dans notre application. Un exemple de résultat est donné en figure 4.7; il s'agit du recalage de toutes les images 3D générées par stéréovision, lors d'une séquence de type parking.

4.3.3 Conclusion

Ces premières approches, bien qu'impossibles à mettre en pratique, ont l'intérêt de préciser les spécifications de notre système. En résumé, les contraintes à appliquer sur notre système peuvent se formuler ainsi :

- le système doit être industriellement réalisable sans oublier le coût du matériel,
- pour pallier la variété possible des cas de détection il est souhaitable d'utiliser un capteur 3D,
- les solutions robustes de segmentation en plan sont difficiles à implémenter dans un contexte temps réel, il faut donc trouver une technique simple et rapide pour extraire le sol tout en gardant la robustesse donnée par l'accumulation des histogrammes,
- la fusion d'images par recalage de points 3D est complexe et il semble difficile de mettre en place une fusion de l'ensemble des points 3D calculés pour chaque scène. La solution retenue doit donc travailler sur des primitives 3D autres que le point ou doit exploiter une mesure du déplacement fournie par ailleurs (odométrie).

Nous allons donc voir dans les sections suivantes les solutions retenues afin de répondre à l'ensemble de ces contraintes. L'algorithme général de modélisation des obstacles est présenté en figure 4.8. Les éléments en rouge pointillé représentent la phase de détection des obstacles, c'est-à-dire d'étiquetage des points 3D avec un numéro d'obstacle. Les éléments en trait plein bleu correspondent à la phase de modélisation des obstacles, c'est-à-dire de création de primitives 3D regroupant les points 3D étiquetés. La phase de détection est présentée en section 4.4 et la fonction de fusion est décrite en section 4.5.



Histogramme des normales

Nombre de points

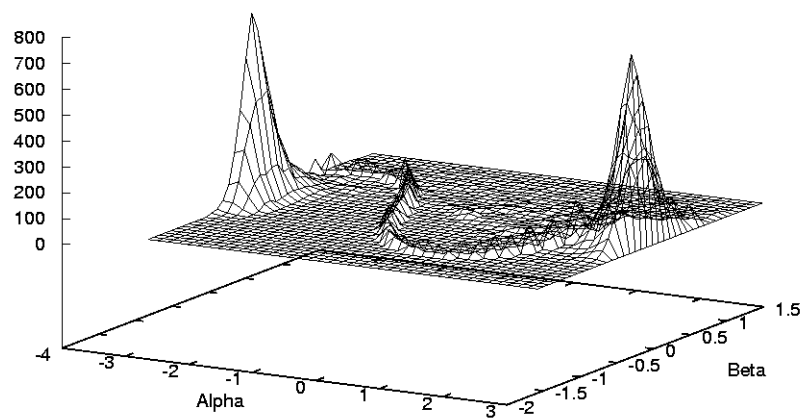


FIG. 4.6 – Détection des obstacles par segmentation de plan : [haut] image gauche de la stereo, [milieu] calcul des normales, [bas] histogramme des normales

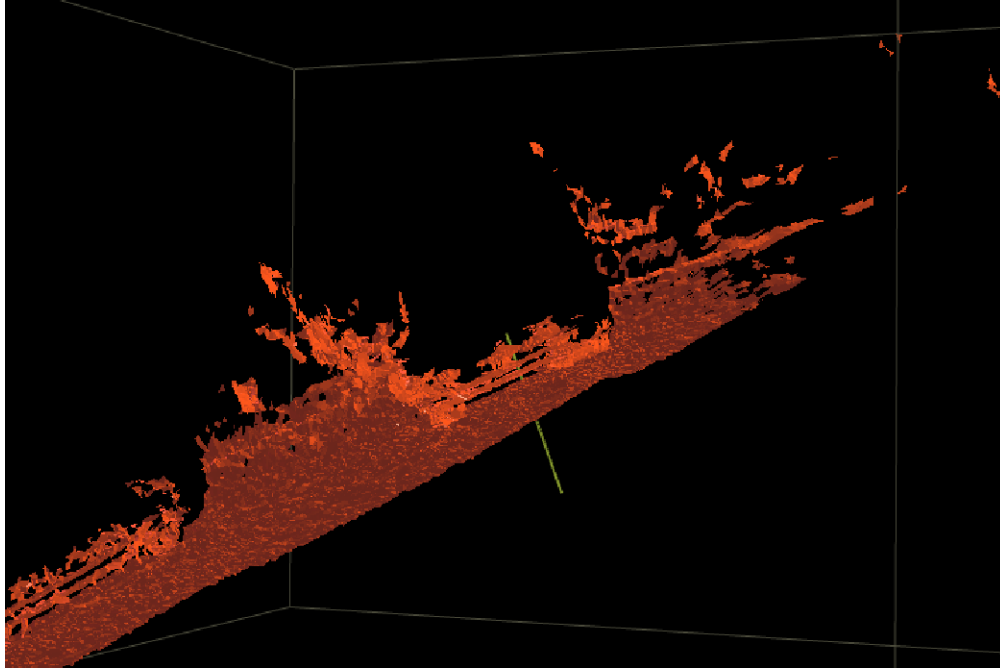


FIG. 4.7 – Recalage de scène avec l’algorithme ICP. L’ensemble des vues de la séquence a été fusionné dans un seul nuage 3D, ici présenté depuis un point de vue aérien de côté. En bas à gauche de la figure, apparaît le premier obstacle de la place en créneau puis, en allant sur la droite, l’espace libre et le deuxième obstacle délimitant la place. On peut “deviner” la présence du trottoir.

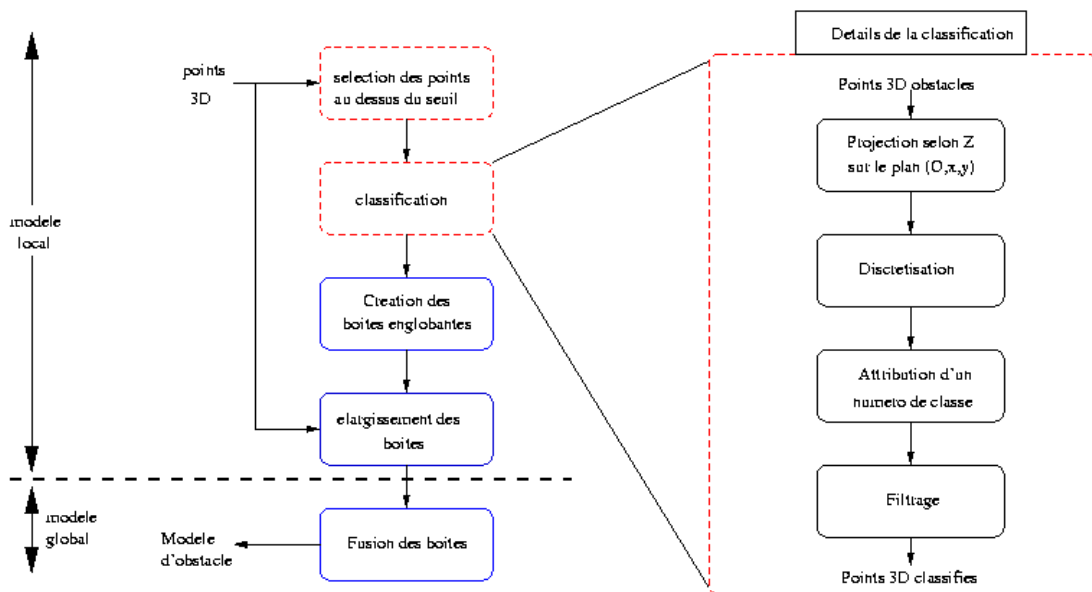


FIG. 4.8 – Algorithme d’extraction et de classification des obstacles

4.4 Détection des composantes de l'environnement

L'algorithme de détection d'obstacles se compose de deux phases : (1) la phase initiale de **détection des points obstacles** et (2) la phase de **classification** (différenciation des obstacles). Afin de répondre à la contrainte de détection rapide du sol, nous faisons ici une hypothèse de route plane. Cette hypothèse est justifiée dans notre contexte car, (1) en milieu urbain, les ruptures de pente sont plus rares qu'en milieu routier, et (2) nous nous intéressons à un champ de perception limité autour du véhicule rendant les variations de pente imperceptibles.

Notons cependant que, généralement les rues ont un sol légèrement bombé, de l'axe central vers les caniveaux ; avec un capteur *3D*, nous disposons de l'information *3D* et cette courbure du sol peut être négligée. Il n'en est pas de même, en vision monoculaire, si l'hypothèse du sol plat est appliquée pour rétroprojeter les points perçus dans les images, sur le plan du sol : les erreurs dues à la courbure du sol, peuvent en ce cas s'avérer non négligeables.

Pour limiter l'effet des artefacts ou autres erreurs issues de la stéréovision, nous appliquons des techniques de moyennage (calcul d'histogrammes) et de discrétisation. Une propriété est validée seulement si un nombre suffisant de points 3D satisfait un certain critère. Ces techniques rendent les méthodes plus **robustes** aux erreurs de perception, et facilitent l'adoption d'un formalisme **probabiliste** pour représenter l'environnement. Par exemple, la présence d'un obstacle dans une cellule discrétisée sur le plan de la chaussée peut facilement être modélisée par une probabilité conditionnelle $P(\text{obstacle}/\text{mesure})$. En Robotique Mobile, ce type de représentation discrète du sol est appelée *grille d'occupation* ; nous l'exploitons uniquement dans le modèle local, pour catégoriser les obstacles.

4.4.1 Extraction des obstacles

En modélisant d'abord les obstacles, nous avons, dans notre contexte, un critère discriminant : la hauteur maximale H_{max} que le véhicule peut franchir. Grâce à ce paramètre, et sur l'hypothèse de route plane ou localement plane, nous pouvons considérer que tout point 3D du nuage extrait par stéréovision, ayant une hauteur supérieure à cette valeur est un point obstacle. Chaque point *3D* $i = (x_i, y_i, z_i)$ est alors labellisé à l'aide d'une probabilité d'appartenance à un obstacle $p(i \in obs)$, probabilité affectée de la manière suivante, avec σ_Z écart type sur la hauteur :

$$p(i \in obs) = \begin{cases} 0 & \text{si } z_i < H_{max} - 1.5\sigma_Z, \\ 0.5 & \text{si } z_i \in [H_{max} - 1.5\sigma_Z, H_{max} + 1.5\sigma_Z], \\ 1 & \text{si } z_i > H_{max} + 1.5\sigma_Z. \end{cases} \quad (4.5)$$

Après la discrimination des points 3D en points obstacles, ou non, il faut regrouper les points *3D* appartenant aux mêmes obstacles. Cette classification a pour but d'attribuer un numéro d'obstacle à chacun des points du nuage 3D (cf. figure 4.8 droite). Pour faciliter les traitements de segmentation inter-obstacles, nous allons projeter l'ensemble des points détectés comme obstacle sur le sol (projection selon z) afin de travailler sur des informations 2D. La phase de discrétisation qui suit nous permet de créer une grille d'occupation facile, et donc rapide, à traiter. Le facteur de discrétisation Δ_d est choisi de telle manière que deux points qui n'appartiennent pas au même obstacle ne soient pas dans deux "cases" voisines de la grille :

$$\Delta d_{obs} = 3 \cdot \sigma_X \quad (4.6)$$

avec σ_X écart type sur la profondeur. Le choix d'un facteur constant Δd_{obs} est explicité sur la figure 4.9 à droite ; il doit garantir que deux obstacles séparés par cette distance minimale, sont

séparés par une cellule libre de la grille discrétisée, quelles que soient leurs configurations vis-à-vis de l'orientation de la grille. Nous avons vu que l'erreur en profondeur fournie par la stéréovision, évolue de manière quadratique ; par ailleurs, vu l'angle de vue, la densité des points sur le sol sera décroissante. Il serait donc judicieux de choisir un pas variable comme illustré en figure 4.9 à gauche.

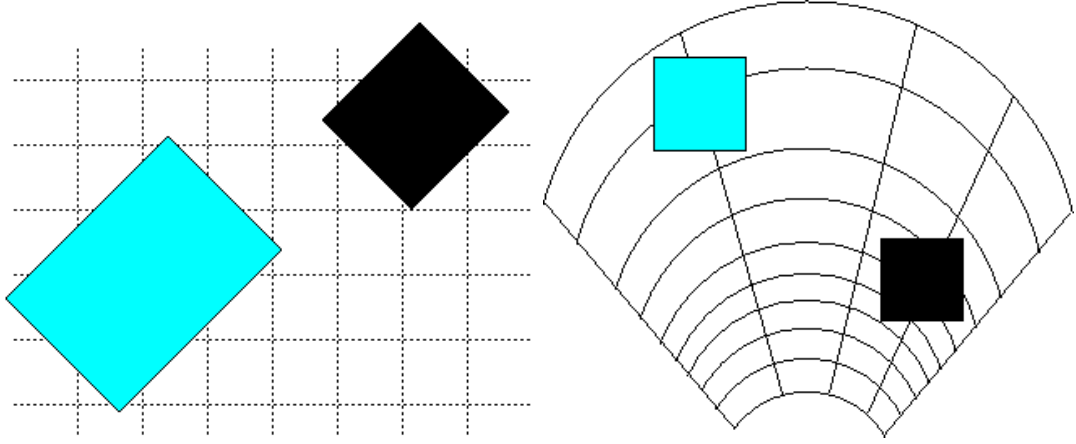


FIG. 4.9 – Grille de différenciation des obstacles de pas constant Δd_{obs} à gauche, de pas variable en profondeur à droite.

Après création de cette grille, il suffit de la parcourir et de regrouper les cellules connexes pour obtenir la classification des points en obstacles. Les cellules connexes ont donc le même numéro d'obstacle contrairement à des cellules totalement séparées dans la grille qui ont des numéros différents. Les points $3D$ projetés dans une cellule, sont étiquetés avec le numéro d'obstacle attribué à celle-ci.

Le filtrage des petits obstacles s'applique sur les données $3D$, et non sur la grille dans le plan Oxy . C'est un test fondé sur le nombre de points $3D$ qui ont le même numéro d'obstacle. Ce nombre n'est pas facile à exprimer analytiquement et dépend de beaucoup de paramètres (illuminations de la scène, angle de perception vis-à-vis de l'obstacle, texture de l'obstacle, ...). En pratique, ce nombre est calculé empiriquement.

4.4.2 Détection de trottoirs

Afin de détecter le trottoir depuis l'image $3D$ courante, nous allons supprimer des données $3D$ initiales (issues de la stéréovision) tous les points correspondant aux obstacles. De cette manière, il ne reste plus que la chaussée et le(s) trottoir(s). L'algorithme est présenté en figure 4.10.

Nous projetons ce nuage sur le plan OYZ afin d'avoir une coupe de la route avec cumul (simple projection et rotation, en fonction de la position des caméras sur le véhicule et du véhicule sur la voie). La détection du trottoir est la détection du point d'inflexion correspondant à la limite sol/trottoir comme le montre la figure 4.11.

Comme nous l'avons fait pour isoler les obstacles dans une grille d'occupation, nous allons discrétiser les points suivants z et y avec la valeur de discrétisation suivante :

$$\Delta d_{trot} = 3 \cdot \sigma_Z \quad (4.7)$$

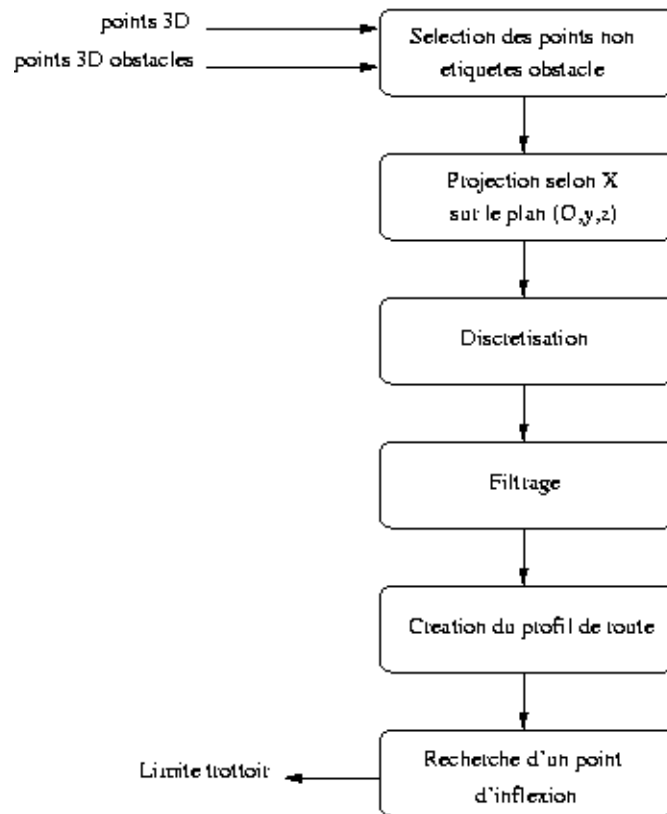


FIG. 4.10 – Algorithme d'extraction du trottoir

avec σ_Z écart type sur les hauteurs des points, donné par le modèle d'erreur du banc stéréoscopique utilisé. Comme pour la séparation des obstacles dans la grille Oxy , cette valeur de discrétisation garantit que les points du trottoir et ceux de la chaussée ne seront pas dans des cellules à la même altitude dans la grille du plan Oyz .

Les techniques de discrétisation et de moyennage que nous mettons en oeuvre dans ces approches ont l'avantage d'être efficaces en temps de calcul d'une part et de limiter l'effet d'artefacts liés à des erreurs de stéréovision d'autre part. Le résultat est une image binaire : chaque cellule est codée par un 1 ou par un 0 selon que des points 3D s'y projettent ou non ; 1 si points présents (en gris), 0 sinon (en blanc). Chaque cellule à la ligne k et la colonne l représente un ensemble de points 3D après discrétisation. Grâce à cette discrétisation, illustrée en figure 4.12, si deux

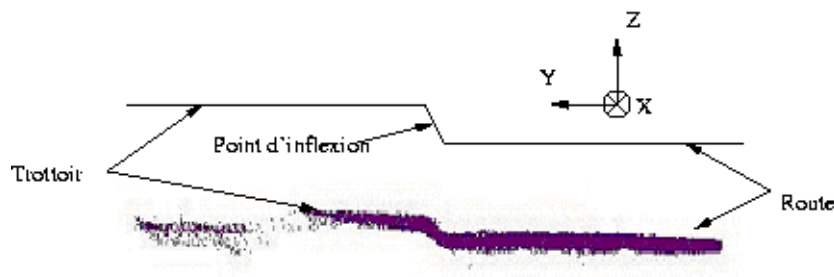


FIG. 4.11 – Projection des points du sol et du trottoir

cellules projetées et discrétisées sont à la même hauteur (suivant z), alors les points 3D qu'ils représentent appartiennent à la même surface horizontale. Du fait des problèmes de bruit dans l'image 3D, il peut arriver qu'il y ait des points isolés (sans voisin). L'analyse de la grille du plan Oyz colonne par colonne permet de calculer une hauteur moyenne, pour chaque colonne. Une suite d'interpolations linéaires est effectuée pour segmenter, par dichotomie, cette coupe transversale moyennée. Ces résultats sont alors analysés pour trouver la configuration qui s'apparie au mieux avec un profil en S typique d'une rupture de pente liée à un trottoir.

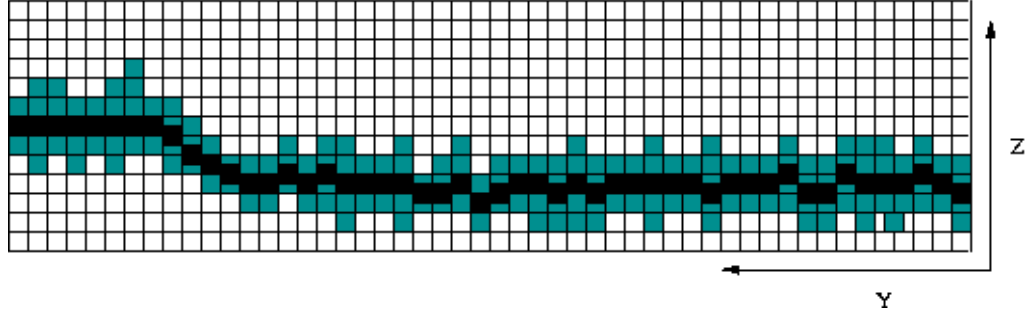


FIG. 4.12 – Interpolation du sol par une courbe de profil (coupe transversale et cumul)

La contrainte principale sur la détection du trottoir réside dans le fait que la première position du véhicule, lors de l'enclenchement du système, doit être parallèle au trottoir. Pour être rapide le système ne teste pas toutes les possibilités d'orientation du véhicule par rapport à la voie ; il corrige seulement l'image courante en orientation autour de Oz , afin de l'orienter comme l'était la première. Il serait possible de s'aider du marquage au sol, si celui-ci est présent, pour trouver la bonne orientation et donc la bonne projection à appliquer pour détecter le trottoir.

4.5 Modélisation de la place de parking

Une scène urbaine typique est constituée du sol de la voie, du sol des trottoirs (à droite et à gauche de la voie) et des obstacles émergeant du sol. Cela n'est pas restrictif : il peut ne pas y avoir de trottoir d'un côté, par exemple lorsque la voie est directement limitée par un obstacle (mur, haie, ...). Certaines situations complexes (bateaux sur les trottoirs, trottoir en bout de voie perçu uniquement au début du parcours de cette voie) ne seront pas considérées.

Dans la section précédente, nous avons décrit les méthodes pour extraire les informations issues d'une seule vue du capteur. Dans cette section nous décrirons comment regrouper ces informations pour en extraire un modèle local de l'environnement. Nous considérerons ensuite la construction d'un modèle de la voie de circulation, par fusion incrémentale des modèles de toutes les scènes perçues pendant que le véhicule parcourt cette voie. Enfin, nous décrirons comment ce modèle sera analysé pour détecter des emplacements possibles de stationnement pour le véhicule, donc des places libres.

4.5.1 Modèle local

Nous considérons un modèle local comme un ensemble d'obstacles et de limites sol/trottoir. Les places libres seront traitées seulement dans le modèle global. Considérant que le véhicule roule relativement en ligne droite le long de sa voie de circulation, nous modélisons une limite

sol/trottoir par une distance de ce trottoir au véhicule le long de l'axe Y . Nous faisons donc l'hypothèse que les trottoirs curvilignes ne font pas partie de l'environnement et que le véhicule n'effectue pas de grand virage au sein même de sa voie. Ces hypothèses ne sont pas trop restrictives puisque rares sont les places de parking dans un virage et il est tout à fait réalisable de ré-initialiser le système après qu'un virage soit détecté, par odométrie, par exemple.

La modélisation d'un obstacle par une enveloppe est, quant à elle, plus complexe. Cette enveloppe est construite à partir de la liste de points 3D affectés à cet obstacle. Il serait possible d'utiliser plusieurs représentations :

- cylindrique : typiquement pour une borne, un tronc d'arbre ou un piéton,
- sphérique : pour un buisson ...,
- cubique : pour un véhicule garé ou circulant sur la voie,
- par un ensemble de polygones ou maillage triangulaire pour une description fine d'un obstacle.

Ces différents types de représentation auront surtout un sens pour décrire un obstacle perçu depuis plusieurs points de vue. En effet, un tronc d'arbre après une seule perception apparaît comme une portion de surface cylindrique, qu'il serait trop long à ce stade d'identifier avec certitude (une telle portion cylindrique s'apparente à une portion de plan). Donc, l'enveloppe d'un obstacle dans le modèle d'une scène (une seule perception) sera seulement un parallélépipède.

Calcul de la boîte englobante d'un obstacle

Initialement, cette enveloppe est construite seulement à partir des points situés au-dessus de la chaussée (altitude supérieure au seuil de détection des obstacles vu en section précédente). Nous pouvons modéliser rapidement un ensemble de points perçus sur un obstacle j , par un parallélépipède rectangle M_{obs}^j orienté selon le repère d'inertie défini par la matrice d'inertie $M_{inertie}$ suivante :

$$M_{inertie} = \frac{1}{n} \sum_{i=1}^n [x_i - X_G, y_i - Y_G, z_i - Z_G]^T [x_i - X_G, y_i - Y_G, z_i - Z_G] \quad (4.8)$$

avec $(x_i, y_i, z_i)_{i=1..n}$ points de l'image 3D et (X_G, Y_G, Z_G) le centre de gravité.

Les axes d'inertie correspondent aux directions des vecteurs propres de $M_{inertie}$ et s'intersectent au centre de gravité. Chaque matrice d'inertie doit être conservée pour être réestimée lors d'une nouvelle observation de l'obstacle. L'intérêt de la modélisation de l'obstacle dans le repère d'inertie est illustré dans l'exemple présenté en figure 4.13. Sur ce schéma, une image 3D est visualisée selon deux points de vue : vue de dessus [gauche] et vue de côté [droite]. Les points bleus correspondent aux points étiquetés comme étant des points du sol et les points rouges comme des points obstacles. Les deux repères *voiture* et *inertiel* sont représentés par des axes de couleur : rouge = X , vert = Y et bleu = Z . Si un obstacle en travers de l'axe du véhicule est observé par son flan, la boîte englobante résultante du repère d'inertie (trait plein) est beaucoup plus proche de la réalité que si l'axe du véhicule avait été utilisé (pointillé). Le volume effectif de l'obstacle est de $0.84m^3$, volume donné par la représentation dans le repère d'inertie mais largement surestimé ($18m^3$) dans le repère voiture.

La représentation de l'obstacle, comme montrée en figure 4.14, est donnée par :

$$M_{obs}^j = [x_{max}^j, x_{min}^j, y_{max}^j, y_{min}^j, z_{max}^j, z_{min}^j]$$

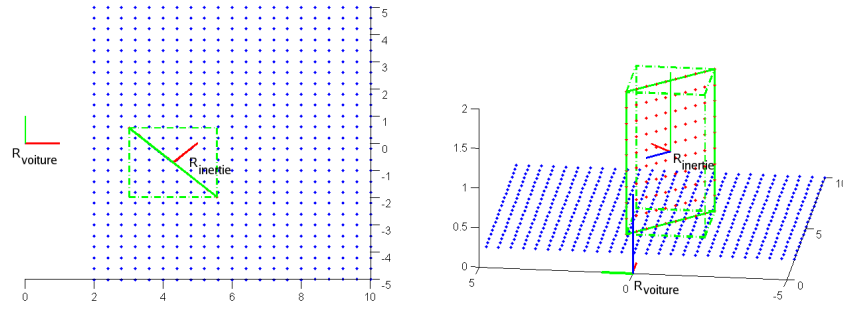


FIG. 4.13 – Modélisation des obstacles dans le repère d'inertie (trait plein) et dans le repère voiture (pointillé) selon deux vues : de dessus [gauche] et de côté [droite]. Les points bleus sont étiquetés comme points du sol et les points rouges comme points obstacles.

tels que $x_{max}^j, x_{min}^j, y_{max}^j, y_{min}^j, z_{max}^j$ et z_{min}^j sont les bornes des coordonnées $[x_{ij}, y_{ij}, z_{ij}]^T$ des N^j points affectés à cet obstacle dans le repère d'inertie, bornes définies par les coordonnées des points qui sont à des distances Max et Min du centre de gravité dans les trois axes, points donnés par : $\forall i = (1..N^j)$ et $\forall j$,

$$x_{max}^j = X(Max(x_{ij})) \quad (4.9)$$

$$x_{min}^j = X(Min(x_{ij})) \quad (4.10)$$

$$y_{max}^j = Y(Max(y_{ij})) \quad (4.11)$$

$$y_{min}^j = Y(Min(y_{ij})) \quad (4.12)$$

$$z_{max}^j = Z(Max(z_{ij})) \quad (4.13)$$

$$z_{min}^j = Z(Min(z_{ij})) \quad (4.14)$$

$$(4.15)$$

Les incertitudes reportées sur les dimensions de l'obstacle (σ_L^j sur la largeur L , σ_H^j sur la hauteur H et σ_P^j sur la profondeur P), correspondent aux incertitudes sur les coordonnées des points extrêmes ayant servi à créer la boîte englobante :

$$L = |y_{max} - y_{min}| \text{ avec } \sigma_L^j = \sigma_Y(Min(y_{ij})) + \sigma_Y(Max(y_{ij})), \quad (4.16)$$

$$H = |z_{max} - z_{min}| \text{ avec } \sigma_H^j = \sigma_Z(Min(z_{ij})) + \sigma_Z(Max(z_{ij})), \quad (4.17)$$

$$P = |x_{max} - x_{min}| \text{ avec } \sigma_P^j = \sigma_X(Min(x_{ij})) + \sigma_X(Max(x_{ij})). \quad (4.18)$$

Notons que cette modélisation par boîte englobante s'apparente à la méthode proposée par *S.Betge-Brezetz* pour modéliser des obstacles émergeant du sol par des ellipsoïdes. Pour ce faire, différentes stratégies avaient été proposées pour éliminer des points aberrants lors du calcul des minimum et maximum sur les axes d'inertie. Nous n'appliquons pas ces stratégies pour éviter d'alourdir les traitements ; par ailleurs, en principe, vu comment les points $3D$ sont associés aux obstacles par discrétisation du sol, nous ne pouvons pas avoir des points aberrants à des distances supérieures au pas de la grille.

Calcul de la boîte englobante totale d'un obstacle

Jusqu'à présent, nous n'avons détecté et modélisé que les parties supérieures (parties plus hautes que le seuil franchissable par le véhicule) des obstacles d'une scène. Etant donné que



FIG. 4.14 – Création des boîtes englobantes

chaque obstacle j est modélisé par un parallélépipède, nous affectons aussi à cet obstacle les points $3D$ perçus sur la chaussée ou sur les trottoirs dans la scène courante, en dessous du parallélépipède M_{obs}^j construit précédemment. Cela revient à sélectionner, dans le repère propre de chacun des obstacles, les points du sol $[x, y, z]$ tels que : $x \in [x_{min}^j, x_{max}^j]$ et $y \in [y_{min}^j, y_{max}^j]$ avec $x_{min}^j, x_{max}^j, y_{min}^j$ et y_{max}^j les bornes du parallélépipède défini précédemment pour représenter la partie supérieure de l'obstacle j . Cette intégration des points est très importante pour la détection du trottoir qui se base sur la détection d'une rupture de pente. En effet, par définition, le sol est une surface plane sans discontinuité en hauteur et en profondeur. Si la base des obstacles reste intégrée au sol, le bruit généré par ces points $3D$ risque de perturber la détection de la cassure de pente.

La seule différence introduite par cette étape, consiste à introduire dans la surface au sol couverte par un obstacle, les points perçus sur le sol sous cet obstacle. Cela a une faible incidence pour les points perçus sous une voiture. Par contre, si une excroissance (rétroviseur, objet dépassant de l'arrière d'un véhicule garé) est perçue, la zone au sol sous cette excroissance sera intégrée dans l'obstacle, donc non navigable si on considère que toutes les parties du sol non couvertes par les obstacles sont l'espace navigable lors de futures manœuvres. Vu ce traitement, les obstacles hauts pourraient poser problème : un cas typique concerne la branche d'arbre au-dessus d'une place mais qui n'empêche pas le véhicule de se garer. De toute évidence, ce cas ne concerne que les objets en hauteur. L'ensemble des configurations testées pour la position et l'orientation du capteur stéréoscopique montre qu'il est important d'incliner sensiblement le capteur vers le bas pour (1) éviter les angles morts autour du véhicule et (2) limiter la détection à la zone que ne peut pas voir le conducteur, c'est-à-dire la zone proche du sol et aussi (3) d'éviter de détecter des objets hauts, tels que les branches d'arbre.

4.5.2 Modèle global

La modélisation d'une scène locale est souvent imparfaite et dépend de nombreux paramètres (bruits, situations ambiguës, manque de perspectives...). De la même manière que l'interprétation d'une scène par l'homme, nous avons besoin de plusieurs scènes successives avant de pouvoir conclure sur l'exactitude des informations perçues. Nous pouvons utiliser les propriétés de la continuité des scènes pour compléter la modélisation de l'environnement et filtrer les erreurs de

modélisation.

A chaque perception, les résultats de la stéréovision sont décrits dans un repère lié au véhicule. Afin de représenter l'environnement par fusion des perceptions successives, nous devons introduire un repère fixe, appelé **repère monde**. Rappelons que les données issues de la détection d'obstacles seront conservées uniquement sur les derniers mètres parcourus, donc dans une **fenêtre glissante** qui suit le mouvement de la voiture en marche avant. Le repère lié à la route sera donc modifié dynamiquement selon une translation le long de l'axe X vers les X positifs pour un mouvement parfaitement rectiligne vers l'avant du véhicule. Une rotation doit être appliquée aux différentes images $3D$ perçues, pour corriger une éventuelle modification de cap du véhicule. La position de la fenêtre glissante dépend de la position du capteur. Par exemple, si le capteur est monté à l'arrière :

- le modèle global se limitera à une zone de 20 m . derrière le véhicule.
- après détection d'une place libre, afin de mémoriser pendant l'exécution de la manoeuvre en marche arrière, les positions des obstacles perçus par le véhicule précédemment en marche avant, la fenêtre glissante est complétée par une zone de 5 mètres vers l'avant.

La fusion des différents modèles construits pendant le mouvement du véhicule produit un modèle fusionné des scènes ou **modèle de voie** de circulation. A partir d'une paire d'images acquises à un instant t , notre système construit un modèle local ML_t , exprimé dans le repère V_t du véhicule. Ce modèle local est fusionné au modèle global MG_{t-1} , le modèle de voie issu de la perception précédente, pour produire le modèle courant de la voie MG_t . Pour cela, plusieurs étapes sont nécessaires :

- exprimer ML_t et MG_{t-1} dans le même repère,
- fusionner les informations contenues dans ces modèles,
- filtrer les objets contenus dans le modèle fusionné pour oublier les informations de MG_{t-1} qui sortent de la fenêtre glissante dans MG_t .

Recalage des données issues de différentes perceptions

Comme nous l'avons vu dans la présentation des techniques issues de la Robotique, il serait possible d'utiliser des algorithmes du type *ICP* afin de recalcr les informations 3D des vues ML_t et MG_{t-1} , mais les temps de calcul nécessaires ne sont pas compatibles avec notre application. Il est beaucoup plus simple et rapide d'utiliser l'odométrie (optique ou non) pour nous donner l'information de position entre deux vues successives. Même s'il est bien connu que l'intégration des mesures obtenues par des odomètres sur les roues du véhicule, pose des problèmes de cumul d'erreurs, dans notre cas, ces erreurs auront peu d'impact, car (1) la mesure du déplacement par odométrie d'un véhicule avançant en ligne droite, est plutôt précise et (2) nous intégrons uniquement les déplacements sur les 20 derniers mètres parcourus.

Cependant, dans notre contexte, selon le type de scènes, d'autres méthodes de recalage sont envisageables :

- si les trottoirs sont détectés, recalage des vues en orientation en fusionnant les droites modélisant les limites sol/trottoir,
- utilisation des boîtes englobantes des obstacles pour le recalage des positions latérales et longitudinales du véhicule le long de la voie, par fusion des limites des boîtes modélisant ces obstacles.

Ces méthodes simples dépendent de la structure des scènes rencontrées (il faut des trottoirs, des obstacles ...). Elle peuvent nous aider dans le cas d'une odométrie défailante (problème d'appariement de points d'intérêt pour l'odométrie optique ou vitesse trop faible $\leq 3km$ pour

l'odométrie mécanique) et aussi afin d'aider à la synchronisation de l'odométrie mécanique avec les images.

Gestion de la fenêtre glissante

Une fois que tous les points $3D$ sont placés dans un même repère de référence, il convient de ne conserver que les informations appartenant à la fenêtre glissante le long de la voie pendant le mouvement vers l'avant du véhicule portant la caméra. A chaque nouvelle acquisition à l'instant t , les données perceptuelles courantes sont recalées avec les données perçues lors des perceptions précédentes, puis le modèle est filtré pour ne considérer que la fenêtre glissante. Cela est fait de manière très simple en ne conservant qu'un nombre donné N_{modele} de modèles locaux dans une liste circulaire dont la longueur est à déterminer en fonction de deux paramètres :

- la vitesse du véhicule qui cherche une place,
- le temps de traitement.

Avec l'hypothèse d'une vitesse de $20km/h$, donc environ $5m/s$, si nous traitons 10 images par seconde, la liste de modèles locaux conservés a 50 éléments. Ce nombre d'éléments pourrait être adapté en fonction de la vitesse. A chaque fusion, le modèle local courant remplace le plus ancien de la liste. Le modèle de la voie est calculé dans le repère courant du véhicule, à partir des modèles locaux présents dans la liste. Pour éviter de stocker trop d'informations,

- seuls les points $3D$ sur les obstacles sont conservés (pour affinement possible des boîtes lors de la planification des manœuvres),
- à chaque reconstruction du modèle de la voie, les changements de repère ne concernent que les boîtes englobantes et les limites de la chaussée. Les points $3D$ associés aux obstacles sont gardés dans le repère d'inertie. Il est inutile (même en cas d'affinement) de les mettre dans le repère global.

En figure 4.15, nous représentons, dans le cas d'un capteur situé à l'arrière, le graphe des repères considérés à l'instant t :

- repère courant du véhicule dans lequel est exprimé le modèle global courant,
- repère des N_{modele} dernières perceptions,
- repères des obstacles détectés et fusionnés lors de ces perceptions.

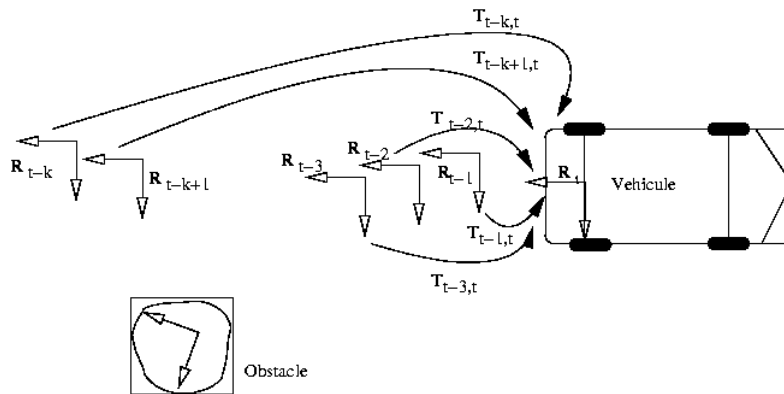


FIG. 4.15 – Fusion des perceptions locales par un graphe de repère

Pour pouvoir faire une transformation sur les données incluses dans ML_i , il faut conserver et mettre à jour à chaque itération, une transformation homogène $T_{i,t}$ entre la position du repère du véhicule dans lequel est exprimé un modèle local ML_i acquis à l'instant i et la position courante

du véhicule à l'instant t . Nous conservons aussi l'incertitude sur cette transformation, avec une matrice de covariance notée $\sigma T_{i,t}$.

Lors de la fusion à l'instant t , on dispose dans la liste circulaire des N_{modele} derniers modèles locaux ML_i , avec pour chaque modèle, la dernière transformation $T_{i,t-1}$ et l'incertitude associée $\sigma T_{i,t-1}$. Notons U_t la transformation homogène qui contient le déplacement du véhicule entre les instants $t-1$ et t . Notons σU_t l'incertitude associée à U_t . La transformation $T_{t-1,t}$ est initialisée avec U_t . La mise à jour des transformations à l'instant t est faite de la manière suivante :

$$T_{i,t} = T_{i,t-1}U_t = f(T_{i,t-1}, U_t) \quad (4.19)$$

$$\sigma T_{i,t} = \Psi_T(T_{i,t-1})\sigma T_{i,t-1}\Psi_T^t(T_{i,t-1}) + \Psi_U(U_t)\sigma U_t\Psi_U^t(U_t) \quad (4.20)$$

La fonction $f(T, U)$ représente la composition des deux transformations données en paramètres, T et U . L'incertitude sur le résultat est obtenue par propagation, en exploitant une décomposition au premier ordre de la fonction f . La fonction Ψ_T représente la jacobienne de f par rapport au paramètre T et la fonction Ψ_U représente la jacobienne de f par rapport au paramètre U . L'exposant t indique la transposition de la matrice.

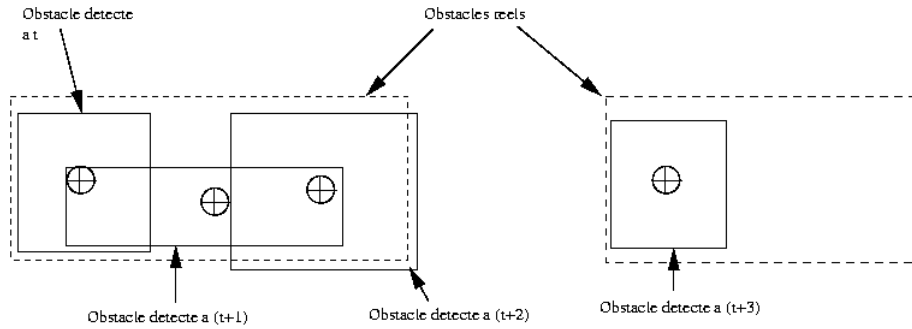


FIG. 4.16 – Fusion de boîtes englobantes

Construction du modèle fusionné

La construction du modèle fusionné des N_{modele} derniers modèles concerne les informations suivantes :

- fusion des boîtes associées aux obstacles statiques,
- fusion des limites des chaussées.

Fusion des obstacles statiques :

Concernant les obstacles statiques, il existe deux cas (cf. figure 4.16) pour chaque combinaison (ou chaque paire) d'obstacles détectés respectivement dans le modèle local et le modèle de la voie :

- les boîtes représentent des obstacles différents,
- les boîtes représentent en réalité, un seul et même objet.

Pour le premier cas, il suffit de mémoriser séparément dans MG_t les deux modèles des deux obstacles issus l'un de ML_t et l'autre de MG_{t-1} . Dans le deuxième cas, il faut créer dans MG_t , un modèle, fusion des deux modèles d'un obstacle observé suivant des points de vue différents. Nous devons construire un nouveau parallélépipède pouvant contenir les deux parallélépipèdes

représentant des parties différentes du même obstacle. Rappelons la définition de la fusion d'informations indépendantes concordantes tirée de [Fabiani, 1996] :

Définition 1 (Informations indépendantes concordantes : fusion associative non-idempotente)
 Soit $(\Omega, \mathcal{A})$ un espace mesurable de mondes possibles, soit $\mathcal{H} = \{\mathcal{H}_i, i \in I\}$ une tribu d'hypothèses d'état de $\mathcal{A}$, si deux sources d'information **indépendantes** S_1 et S_2 donnent respectivement une mesure de probabilité P_{S_1} et P_{S_2} sur $\mathcal{H}$ avec des degrés de confiance respectifs C_{S_1} et C_{S_2} , soit $d_{s,1,2}$ un seuil de décision associé à une fonction d'évaluation D_{1-2} de la distance entre P_{S_1} et P_{S_2} :

$$D_{1-2} : (P_{S_1}, P_{S_2}) \rightarrow D_{1-2}(P_{S_1}, P_{S_2}) \geq 0$$

$$D_{1-2} = 0 \Leftrightarrow P_{S_1} = P_{S_2}$$

D_{1-2} distance de Mahalanobis, alors :

$$\left\{ \begin{array}{ll} \text{si } D_{1-2}(P_{S_1}, P_{S_2}) \leq d_{s,1,2} & P_{1\&2} = \frac{C_{s1} \cdot P_{s1}(\mathcal{H}_i) + C_{s2} \cdot P_{s2}(\mathcal{H}_i)}{C_{s1} + C_{s2}} \\ \text{sinon} & \begin{array}{l} C_{1\&2} = C_{s1} + C_{s2} \\ \text{les informations 1 et 2 ne sont pas fusionnées.} \end{array} \end{array} \right. \quad (4.21)$$

Cela revient à utiliser la distance de Mahalanobis pour évaluer la pertinence de la fusion entre deux obstacles. La fusion des probabilités et des confiances repose sur les règles classiques de fusion de données en statistique ou en estimation stochastique. Différentes heuristiques peuvent servir à déterminer les valeurs des confiances liées aux obstacles comme, par exemple, la densité des points dans la boîte englobante.

En pratique, nous définissons le critère de fusion suivant : si le barycentre d'un des deux parallépipèdes est inclus dans le volume de l'autre, alors il faut fusionner. Après vérification de ce critère, le calcul de la nouvelle boîte englobante est relancé avec la matrice d'inertie.

Par exemple, en figure 4.16, les obstacles perçus aux instants t , $t+1$ et $t+2$ seront fusionnés dans un obstacle unique appelé A , vu le recouvrement des boîtes et que le barycentre de l'une appartient à l'autre. Le nouvel obstacle B détecté à l'instant $t+3$ ne sera pas fusionné avec A vu que :

- le barycentre de la boîte associée à B n'appartient pas à la boîte fusionnée de l'obstacle A détecté aux instants précédents,
- la boîte englobante de B ne contient pas le barycentre de la boîte associée à A .

Un autre aspect concerne la détermination de l'incertitude sur la dimension des boîtes englobantes. Pour un modèle local, cette incertitude est calculée à partir des incertitudes associées aux points 3D attribués à cet obstacle. Sur la figure 4.17, nous présentons les perceptions d'un obstacle à trois instants t , $t+i$ et $t+i+j$. La dimension des ellipses représente l'importance des incertitudes sur les points extrêmes des boîtes englobantes. Cette incertitude est minimale pour la perception la plus proche de cette partie extrême (ici, perception à l'instant t) et non pour la perception la plus récente (perception à l'instant $t+i+j$), ni pour la moyenne des perceptions entre les instants t et $t+i+j$.

Donc, la limite d'une boîte englobante n'est modifiée que si la perception courante sur la partie de l'obstacle considérée est meilleure que les perceptions précédentes. Dans la figure 4.17, c'est à l'instant t qu'est obtenue la meilleure perception sur la limite en x_{max} de la boîte mais c'est à l'instant $t+i$ qu'est obtenue la meilleure limite en x_{min} . La perception à l'instant $t+i+j$ n'apporte pas d'informations supplémentaires.

Fusion des limites chaussée/trottoir :

Concernant la fusion des limites chaussée/trottoir, les droites approximant ces limites dans les N_{modele} modèles locaux conservés dans la liste circulaire, sont fusionnées à chaque itération, en appliquant des techniques robustes de fusion (*RANSAC* ou filtrage médian). Ces méthodes permettent d'approximer un modèle (ici la position du trottoir) à partir de mesures (ici, les différentes perceptions de ce trottoir dans les N_{modele} dernières acquisitions), sachant que certaines mesures peuvent être erronées. Par exemple, pour un modèle exprimé en $t + 3$, une méthode robuste permet de rejeter la détection du trottoir perçue en $t + 2$ dans la figure 4.18. Seules les perceptions obtenues aux instants t et $t + 1$ seront exploitées pour générer la position du trottoir fusionnée.

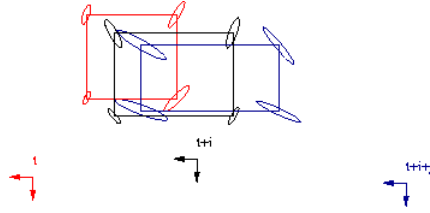


FIG. 4.17 – Problème de l'incertitude sur les dimensions des boîtes englobantes

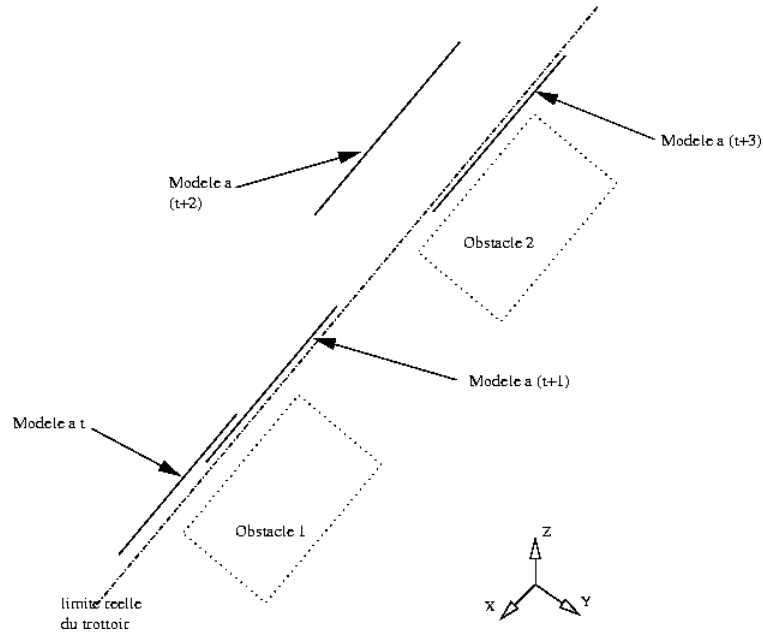


FIG. 4.18 – Principe de filtrage des positions du trottoir par une approche robuste

Affinement de la boîte englobante décrivant un obstacle

Jusqu'alors, notre représentation des obstacles par des boîtes englobantes, est assez frustrante. De manière à éviter le grossissement des boîtes du fait d'excroissances (rétroviseur ou boule d'attelage élargissant la boîte entière) ou simplement du fait de la forme des véhicules (aile ou parechoc plus ou moins arrondi), nous pouvons réaliser un affinement de cette représentation.

Cela rendra plus facile la planification des trajectoires. Les points acquis en chaque perception sur un obstacle sont conservés pour permettre un éventuel affinement de la boîte. Pour affiner une boîte, plusieurs possibilités présentées dans [Specht, 2005] sont possibles :

- approximation des points perçus sur un obstacle par un cylindre ou une sphère,
- construction d'un maillage à partir du nuage de points perçus sur l'obstacle depuis tous les points de vue,
- construction d'une carte d'occupation volumique dans le parallélépipède décrivant l'obstacle (cf. figure 4.19).

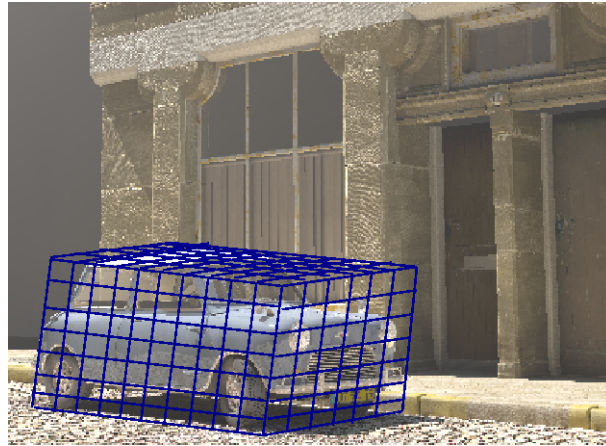


FIG. 4.19 – Affinement des boîtes englobantes par boîtes d'occupation volumique

L'approximation d'un ensemble de points par une surface quadrique (cylindre, sphère, ...) ne pose pas de problème, si tous les points perçus sur l'obstacle appartiennent à cette surface. A part des cas simples (une borne cylindrique, un ballon, un tronc d'arbre, ...), cette situation risque d'être rare. Par exemple, pour représenter un arbre (le tronc qui touche la chaussée, et la partie basse du feuillage qui est en général plus haute que le véhicule), une telle description nécessiterait de segmenter l'ensemble des points associés à l'obstacle, pour représenter différemment le tronc et le feuillage.

La construction d'un maillage à partir d'un nuage de points est un traitement lourd dont le temps de calcul paraît, à ce jour, incompatible avec les contraintes de l'application. Nous disposons de plusieurs méthodes pour ce faire :

- *Marching Cubes*, la plus efficace : méthode exploitant une discrétisation volumique préalable de l'espace contenant le nuage. Chaque sommet de cette discrétisation est analysé pour savoir s'il est dans l'objet (étiquette +) ou en dehors (étiquette -). Les cellules (ou voxels), dont les huit sommets ne portent pas tous la même étiquette, intersectent la surface limitant l'objet. Quinze configurations existent sur la répartition des étiquettes - et + sur une telle cellule. Pour chaque situation, un maillage optimal est généré, de sorte que les maillages construits pour les cubes adjacents se concatènent automatiquement. La résolution du maillage finalement construit pour l'objet est fonction de la finesse de la discrétisation volumique.
- *Ball Pivoting* : méthode qui construit itérativement un maillage en faisant "rouler" une boule virtuelle sur le nuage de points. Initialement, le plus grand triangle contenant trois points voisins du nuage est sélectionné. Une boule est posée sur ce triangle et va pivoter successivement autour de ses trois arêtes. Dès qu'un autre point du nuage est rencontré

durant un tel pivotement, un nouveau triangle est construit. La résolution du maillage final est fonction du rayon de la boule.

- *Surface déformable* : un maillage initial, généralement régulier, est tout d’abord défini par discrétisation d’une sphère (découpage itératif d’un icosaèdre inclus dans une sphère). Ce maillage est placé au centre de gravité du nuage de points à modéliser et est dimensionné soit pour englober le nuage, soit au contraire pour y être inclu. Ensuite, des forces sont appliquées sur les sommets de ce maillage afin de le déformer itérativement jusqu’à ce qu’il colle au mieux aux points de mesure. Il existe des forces externes (les sommets sont attirés par les points de mesure) et des forces internes (entre sommets, afin de garantir une courbure régulière au maillage finalement construit). La résolution du maillage final dépend du nombre de sommets discrétisés sur la sphère initiale. Dans les algorithmes plus complexes, de nouveaux sommets sont rajoutés là où la surface à représenter est la plus complexe.

Mais, dans ce contexte, ces méthodes sont difficilement applicables en temps réel. Typiquement, pour mailler 20000 points, il faut plusieurs secondes de calcul, si ce n’est plus, avec les surfaces déformables. Vu la complexité des méthodes d’approximation surfacique ou de construction de maillage, la construction d’une carte d’occupation volumique est la seule méthode exploitable dans notre contexte : elle est d’ailleurs un préalable pour la construction d’un maillage par l’algorithme des *Marching Cubes*.

Cette méthode d’énumération spatiale consiste à discrétiser le volume que l’on veut décrire (ici le parallélépipède), en un tableau à trois dimensions de voxels (pour éléments de volume). En général, un voxel est un cube de côté D_{voxel} ; la dimension D_{voxel} définit la résolution de la représentation. Cette résolution peut être divisée itérativement par deux pour construire des représentations de plus en plus précises, selon les besoins de la tâche utilisant le modèle de l’obstacle (typiquement, un planificateur de trajectoires). Une carte de voxels (ou *VoxelMap*) découpée itérativement de cette manière est appelée un *octree*.

Pour construire une carte d’occupation de résolution D_{voxel} , il suffit donc de discrétiser le volume à décrire par des cubes de côté D_{voxel} et d’affecter les points perçus dans ce volume aux voxels. Si un voxel contient des points, il est marqué occupé, sinon il est libre. La résolution peut être améliorée en découpant itérativement les voxels occupés jusqu’à obtenir la précision désirée ; ou jusqu’à atteindre le niveau d’incertitude sur les points perçus. Ce découpage itératif peut être exploité, par exemple et avec la prise en compte du véhicule porteur du système, dans le *Collision Checker* du planificateur de trajectoires : s’il n’y a pas d’intersection entre le volume du véhicule et celui de l’obstacle à la résolution D_{voxel} , inutile de poursuivre l’analyse ; s’il y a collision, le test peut être itéré avec le volume de l’obstacle décrit à la résolution $\frac{D_{voxel}}{2}$.

De toute évidence, ces traitements sont une charge pour le système et ne doivent pas être effectués pour chacun des modèles locaux. Par contre, il est possible de profiter de l’instant d’arrêt du véhicule, juste après l’enclenchement de la marche arrière et avant la manœuvre pour affiner l’ensemble du modèle global. Cet affinement ne se produira donc qu’une seule fois. L’inconvénient majeur de cette représentation d’énumération spatiale, est qu’il est difficile de lui appliquer un changement de repère. Dans notre application, cet inconvénient n’a pas d’incidence.

4.5.3 Mesure de la place libre

Une fois acquis un modèle de la voie par fusion des N_{modele} modèles locaux conservés dans la fenêtre glissante, il convient de détecter les places de stationnement et de mesurer leurs dimensions. Pour cette fonction, un très grand nombre de scénarii sont possibles :

- le plus "standard" : place en créneau le long d'un trottoir, entre deux obstacles de type véhicule et un trottoir, qui délimite la chaussée,
- des versions plus ou moins complexes du créneau :
 - place le long d'un trottoir, devant ou derrière un seul obstacle,
 - place le long d'un obstacle continu qui limite la voie (mur, haie, ...)
 - place sans trottoir, mais avec un ou plusieurs obstacles en limite de voie (bornes),
 - place sans trottoir et sans limite de voie (par exemple, place limitée par un marquage au sol, non pris en compte dans nos travaux). En ce cas, la place doit être détectée et une limite virtuelle de la chaussée est positionnée 10cm au-delà des obstacles limitant la place.
- place en épis : en fonction de la disposition des caméras, des difficultés liées aux angles morts nous empêchent de voir la totalité de la place. La place peut être détectée mais sans aucune garantie sur sa profondeur,
- place en bataille : cette configuration peut poser des problèmes en fonction de la configuration des caméras. La place est définie en fonction de sa largeur sans information de profondeur. Il sera important de surveiller l'environnement pendant la manoeuvre pour éviter d'éventuels obstacles cachés.

Une place libre est définie par :

- son type (créneau, bataille, épi),
- les quatres coins en $(X, Y)_i$ (ou plus) définissant la zone libre,
- la largeur L de la place est prise sur une parallèle à la limite de la chaussée (trottoir ou obstacle longeant la chaussée) dans le cas où cette limite a été perçue. Dans le cas contraire, la largeur est prise dans une direction parallèle au bloc définissant l'obstacle limitant la place, le plus proche du véhicule,
- les profondeurs appelées P_{near} et P_{far} sont mesurées également selon des orthogonales à la limite de la chaussée quand elle a été perçue. Sinon, ce sont les dimensions des boîtes limitant la place,
- les incertitudes Inc_i sur la position (X, Y) et sur les dimensions L , P_{near} et P_{far} . Elles sont déduites des incertitudes sur les dimensions des boîtes limitant la place,
- la tolérance t qui est la différence entre la largeur L de la place et la longueur l du véhicule.

Finalement, la décision pour savoir si une place libre peut servir au stationnement, est basée sur le critère suivant :

$$L - \sum_i Inc_i \geq l + t \quad (4.22)$$

4.6 Résultats

Nous présentons dans cette section des résultats sur images de synthèse pour valider nos travaux sur la modélisation de la place de parking. Nous avons vu que la diversité des configurations de parking est très importante et la génération d'images est très longue. Nous verrons donc des résultats beaucoup plus détaillés sur images réelles en chapitre 5. Cette section donnera donc des informations sur la configuration du banc stéréoscopique à utiliser et nous familiarisera avec les interfaces graphiques permettant d'afficher les résultats. Le paragraphe suivant présente

une séquence sur un banc *idéal pour le constructeur*. Nous verrons alors les difficultés et nous proposerons dans le paragraphe suivant une configuration plus adaptée à notre application.

4.6.1 Capteur *idéal* (selon le constructeur du système)

La séquence générée correspond à une configuration de créneau dont les deux obstacles ont respectivement 5 et 6 mètres de longueur. L'espace libre, quant à lui, a une longueur de 6 mètres. Le véhicule simulé porte les caméras au-dessus de sa plaque d'immatriculation arrière, les caméras orientées vers l'arrière. La figure 4.20 présente la séquence vue du dessus.

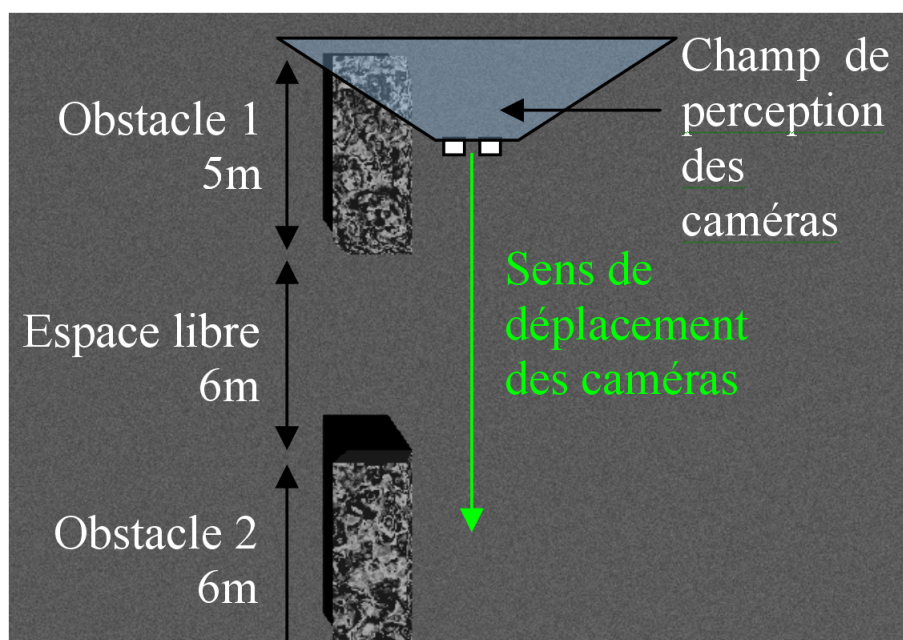


FIG. 4.20 – Plan de la séquence de parking avec images de synthèse

Présentation des caractéristiques du banc de synthèse

Le banc de stéréovision est similaire au banc *STH-MDCS* de la société *VidereDesign*²⁰. Son faible encombrement en fait un capteur idéal pour l'intégration sur un véhicule. Les caractéristiques précises du banc sont :

- distance inter-caméras de 9 cm,
- capteur de dimension 7.56×5.76 mm,
- résolution de 1280×960 pixels,
- focale de 3.5 mm.

Ces optiques et ces capteurs permettent un champ de perception très large : 94.4° pour chacune des caméras. Par contre, la plage de disparité (pour une résolution d'image de 320×240) est faible (cf. figure 4.21) et ne permet pas de voir à plus de 4 mètres. Par exemple, la disparité pour 2 mètres est de 6.66 pixels, de 4.44 pour 3 mètres et n'est plus que de 3.33 pixels pour 4 mètres. Même si ce capteur répond à beaucoup de contraintes (coûts, encombrement, champ de perception), il ne semble pas propice à la stéréovision pour cette application. De plus, ici les

²⁰www.videredesign.com

aberrations géométriques ne sont pas simulées, aberrations qui se sont révélées très difficiles à corriger sur le banc réel.

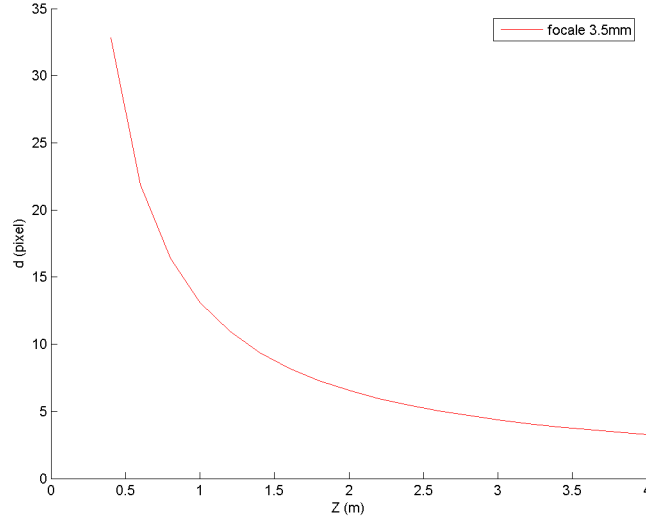


FIG. 4.21 – Courbe de la valeur de la disparité en fonction de la distance

Résultats de modélisation de la voie

Cette section donne les résultats graphiques de la détection et de la fusion des obstacles. Dans un premier temps nous verrons les résultats des pixels détectés comme obstacle sur l'image gauche de la paire stéréo, pixels colorisés si étiquetés comme obstacle. A une couleur correspond un unique obstacle, l'attribution des couleurs étant aléatoire.

Dans un deuxième temps nous verrons les résultats des boîtes englobantes fusionnées vues du dessus. Les rectangles pleins correspondent à des obstacles détectés, le rectangle bleu correspond au véhicule porteur du système et le point vert représente le repère $\mathcal{R}_{voiture}$ du véhicule. La fenêtre glissante étant centrée sur ce repère $\mathcal{R}_{voiture}$, la voiture ne semble pas bouger, par contre les obstacles perçus s'éloignent avec l'avancement du véhicule. La figure 4.22 présente la détection du premier obstacle. Seuls les pixels avant élargissement des boîtes englobantes sont affichés, ce qui explique cette limite franche entre le bas et la partie supérieure des obstacles. En figure 4.23, l'espace libre ne déclenche pas de fausse détection malgré la faible qualité du nuage 3D généré par la stéréovision. En effet, si on regarde une coupe du sol, on voit une "épaisseur" de celui-ci d'environ 20cm, épaisseur relative à la variance des mesures sur les points 3D pour ce banc de stéréovision. La figure 4.24 montre la détection du deuxième obstacle.

Intéressons nous maintenant aux dimensions des boîtes englobantes générées et à leurs fusions. La figure 4.25 donne la vue de dessus des détections d'obstacles de la fenêtre glissante relative à la figure 4.22. Nous pouvons remarquer l'épaisseur de la boîte (en y) alors que nous n'avons pas encore observé un coté du véhicule nous permettant de la calculer. Cette épaisseur correspond seulement à la variance des points 3D sur le flanc de l'obstacle. La grille de $1m \times 1m$ nous donne un ordre de grandeur. Au fur et à mesure du passage devant la place de parking,

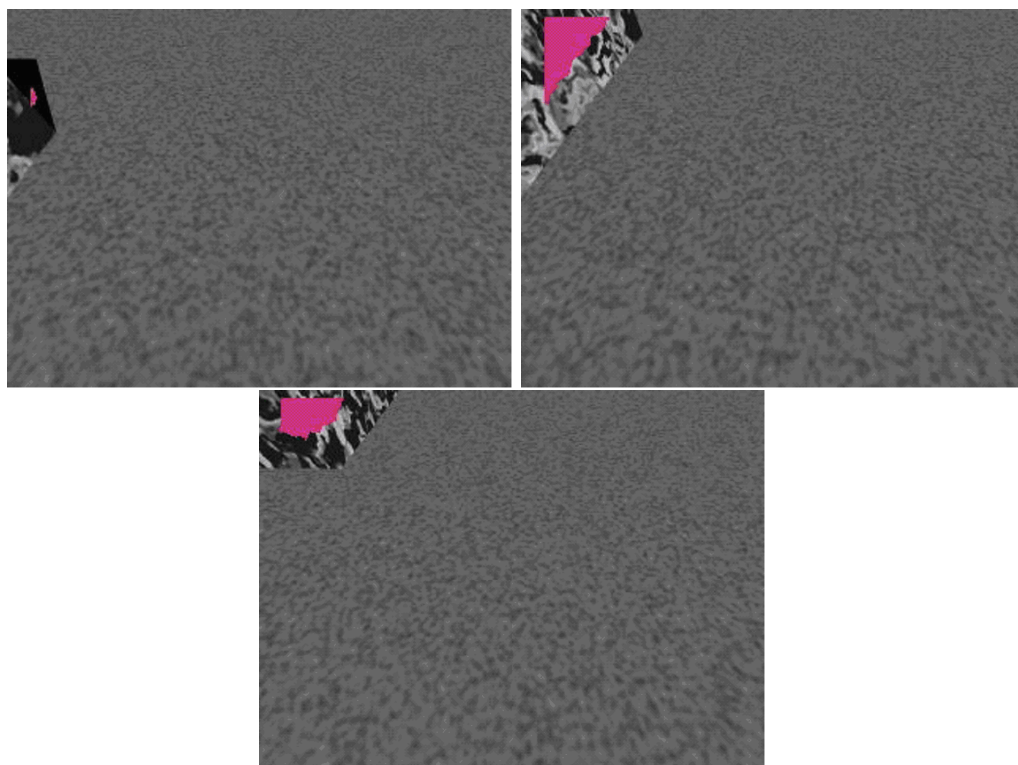


FIG. 4.22 – Détection du premier obstacle

nous voyons en figure 4.26 et 4.27 le modèle de voie se construire incrémentalement.

A la fin de la séquence, nous pouvons mesurer une erreur de 0.32 mètre sur le premier obstacle et 0.44 mètre sur l'espace libre (le deuxième obstacle n'étant pas perçu entièrement, il n'est pas possible de donner une erreur sur sa mesure). De toute évidence, nous devons relâcher des contraintes sur le banc de stéréovision si nous souhaitons améliorer les performances de la détection. Le chapitre 5 présentera des résultats sur images réelles avec une configuration du banc satisfaisante.

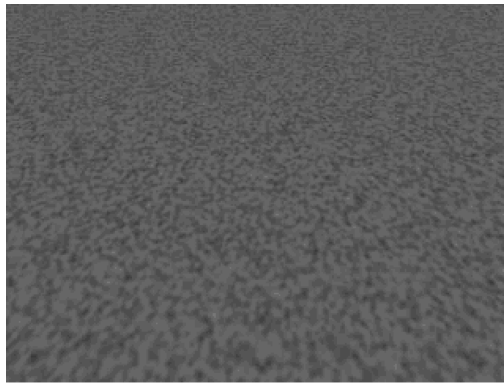


FIG. 4.23 – Détection de l'espace libre

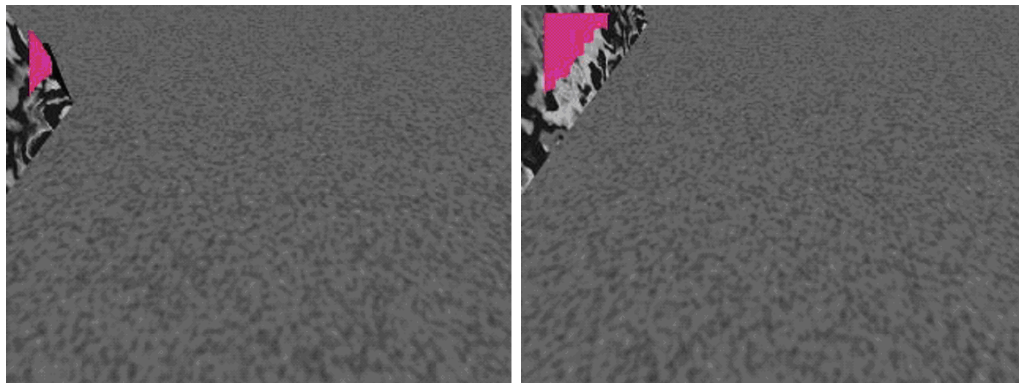


FIG. 4.24 – Détection du deuxième obstacle

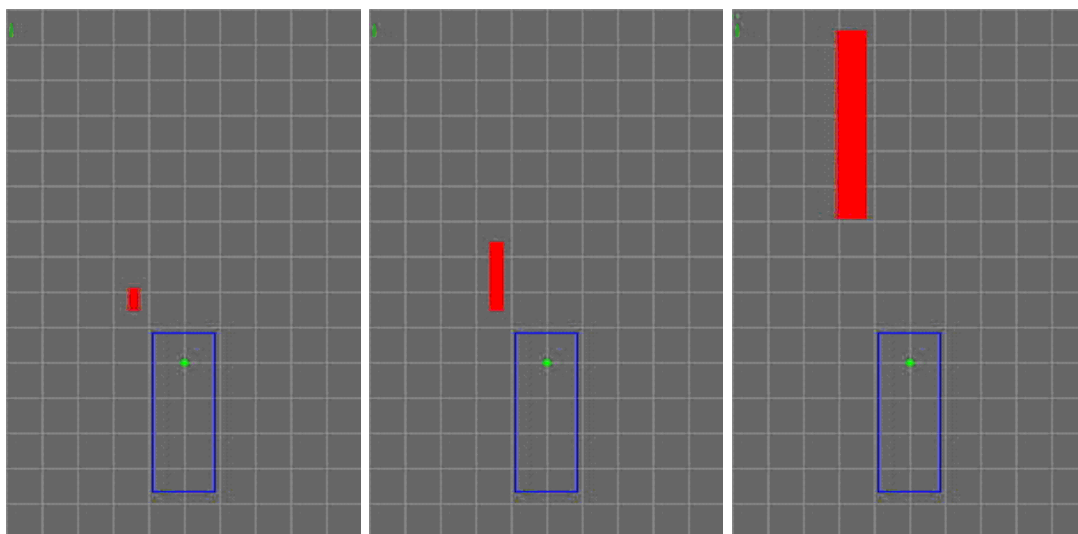


FIG. 4.25 – Résultat de la détection du premier obstacle (vue du dessus)

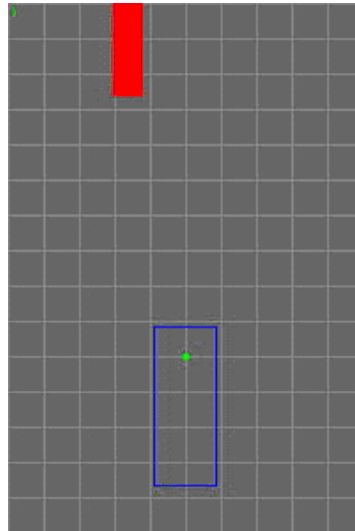


FIG. 4.26 – Résultat de la détection de l'espace libre (vue du dessus)

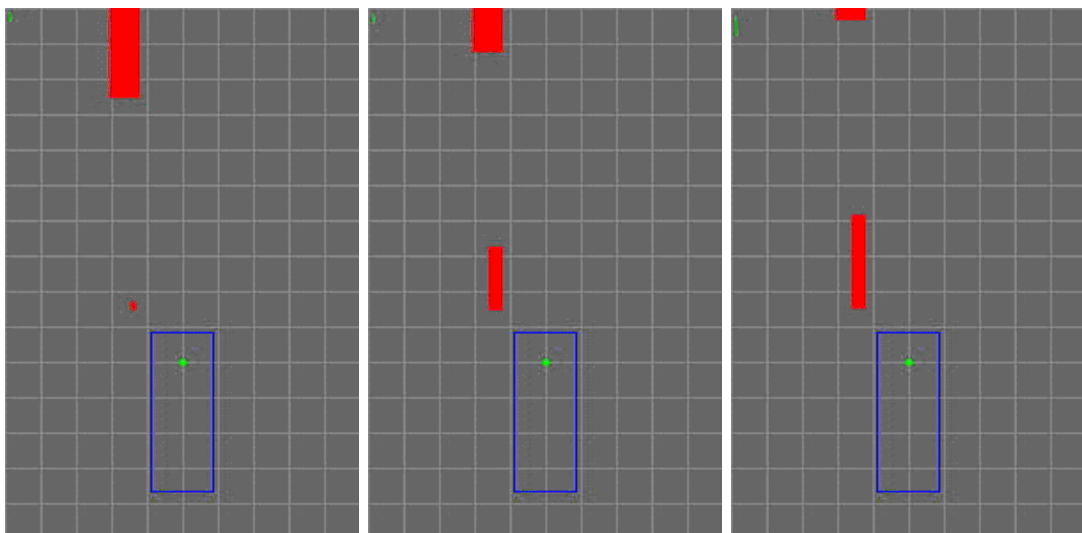


FIG. 4.27 – Résultat de la détection du deuxième obstacle (vue du dessus)

4.6.2 Capteur idéal (selon le développeur)

Présentation des caractéristiques du banc de synthèse

Nous avons vu qu'il est très difficile de modéliser la place avec un capteur dont les caractéristiques (encombrement, champ de vue...) doivent satisfaire des contraintes trop importantes. Relâcher une contrainte signifie soit changer de caméra (utilisation d'une focale plus grande ou d'un capteur CCD (ou CMOS) plus petit), soit jouer sur la résolution de traitement de la stéréovision, soit modifier la base inter-caméras. C'est ce dernier paramètre que nous allons modifier ici pour utiliser une base de 40cm au lieu des 9cm précédents. Le gain en terme de plage de disparité n'est pas négligeable au vu de la courbe de la figure 4.28. En effet, à 2m la disparité est de 30.47pixels où elle n'était que de 6.66pixels pour la base de 9cm et de 15.23pixels à une distance de 4m au lieu de 3.33pixels .

La séquence utilisée est similaire à la séquence précédente avec deux obstacles séparés d'un espace libre.

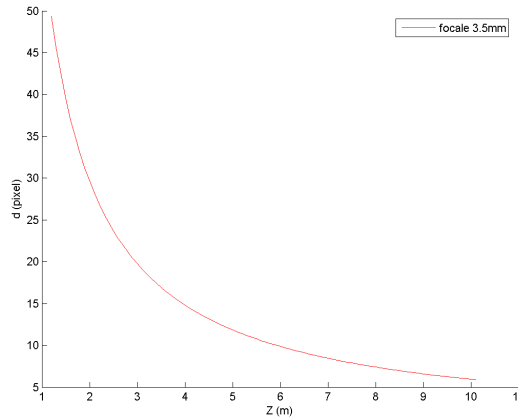


FIG. 4.28 – Disparité en fonction de la distance pour un banc de 40cm de base.

Résultats de modélisation de la voie

La figure 4.29 présente un extrait des résultats sur la nouvelle séquence de synthèse. Comme il était prévisible, le gain est important. Nous verrons, dans le chapitre 5, quelles solutions sont envisageables à la réalisation d'un système de modélisation de place de parking.

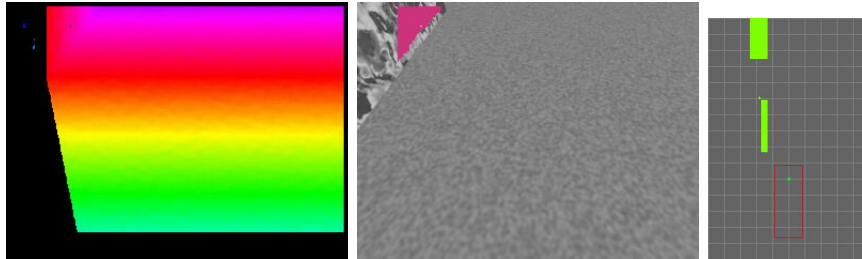


FIG. 4.29 – Image de disparité d'une paire stéréo [gauche], détection des points obstacles [milieu] et modèle d'obstacle vue de dessus [droite].

Les résultats présentent des erreurs de :

- 8cm sur le premier obstacle,
- et 15cm sur l'espace libre.

4.7 Conclusion

Nous avons présenté dans ce chapitre un algorithme original permettant la modélisation d'une place de parking dans de nombreuses configurations. L'approche *3D* que nous avons adoptée permet la discrimination des obstacles dans un environnement proche du véhicule. A partir d'une représentation géométrique de ces obstacles, nous avons présenté une méthode de construction de l'espace libre disponible pour le parking.

Tenant compte des contraintes de temps de calcul, nous avons proposé une approche simple mais pertinente vis-à-vis de l'application. Le système final, de la stéréovision à la détection d'obstacles, fonctionne à 10Hz sur un ordinateur portable pentium *IV* centrino 1.7GHz. Le point difficile de cette étude est la qualité des données 3D issues du capteur. La stéréovision bas coût rend la tâche difficile même si des configurations existent mais à un prix plus élevé. Nous verrons dans le chapitre 5 les choix que nous avons faits afin d'améliorer les performances du système et une proposition de configuration capteur bas coût.

De plus, des recherches de nouveaux capteurs sont menées, notamment concernant des capteurs *3D* à temps de vol optique. Privé du temps nécessaire à la stéréovision, le système atteint des fréquences supérieures à 30Hz largement suffisantes à un système d'aide au parking. Les premiers résultats montrent une faisabilité du système malgré quelques difficultés liées au temps d'intégration des photosites qui tend à déformer les obstacles au-delà d'une vitesse de 2 ou 3km.h⁻¹. Des problèmes persistent sous de fortes illuminations, problèmes aussi présents avec la stéréovision. Les premières versions du capteur restent des prototypes et nous pouvons espérer des améliorations des performances pour les versions suivantes.

Bibliographie du chapitre

- [Betge-Brezetz, 1996] S. Betge-Brezetz (1996). *Modelisation Incrementale et Localisation par Amer pour la Navigation d'un Robot Mobile Autonome en Environnement Naturel*. Thèse de doctorat, Université Paul Sabatier.
- [Driss et al., 1994] A. Driss, V. Rodrigues, et P. Cohens (1994). Parking a vision-guided automobile vehicle. Dans *IEEE Conference on Control Applications*, volume 1, pages 59 – 64.
- [Fabiani, 1996] P. Fabiani (1996). *Représentation dynamique de l'incertain et stratégie de prise d'information pour un système autonome en environnement évolutif*. Thèse de doctorat, Ecole Nationale Supérieure de l'Aéronautique et de l'Espace.
- [Kaempchen et al., 2002] N. Kaempchen, U. Franke, et R. Ott (2002). Stereo vision based pose estimation of parking lots using 3d vehicle models. Dans *Intelligent Vehicle Symposium*, volume 2, pages 459 – 464.
- [Kondak et al., 2001] K. Kondak et G. Hommel (2001). Computation of time optimal movements for autonomous parking of non-holonomic mobile platforms. Dans *IEEE International Conference on Robotics and Automation*, volume 3, pages 2698 – 2703.
- [Krebs, 2000] B. Krebs (2000). *Probabilistische erkennung von 3d freiformobjekten mit bayesischen netzen*. Thèse de doctorat, Technische Universität Braunschweig.
- [Lemondé, 2002] V. Lemondé (2002). Exploitation de la stéréovision passive sur un véhicule intelligent. Rapport technique, Rapport de fin d'étude INSA Toulouse Génie Informatique et DEA Systèmes Informatiques (EDSYS).
- [Nguyen, 2003] L. Nguyen (2003). Stéréovision pour l'aide au stationnement de véhicules intelligents. Rapport technique, Rapport de fin d'étude INSA Toulouse Temps Réel et Système et DEA Systèmes Informatiques (EDSYS).
- [Ono et al., 2002] S. Ono, M. Kagesawa, et K. Ikeuchi (2002). A probe car for parking-vehicle detection by using laser range sensor. Dans *Intelligent Vehicle Symposium*, volume 2, pages 322 – 327.
- [Simeon et al., 2001] T. Simeon, J. Laumond, et F. Lamiroux (2001). Move3d : A generic platform for path planning. Dans *IEEE International Symposium on Assembly and Task Planning*, pages 25 – 30.
- [Specht, 2005] J. R. Specht (2005). *Modélisation d'objets 3D par construction incrémentale d'un maillage triangulaire, dans un contexte robotique*. Thèse de doctorat, Université Paul Sabatier, Toulouse III.
- [Suski, 2001] D. Suski (2001). *3d object detection in traffic scenes using stereoscopic vision*. Thèse de doctorat, Fachhochschule.
- [Wada et al., 2003] M. Wada, K. S. Yoon, et H. Hashimoto (2003). Development of advanced parking assistance system. Dans *IEEE Transactions on Industrial Electronics*, volume 50, pages 4 – 17.

- [Walsh et al., 1994] G. Walsh, D. Tilbury, S. Sastry, R. Murray, et J. Laumond (1994). Stabilization of trajectories for systems with nonholonomic constraints. Dans *IEEE Transactions on Automatic Control*, volume 39, pages 216 – 222.
- [Xu et al., 2000] J. Xu, G. Chen, et M. Xie (2000). Vision-guided automatic parking for smart car. Dans *Intelligent Vehicles Symposium*, pages 725 – 730.

Chapitre 5

Expérimentations

Sommaire

5.1	Plateformes expérimentales	130
5.1.1	Véhicule expérimental	130
5.1.2	ATRV	132
5.1.3	Véhicule ARCOS	133
5.2	Recalibration d'un système stéréoscopique	133
5.2.1	Evaluation et recalibration sur des images d'intérieur	133
5.2.2	Recalibration sur véhicule	135
5.2.3	Conclusion	138
5.3	Estimation de l'odométrie optique	139
5.4	Détection d'obstacles en milieu routier	139
5.5	Modélisation de la place de parking	145
5.5.1	Banc de stéréovision sur l'axe central, à l'arrière	148
5.5.2	Banc stéréo sur le côté, à l'avant	148
5.5.3	Conclusion	148

Ce chapitre complète les résultats sur images de synthèse, présentés jusqu'à présent. La première section est consacrée à la description des moyens expérimentaux mis en place pour évaluer la faisabilité des algorithmes et méthodes décrits dans cette thèse. Le véhicule expérimental du LAAS est brièvement présenté bien que les résultats n'aient pas utilisé cette plate-forme. Pour l'instant, les difficultés de mise en œuvre et de rééquipement de ce démonstrateur, nous ont contraint à n'utiliser que les bases d'images issues du projet ARCOS.

La structure de ce chapitre s'appuie sur la structure même de cette thèse. Nous verrons donc tout d'abord les résultats de recalibration, méthode décrite en chapitre 2. Ensuite, les résultats du chapitre 3 de détection d'obstacles en milieu routier seront présentés avant de terminer par les résultats du chapitre 4 de modélisation de la place de parking.

5.1 Plateformes expérimentales

5.1.1 Véhicule expérimental



FIG. 5.1 – Plate-forme expérimentale du LAAS-CNRS

Le véhicule expérimental du LAAS-CNRS est une Renault Safrane (cf. figure 5.1) issue du projet *Prometheus* (1988-1995) traitant du développement de véhicules expérimentaux pour la fusion multi-sensorielle. Dépourvu de CAN²¹, le véhicule a récemment été équipé d'un odomètre, d'un capteur d'angle au volant et d'un capteur de vitesse, reliés à un réseau ethernet permettant l'interaction des capteurs avec les applications embarquées (cf. figure 5.3). En plus de ces capteurs proprioceptifs, un banc de stéréovision est directement relié par bus *IEEE 1394* à l'ordinateur de bord.

L'emplacement initial de ces caméras avait été prévu au niveau des phares avant du véhicule. La configuration obtenue donne une distance inter-caméra de 1.30 m à une hauteur de 70 cm du sol. Avec une focale de 8 mm, le système se rapproche des spécifications ARCOS. De plus, à la vue de la courbe de disparité présentée en figure 5.2, cette configuration est très satisfaisante pour la stéréovision en présentant une disparité de près de 25 pixels à 45 m de distance. L'encombrement minimal et la facilité d'intégration des caméras rend d'autant plus cette configuration attrayante. Par contre, plusieurs problèmes se sont posés à nos systèmes embarqués : (1) la détection de la route est rendue difficile vue la position des caméras très basse et sensiblement du

²¹Controller Area Network

côté gauche de la voie, et (2) la stéréovision corrèle difficilement vu le peu de texture de la route à l'avant du véhicule, surtout à grande vitesse. Associé aux problèmes de nettoyage des optiques des phares et de *pollution* lumineuse la nuit, l'intégration finale des caméras est revenue à une position plus classique, à l'intérieur de l'habitacle. Le système embarqué est plus performant au détriment d'une intégration moins attirante industriellement (soumis aux chocs des passagers). La bonne solution, non envisageable sur un prototype du LAAS-CNRS, serait d'intégrer chacune des caméras aux montants latéraux du pare-brise dont l'emplacement est déjà fortement utilisé pour le passage de câble.

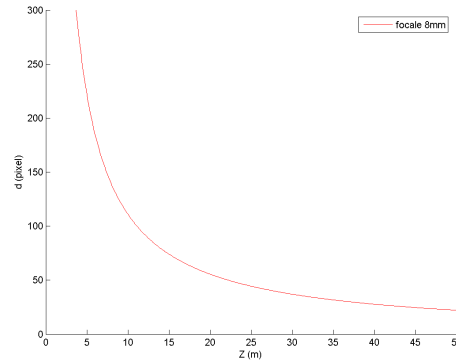


FIG. 5.2 – Courbe de disparité en fonction de la distance pour une base de 1.30 m, une résolution de 512×384 et une focale de 8 mm.

La gestion des flux capteurs est réalisée par le logiciel *RT-Maps* distribué par la société *Intempora*²² ; initialement, ce logiciel a été développé à l'Ecole des Mines de Paris, pour effectuer des enregistrements synchronisés de séquences données pour des applications multi-capteurs temps réel, et pour rejouer ces séquences (fonction *Digital Recorder/Player*) ; mais rapidement RT-MAPS est devenu une plate-forme d'acquisition de données multi-capteurs, modulaire et évolutive pour le prototypage rapide d'applications temps-réel. Par le biais du SDK²³, il est possible d'écrire les modules propres aux applications (un module de stéréovision, un module de modélisation de place de parking, un module de détection d'obstacles, ...) pour lesquels le *Studio* permet de créer des scripts à la façon de *Simulink* pour relier et faire interagir les modules entre eux. RT-MAPS est l'environnement de développement et d'exécution utilisés par un grand nombre des acteurs du monde automobile français ainsi que par des projets français et européens (ARCOS, CARSENSE, REACT, PUVAME).

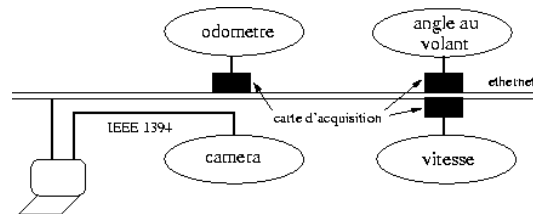


FIG. 5.3 – Architecture du système embarqué

²²www.intempora.com

²³Software Development Kit

5.1.2 ATRV



FIG. 5.4 – Dala

L'ATRV²⁴ *Dala* de son prénom, est un robot d'extérieur de la société *iRobot*²⁵ monté sur quatre roues équipées de pneus. C'est un rover de type *skid-steering*, tournant sur lui-même par différentiel de vitesse sur ses roues droites et gauches. Son châssis permet des déplacements en milieu accidenté mais sans grandes amplitudes. Ses caractéristiques détaillées sont décrites dans le tableau ci-dessous :

Motricité	4 roues motrices
Vitesse maximale	2 m/s
Batteries	24V, 1440 Wh
Bus	PCI
Processeur	1 Intel Pentium IV
Système d'exploitation	Linux (noyau 2.6.10)
Communication	IEEE802.11b
Capteurs	1 odomètre 12 capteurs ultra-sons 2 caméras IEEE1394 1 caméra panoramique
Dimensions	105cm × 80cm × 100cm
Poids	120 kg

²⁴All TerRain Vehicle²⁵www.irobot.com

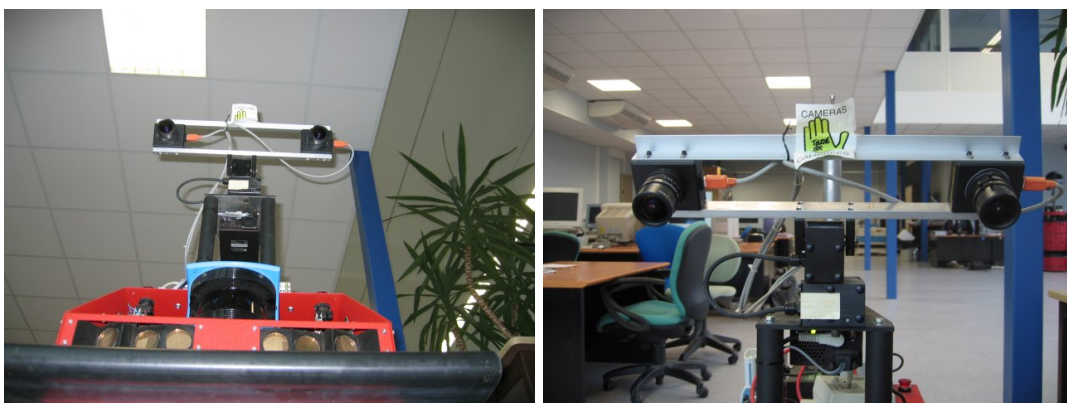


FIG. 5.5 – Dala (face avant)

Le banc de stéréovision embarqué dispose d'une base de 33 cm ; chacune des caméras est équipée de matrice $1/3$ de pouce et de focale 3.5 mm . Les champs de perception horizontal et vertical résultants sont respectivement de 70° et 55° .

5.1.3 Véhicule ARCOS

Le véhicule dénommé VIPER²⁶ utilisé pour les acquisitions du projet ARCOS est un véhicule expérimental du LIVIC²⁷, laboratoire de l'INRETS²⁸. VIPER est dédié à la perception de l'environnement grâce à ses capteurs proprioceptifs (centrale inertielle, odomètre et GPS) et extéroceptifs (télémètres laser à balayage, trois caméras synchronisables et un radar). L'équipement détaillé est présenté dans [Labayrade 2004].

5.2 Recalibration d'un système stéréoscopique

Les résultats de recalibration sont présentés pour le cas d'un robot mobile sur des images d'intérieur puis dans le cas d'un système de vision embarqué sur un véhicule grâce aux images ARCOS.

Dans un premier temps, il est intéressant de regarder le comportement du score d'évaluation sur des images réelles. La figure 5.6 présente les résultats d'évaluation de la calibration sur une même séquence en utilisant (1) les paramètres donnés par calibration avec mire (points rouges) et (2) des paramètres grossiers (croix bleues) donnés par simples mesures à l'aide d'une règle et de la matrice identité pour l'orientation des caméras. Dans le premier cas, l'évolution du score reste centré sur 0.2 avec un écart type de 0.05 alors que le score moyen du banc non calibré a pour moyenne 1.2 et écart type 0.15.

5.2.1 Evaluation et recalibration sur des images d'intérieur

Les résultats présentés dans cette section sont basés sur une séquence de 50 images prises dans un bureau dont une des paires stéréoscopiques est donnée en figure 5.7. La séquence entière

²⁶Véhicule Instrumenté de Perception de l'Environnement Routier

²⁷Laboratoire sur les Interactions Véhicules-Infrastructure-Conducteurs

²⁸Institut National de REcherche sur les Transports et leur Sécurité

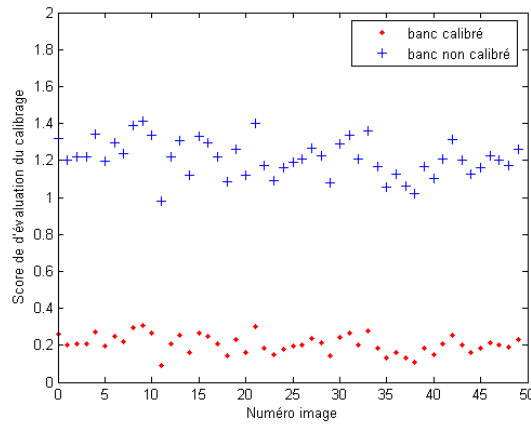


FIG. 5.6 – Evaluation de la qualité de calibration d'un banc calibré (points rouges) et non calibré (croix bleues)

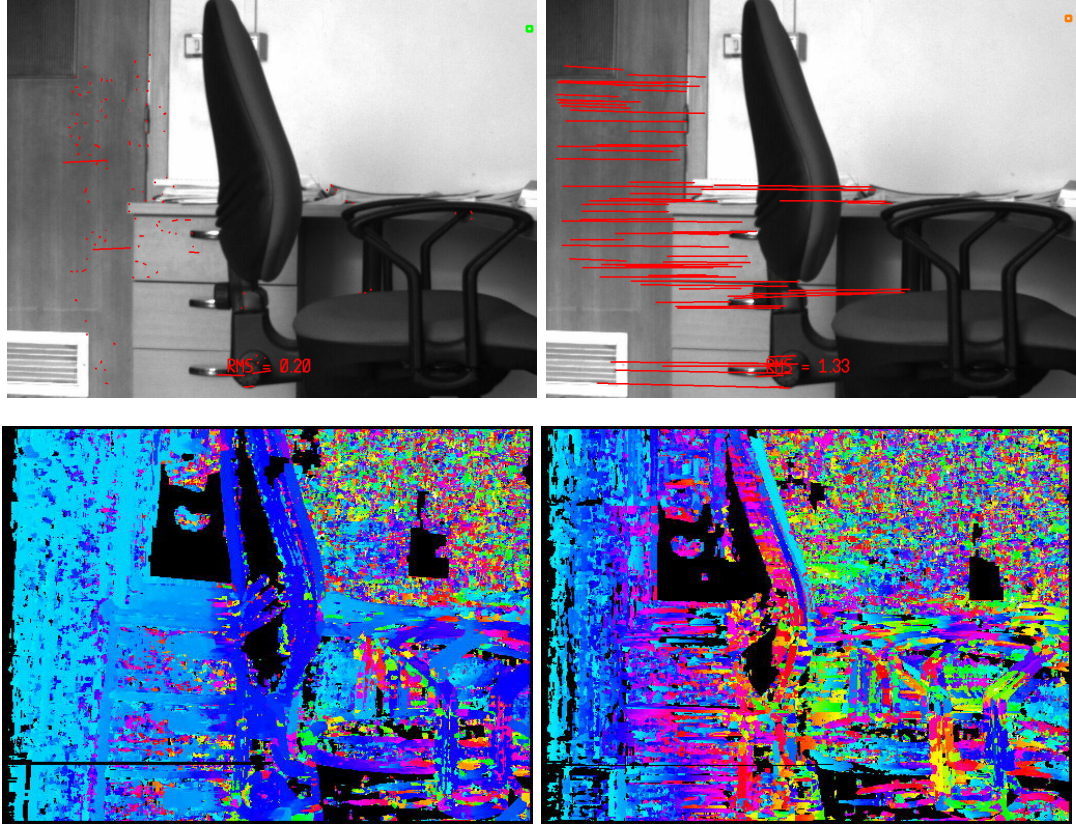


FIG. 5.7 – Paire stéréoscopique utilisée pour la recalibration

a été traitée avec des paramètres de calibration données par défaut à l'aide des paramètres intrinsèques théoriques et d'une transformation inter-caméra estimée à l'aide d'un mètre et l'orientation relative des caméras données à l'identité. Un exemple de carte de disparité obtenue dans ces conditions est présenté en figure 5.8[droite] avec son évaluation de la calibration. La carte de disparité est fortement bruitée et seuls certains contours très marqués de l'image peuvent se distinguer. Le score d'évaluation de la calibration est élevé avec une valeur de 1.33.

La figure 5.8[gauche] présente la même image mais après recalibration du système sur la base des estimées utilisées pour le test précédent. Le score d'évaluation de la calibration est sensiblement plus faible avec une valeur de 0.2. La carte de disparité est encore très bruitée mais ce bruit est limité aux zones uniformes dans l'image. Comme l'image de disparité n'est pas filtrée pour garder un maximum d'information lors de l'évaluation, ce bruit est conservé. Par contre, une fois filtrée, la carte de disparité donnée en figure 5.9[haut] est beaucoup plus cohérente avec la structure de l'image.

Les résultats sur la séquence complète sont présentés en figure 5.9[bas]. Les croix bleues représentent les scores d'évaluation de calibration sans utiliser la recalibration. Ces scores ont pour moyenne 1.31 et écart type 0.035 et permettent de diagnostiquer la décalibration (ici le



(a) Disparité et évaluation après recalibration

(b) Disparité et évaluation sans recalibration

FIG. 5.8 – Effet de la recalibration sur les résultats de la stéréocorrélation

manque de calibration) du système. Les points rouges, obtenus après recalibration, montrent le gain apporté par notre méthode avec une moyenne de 0.4 et un écart type de 0.2. Intéressons nous aux quatre premiers points de la séquence. Le premier score, issu de la première image est identique au cas de la non calibration. C'est cette image qui va déclencher la recalibration sur l'image suivante ce qui explique l'égalité de ce deuxième score (point rouge sur croix bleue). Par contre, le troisième score encore élevé est issu de l'échec de la recalibration. Cet échec détecté, les paramètres précédents de calibration sont rechargés pour l'image suivante qui va servir de base à la réestimation. Le quatrième score est donc lui aussi équivalent au système non calibré. Par contre, cette dernière réestimation est correcte et est conservée pour la suite de la séquence.

5.2.2 Recalibration sur véhicule

Les résultats présentés dans cette section sont issus des séquences *20030403_092913* et *20030130_142121* de la base de données ARCOS. Le gain apporté par la recalibration du système stéréoscopique est beaucoup moins visible que pour les séquences d'intérieur. La figure 5.10 présente le cas de la recalibration sur cette deuxième séquence. Sans recalibration la moyenne des scores est de 0.4 contre 0.3 avec recalibration. Par contre, l'écart type sans recalibration est de 0.05 contre 0.1 avec recalibration. Le gain en terme de carte de disparité n'est pas visible, le pourcentage de points corrélés est de 30% dans les deux cas. De plus, un grand nombre de

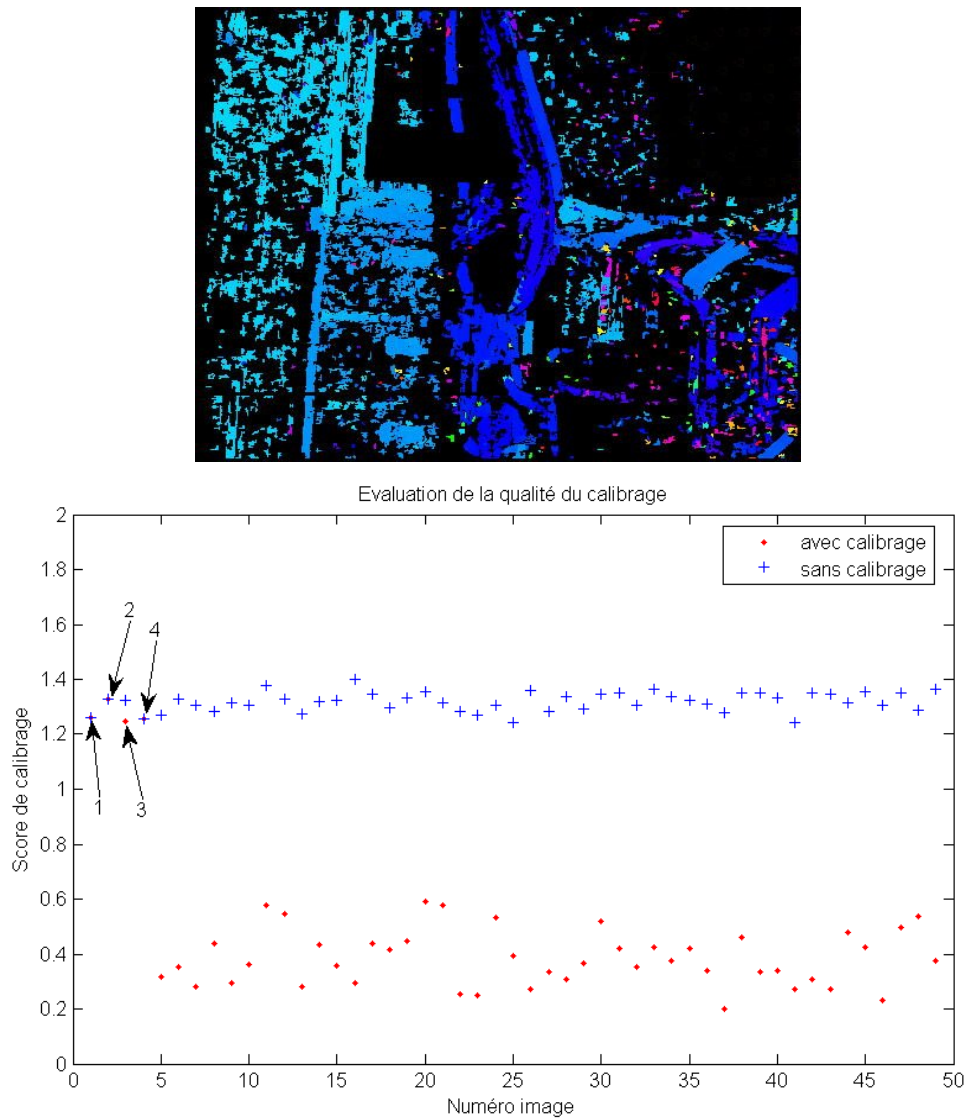


FIG. 5.9 – Carte de disparité après filtrage et courbes de résultats d'évaluation de calibration. La détection de la mauvaise calibration s'effectue sur la première image notée **1**, l'image **2** sert à la recalibration, l'image **3** détecte un échec de recalibration et l'image **4** recalibre correctement le système.

séquences ARCOS aboutissent à des échecs complets de recalibration avec une grosse dérive des paramètres de calibration.

Plusieurs phénomènes peuvent expliquer ces résultats :

- les points d'intérêt appariés sont, pour la plupart, répartis au loin dans la scène (sur les nuages, les arbres lointains, le paysage). La triangulation effectuée sur ces points est alors fortement bruitée et conduit à une mauvaise réestimation.
- la réestimation des paramètres intrinsèques n'étant pas encore implémentée, le bruit occasionné sur le modèle de caméra est trop fort et perturbe la réestimation.

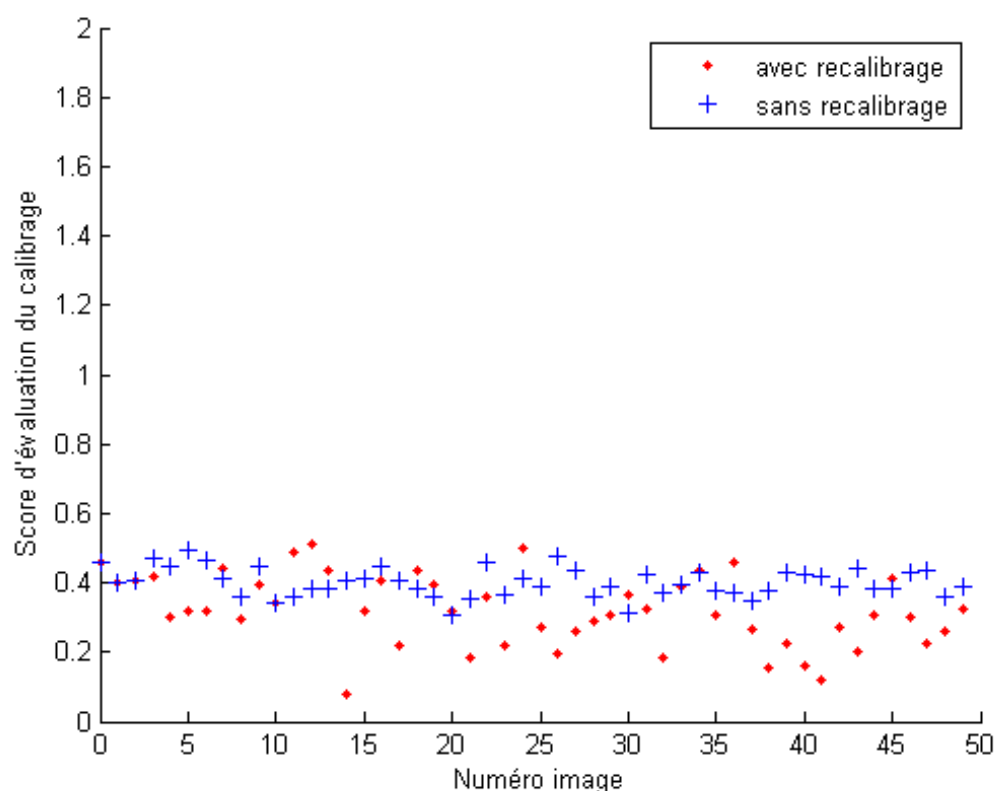


FIG. 5.10 – Recalibration sur une séquence routière ARCOS

Afin de valider notre première explication, nous avons cherché une séquence présentant des points d'intérêt relativement proches de la caméra. Pour cela, nous avons utilisé les obstacles à l'avant de notre véhicule sur la séquence *20030403_092913*. Jusqu'à l'apparition de la voiture dans les images les scores sont identiques à ceux obtenus sans recalibration (cf. figure 5.11). Lorsque des points d'intérêt proches sont détectés (à environ 13 m), la recalibration est activée. Contrairement aux échecs rencontrés sur cette même séquence dans les images précédentes, la recalibration converge et donne des résultats légèrement meilleurs (moyenne de 0.3 contre 0.4).

Afin de tester l'hypothèse sur l'estimation des paramètres intrinsèques, nous avons recalibré une même séquence avec la connaissance des intrinsèques dans un cas et sans cette connaissance dans le deuxième cas. Les paramètres intrinsèques des séquences ARCOS étant inconnus nous avons réalisé ces tests sur les images d'intérieur présentées en section précédente. Toutes les recalibrations effectuées sans la connaissance des intrinsèques ont échoué alors que les recalibrations avec cette connaissance ont abouti à une meilleure estimation des paramètres de calibration

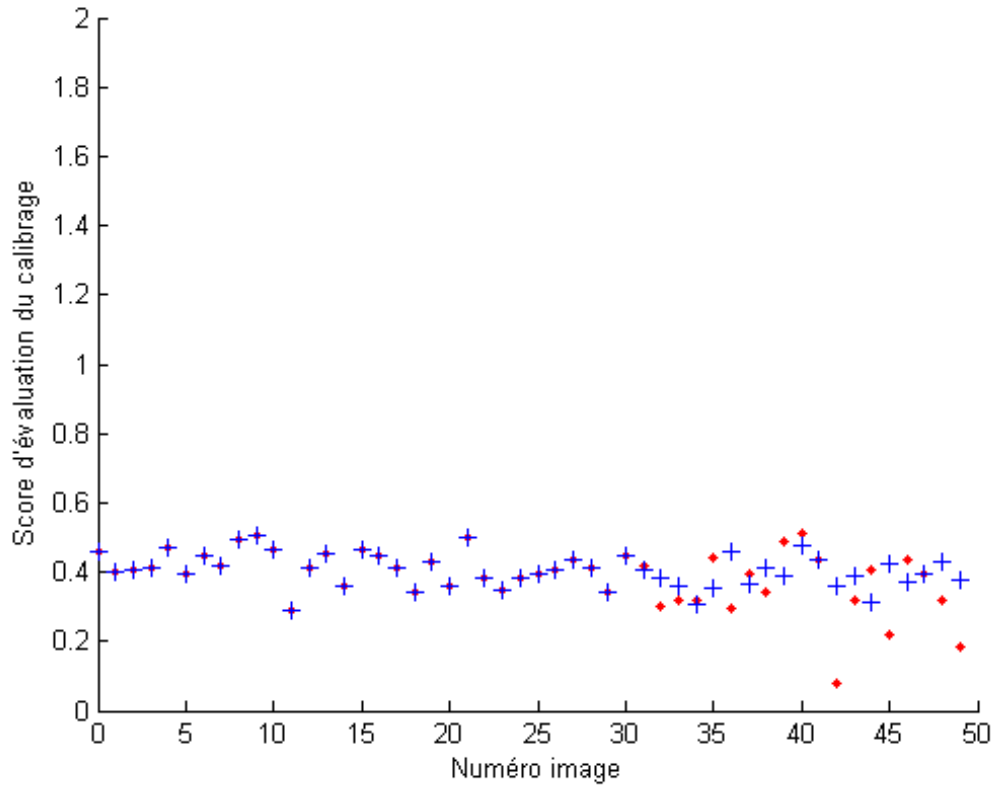


FIG. 5.11 – Recalibration sur une séquence routière ARCOS grâce à un véhicule proche

(meilleure, au sens de notre critère d'évaluation).

5.2.3 Conclusion

Les résultats obtenus pour notre méthode de calibration sont encourageants et montrent qu'il serait intéressant de poursuivre dans cette voie. Le système semble rapidement embarquable sur des robots mobiles dont la configuration des bancs stéréoscopiques est plus favorable au système (distance inter-caméras faible et inclinaison du banc limitant la portée de détection des caméras). Pour garantir une facilité d'utilisation de la stéréovision il serait aussi très intéressant de valider cette méthode, avec estimation des paramètres intrinsèques afin de s'affranchir de toute utilisation de mires ; nous travaillons sur cette validation.

En plus du confort, la réestimation en ligne des paramètres intrinsèques semble aussi être la réponse au bon fonctionnement de la recalibration d'un capteur stéréoscopique monté sur un véhicule. En effet, dans l'état actuel de nos résultats, du fait de la difficulté de trouver des points de calibration proches sur les séquences ARCOS, nos résultats sont mitigés. Nous avons aussi essayé de traiter la recalibration avec les points issus des images présentées en figure 5.13 : dans cette séquence, vu la texture non uniforme du revêtement de la chaussée, il existe de très nombreux points de contrôle appariés sur le sol ; le problème est que la méthode diverge si trop de points sont coplanaires, ce qui est ici le cas.

Les performances du système restent correctes avec une fréquence de 10 Hz pour l'évaluation de la qualité de la calibration et une fréquence de recalibration allant de 5 à 20 Hz, fonction du nombre de points d'intérêt détectés, sur un ordinateur portable 1.7 GHz. Ces fréquences de fonctionnement sont grandement limitées par l'appariement des points d'intérêt qui ne peut dépasser les 10 Hz pour une résolution d'image de 320×240 pixels. Le fait de limiter cet appariement de points à l'espace route pourrait améliorer d'un facteur 4 cette fréquence.

5.3 Estimation de l'odométrie optique

L'odométrie optique est testée sur une séquence de parking réalisée à l'aide du robot Dala. La figure 5.13 montre quelques images sur les 500 qui composent la séquence. Chaque croix rouge correspond à un point apparié avec un point de l'image précédemment acquise. Ces couples de points sont utilisés pour l'estimation du mouvement entre deux vues successives (en exploitant ici la stéréovision, qui doit être bien calibrée).

Les résultats présentés en figure 5.12 montrent une équivalence des deux méthodes d'odométrie : il s'agit de l'intégration des mesures du déplacement du robot, soit par odométrie mécanique (en rouge, en bas), soit en odométrie optique (en bleu, en haut). La position de départ est $(0, 0)$; comme le montrent les deux courbes rouge et bleu, le chemin réalisé par le robot lors de sa phase de recherche de place de parking, est une ligne quasiment droite (variation de 12 m en X , de moins de 20 cm en Y). Il est à noter une différence finale de 4.8 cm sur X et 2.4 cm sur Y . Dala n'étant pas équipé d'un GPS, il n'est pas encore possible de comparer ces résultats avec la vérité terrain. L'odométrie mécanique étant surtout défaillante pour des dérapages occasionnés lors des virages, en ce cas, l'avantage du tout optique n'est pas évident.

Dans nos travaux, l'odométrie optique étant utilisée pour l'application parking, l'intérêt n'est pas dans la précision des mesures de déplacement pendant la recherche de la place, mais réside surtout dans la synchronisation des données odométriques avec les images acquises servant à la modélisation de la place de parking. Avec l'odométrie optique, il n'est pas nécessaire de gérer les erreurs dues aux fréquences de traitement différentes entre le système de vision (environ 10 Hz) et le système de locomotion (environ 100 Hz sur un CAN). Rappelons aussi que l'odométrie mécanique donnera des résultats aléatoires pendant la manoeuvre de créneau, vu la faible vitesse du véhicule.

5.4 Détection d'obstacles en milieu routier

Les résultats qui suivent sont obtenus avec les images de la base de données ARCOS, images acquises pour valider la fonction dite de *Prévention des collisions*. Le système de stéréovision embarqué sur le véhicule du LIVIC répond aux spécifications élaborées lors de ce projet, c'est à dire :

- le plus petit objet à percevoir est un piéton de 0.3m de large et 1m de haut au minimum,
- la distance maximale de perception est de 43.2m,
- la distance minimale de perception est de 6m,
- un espace latéral de 0.5m entre deux objets à 45m doit pouvoir être distingué,
- un espace en profondeur de 2.5m doit être distingué entre deux objets à 45m,
- le champ de vue du capteur doit être d'au moins 31° (largeur de 3.5m à une distance minimale de 6m),

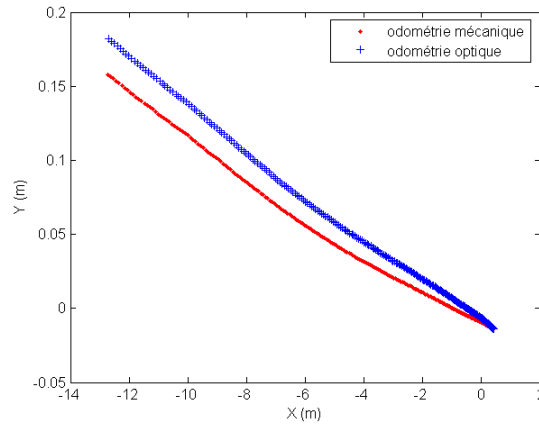


FIG. 5.12 – Comparaison de l'odométrie visuelle avec l'odométrie optique

- le capteur doit pouvoir fonctionner de jour comme de nuit (avec éclairage des feux de croisement).

A titre d'exemple, la figure 5.14 présente la plage de disparité du système de stéréovision avec l'utilisation de différentes focales. La focale 35 mm présente une grande plage de disparité qui serait très utile à la détection d'obstacles éloignés mais rendrait le système *aveugle* à de faibles distances. *A contrario*, une focale très ouverte telle que 2.6 mm pourrait permettre la détection d'obstacles très proches au prix d'une certaine *myopie*. La focale choisie de 8 mm correspond au meilleur compromis vis-à-vis des spécifications ARCOS.

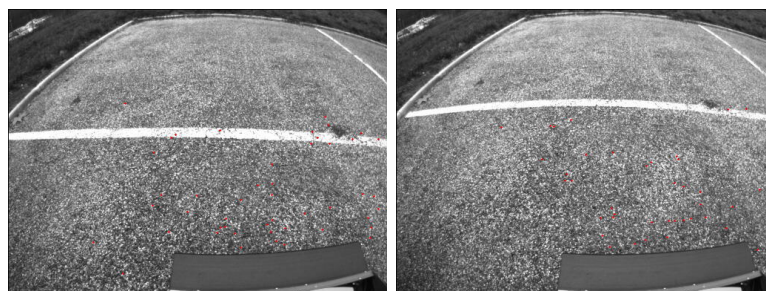
La figure 5.15 présente la détection d'un obstacle proche (moins de 12 mètres). La reconstruction d'une ligne de la route est présentée en (a) avec les histogrammes correspondant en (b), (c) et (d). La présence de pics discriminants donne les informations de hauteur et de profondeur de cette ligne dans l'environnement. Dans ce cas précis, la ligne se trouve à environ 11.3 mètres à l'avant du véhicule et à 75 cm en dessous des caméras. L'histogramme sur les disparités, utile à la *v-disparité*, correspond à une valeur de 45.5 pixels, valeur équivalente aux 11.3 m trouvés par reconstruction 3D.

Le profil de route présenté en (e), est beaucoup plus bruité que les résultats obtenus sur images de synthèse mais permet tout de même l'extraction de l'obstacle. On remarque un certain angle de la route par rapport à l'horizontal (ou plus exactement par rapport à l'axe des caméras) certainement dû à un angle de tangage du véhicule. L'*intervalle obstacle* représenté par des lignes vertes indique un obstacle entre 10 et 12 m. L'extraction des points obstacles au sein de cet intervalle est présentée par des points rouges.

La caractérisation de l'obstacle visualisé en (f) montre bien les défauts de la méthode employée :

- la largeur de l'obstacle est surestimée à cause du score utilisé pour la stéréocorrélation qui *grossit* les contours,
- la hauteur de l'obstacle n'est pas entièrement détectée du fait de la limitation de la corrélation à la surface de la route, effet qui peut être limité en forçant la corrélation au-dessus d'une zone détectée comme obstacle.

La figure 5.16 présente la détection d'un obstacle éloigné à une distance d'environ 30 mètres. A cette distance, le profil de route présenté en 5.16[gauche] est très bruité et complique le calcul de la surface de la route. Malgré tout, l'obstacle est détecté (cf. 5.16[droite])



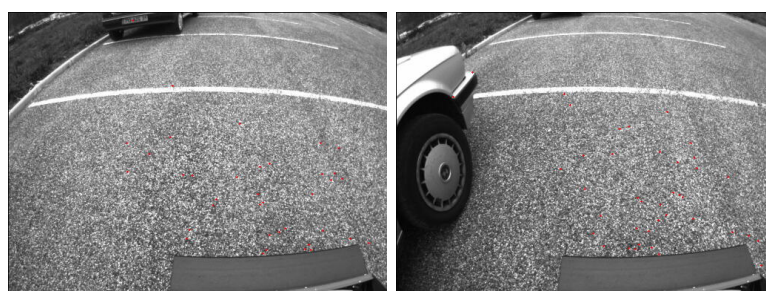
(a) image n° 0

(b) image n° 18



(c) image n° 31

(d) image n° 44



(e) image n° 49

(f) image n° 51



(g) image n° 54

(h) image n° 59

FIG. 5.13 – Suivi des points dans le temps (sur l'image gauche de la stéréovision). Les croix rouges correspondent aux points appariés avec l'image précédente.

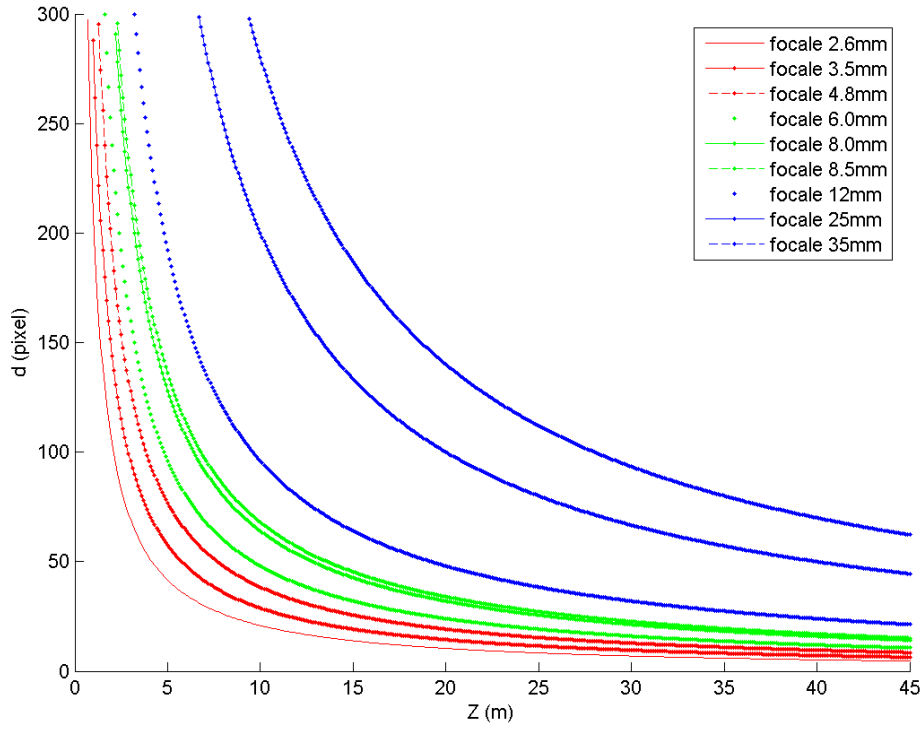


FIG. 5.14 – Courbes des disparités en fonction de la distance d'un point. Distance inter-caméras = 1 m, résolution des images 384×288 .

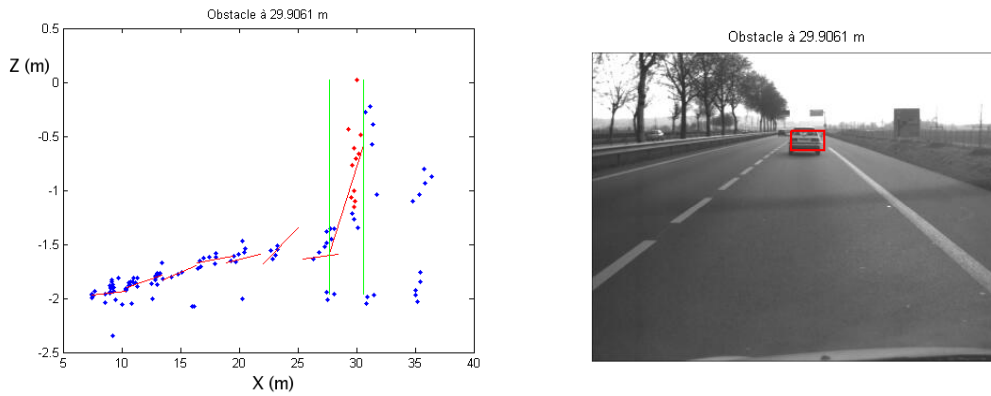
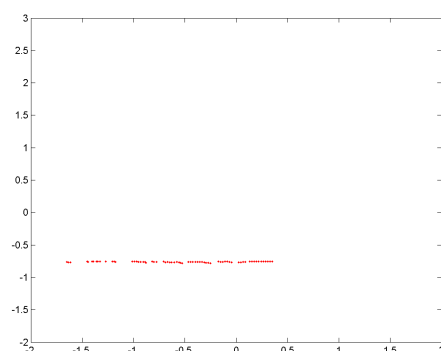
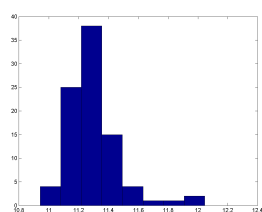


FIG. 5.16 – Profil de route [gauche] et boîte obstacle [droite] d'un véhicule éloigné (30 m).

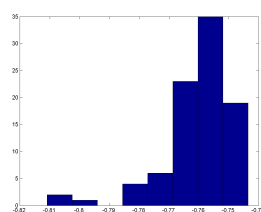
La figure 5.17 présente les résultats triviaux de détection d'obstacle sur une route sans obstacle. On remarque la densité des points dans le profil de route qui décroît avec la distance relativement à la résolution en profondeur de la stéréovision.



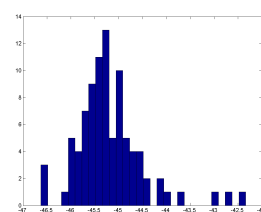
(a) Vue 3D d'une ligne de la route



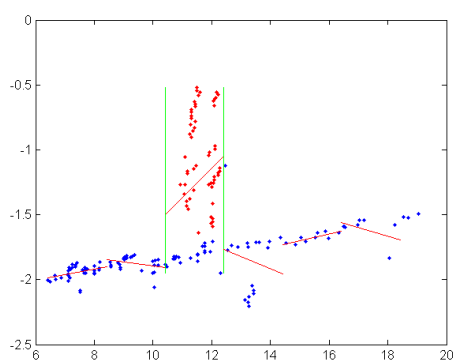
(b) Histogramme en X d'une ligne de la route située à environ 11.3 m



(c) Histogramme en Z d'une ligne de la route située à une hauteur d'environ 75 cm sous l'axe des caméras



(d) Histogramme sur la disparité faisant apparaître un pic à 45.4 pixels



(e) Profil de route



(f) Détection de l'obstacle dans les images

FIG. 5.15 – Détection d'un obstacle proche (~ 11.5 m)

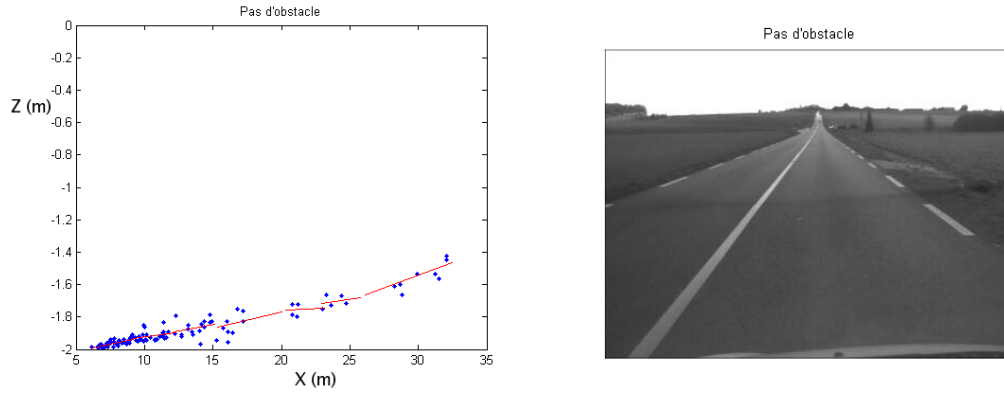


FIG. 5.17 – Profil de route [gauche] et boîte obstacle [droite] d'une route sans obstacle.

La figure 5.18 présente la détection d'un piéton traversant la route. Le haut du corps du piéton pose les mêmes problèmes que la détection du haut du véhicule lorsque ceux-ci sont en-dehors de la zone étiquetée comme *route*.

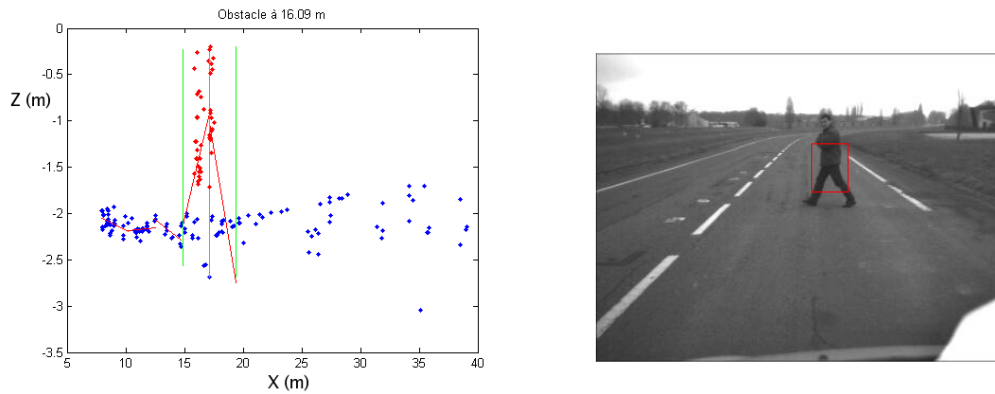


FIG. 5.18 – Profil de route [gauche] et boîte obstacle [droite] d'une route avec détection d'un piéton.

La figure 5.19 présente le résultat d'une détection de nuit. La faible luminosité a tendance à limiter le bruit dans l'image de disparité mais limite aussi énormément le nombre de points sur une ligne de la route. Ceci conduit à un nombre de points un peu plus faible dans le profil de route, surtout à grande distance. De gros problèmes de *smearing* apparaissent lorsqu'un véhicule arrive de face les phares allumés (cf. figure 5.20) et peuvent aboutir à de fausses détections. L'utilisation de caméra CMOS pourrait limiter cet effet au détriment du coût du système et du bruit.

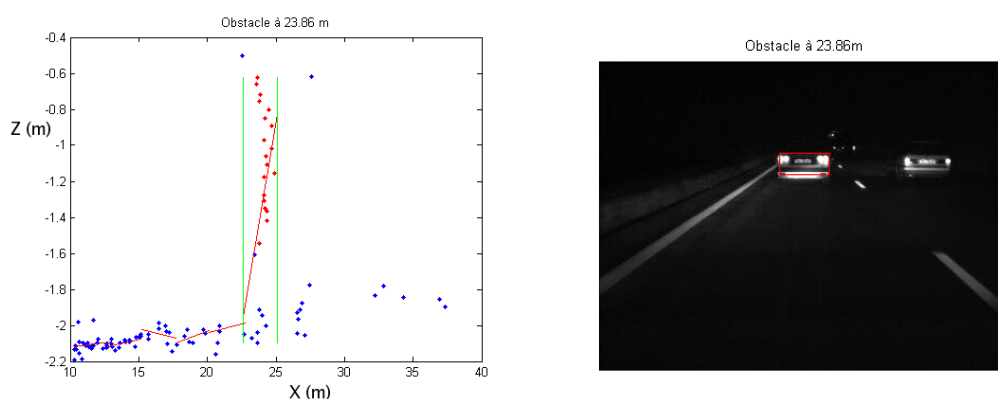


FIG. 5.19 – Profil de route [gauche] et boîte obstacle [droite] d'une route la nuit.

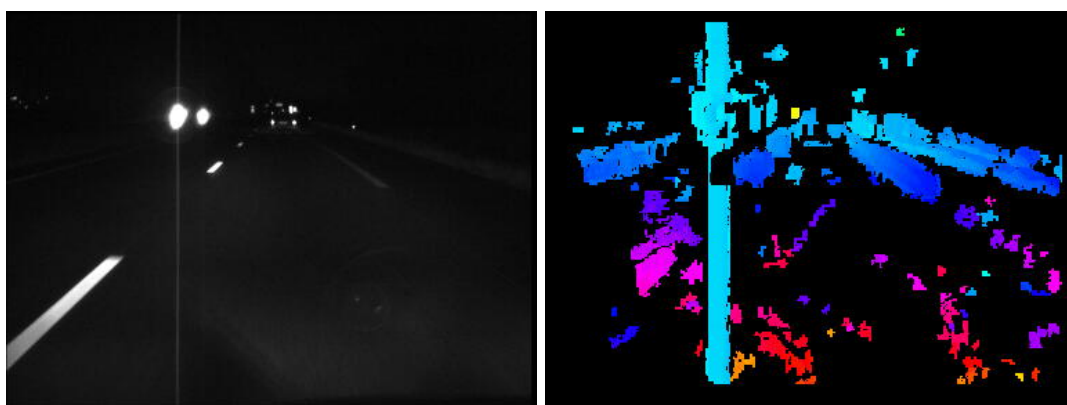


FIG. 5.20 – Apparition d'artefacts dus à la saturation du capteur CCD lors de fortes illuminations.

5.5 Modélisation de la place de parking

Pour la modélisation de la place de parking, le système doit détecter des obstacles proches, et le champ de perception doit être très large afin d'éviter les angles morts. De plus, vu les contraintes économiques posées par l'intégration de capteurs sur une automobile, il est impératif de limiter au maximum le nombre de capteurs : nous avons donc évalué deux configurations avec un seul banc stéréo

- banc stéréo positionné sur l'axe central du véhicule, soit à l'avant (place détectée en passant devant comme le fait le conducteur), soit à l'arrière (place détectée quand elle est dépassée).
- banc stéréo positionné sur une aile avant, dans ce cas regardant à 45 deg vers la gauche.

La figure 5.21 présente la plage de disparité d'un banc stéréoscopique de 33cm de base pour une résolution de 320×240 . Nous pourrions choisir une valeur très faible pour la focale, mais cela poserait plusieurs problèmes : (1) la résolution en profondeur serait très faible et limiterait la distance de perception et (2) la correction des distorsions fortement marquées poserait problème. Par contre, le choix d'une grande focale n'est pas possible vu les distances très faibles des obstacles.

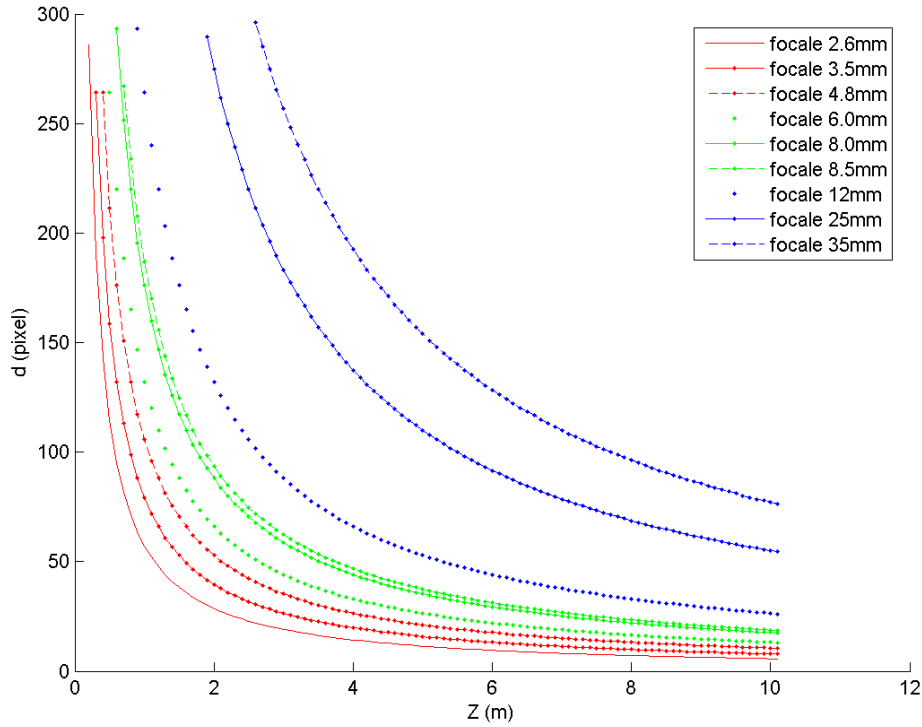


FIG. 5.21 – Courbes des disparités en fonction de la distance d'un point. Distance inter-caméras = 33 cm, résolution des images 320×240 .

La figure 5.22 présente les différentes étapes de la modélisation locale, donc du traitement d'une seule paire d'images. La figure (a) présente l'image gauche de la paire stéréo ; on y voit une voiture garée le long d'un trottoir, une autre derrière le trottoir et un obstacle cylindrique sur ce trottoir. La figure (b) présente l'image gauche rectifiée ; la figure (c) représente la reconstruction 3D d'une vue à l'aide de cette vue stéréoscopique. Après seuillage des points selon la hauteur, les points obstacles sont retirés de cette image 3D pour ne laisser que les points de la chaussée et du trottoir, visualisés en figure (d). La vue de profil présentée en (e) est alors utilisée pour la détection du trottoir. Les points obstacles sont regroupés en obstacles distincts dans des boîtes englobantes (f). Après détection du trottoir, l'ensemble du modèle local présenté en (g) permet de retrouver les éléments caractéristiques de la modélisation de la place de parking :

- les obstacles dans des boîtes englobantes,
- la limite chaussée/trottoir matérialisée par une ligne verte,
- l'espace libre le long du trottoir matérialisé par une ligne violette.

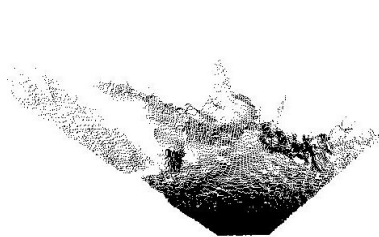
Les séquences présentées ici ont été réalisées sur le robot *Dala* avec des focales de 3.5mm et une distance intercaméra de 33cm permettant un champ relativement large pour une bonne résolution en profondeur jusqu'à une dizaine de mètres. Les deux séquences présentées ci-dessous illustrent les deux configurations choisies : la première séquence exploite un banc stéréo, simulant un système fixé sur l'axe central d'un véhicule, regardant vers l'arrière, permettant de rechercher une place aussi bien du coté gauche que du coté droit. La seconde séquence utilise un banc orienté à 45° sur le coté, simulant un système intégré dans l'aile avant gauche d'un véhicule, permettant uniquement de rechercher une place du coté gauche. Ces deux séquences ont ensuite été rejouées



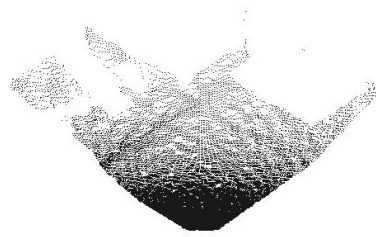
(a) Image gauche



(b) Image gauche rectifiée



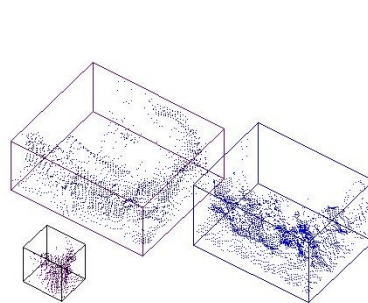
(c) Reconstruction 3D



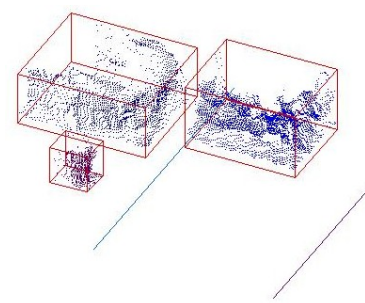
(d) Reconstruction 3D privée des points obstacles



(e) Chaussée vue de profil



(f) Obstacles et boîtes englobantes



(g) Modèle de scène

FIG. 5.22 – Les étapes de la modélisation locale

sous RT-Maps, avec une fréquence de 10Hz environ.

5.5.1 Banc de stéréovision sur l'axe central, à l'arrière

Seules 8 images sur les plus de 500 de la séquence sont présentées sur la figure 5.23. Les colonnes (a) et (b) représentent les images gauche et droite du banc stéréoscopique. Le fait de ne pas orienter la caméra d'un côté ne permet pas de distinguer le trottoir correctement dans les images 3D rendant impossible sa détection. La colonne (c) en fausse couleur représente l'image de disparité de bonne qualité sur cette séquence. En effet, la texture du goudron est généralement propice à l'utilisation de la stéréocorrélation. Les pixels étiquetés comme obstacles sont présentés en colonne (d). Le côté gauche des obstacles n'est pas détecté du fait des champs de perception différents des deux caméras. Une vue de dessus du modèle de voie est donné en colonne(e). Le rectangle bleu englobant le point vert représente le véhicule porteur du système. La fenêtre visualisée étant centrée sur le véhicule, celui-ci ne semble pas bouger mais les obstacles représentés par des rectangles pleins s'éloignent au fur et à mesure que le véhicule avance.

A la fin de la séquence, l'erreur sur le premier obstacle est de 9cm et l'erreur sur l'espace libre est de 8cm. Les mêmes tests, dans les mêmes conditions (position et orientation des caméras), ont été réalisés avec le banc de la marque Videre STH-MDCS dont la distance inter-caméra est de 9cm. Malgré l'attrait de la compacité, la qualité des images 3D rend son utilisation très difficile pour cette application avec des erreurs dépassant les 30cm.

5.5.2 Banc stéréo sur le côté, à l'avant

L'orientation du capteur sur un côté spécifique de la chaussée permet de percevoir plus d'informations sur les obstacles et sur l'espace libre, rendant possible l'extraction du trottoir. La séquence présentée en figure 5.24 représente en colonne (a) l'image gauche de la paire stéréoscopique, en colonne (b) la reconstruction 3D de la vue courante et en colonne (c) le modèle global de la voie. Sur la première image, seul un obstacle est visible et est représenté par un parallélépipède rectangle dans le modèle de voie. Remarquons que sur cette première paire, un deuxième obstacle est détecté très proche du premier.

Le véhicule avance et perçoit le premier obstacle sous un autre angle de vue. Les deux obstacles précédents sont alors fusionnés et identifiés comme étant un seul et même obstacle. Sur la troisième image, l'espace libre devient visible et laisse percevoir le trottoir. Le modèle de voie s'enrichit donc de cette limite (ligne verte) ainsi que la longueur disponible le long du trottoir (ligne violette). De nouveaux obstacles sont visibles comme le second obstacle délimitant la place de parking ainsi qu'un petit obstacle sur le trottoir et un autre véhicule de l'autre côté du trottoir. Au fur et à mesure de l'avancement du véhicule, le modèle se complète. L'erreur sur le premier véhicule n'est plus que de 6cm et 4cm sur l'espace libre. Le gain s'explique par une meilleure résolution des données 3D, et donc une meilleure représentation des obstacles dans les modèles locaux.

5.5.3 Conclusion

Imposer des contraintes trop sévères sur la configuration du capteur stéréoscopique, rend le système défaillant. Cela ne permet pas une détection fiable de l'espace libre et des obstacles. Par contre, en choisissant correctement les paramètres du système (distance inter-caméra, focale, position et orientation du banc sur le véhicule), la stéréovision semble tout à fait adaptée

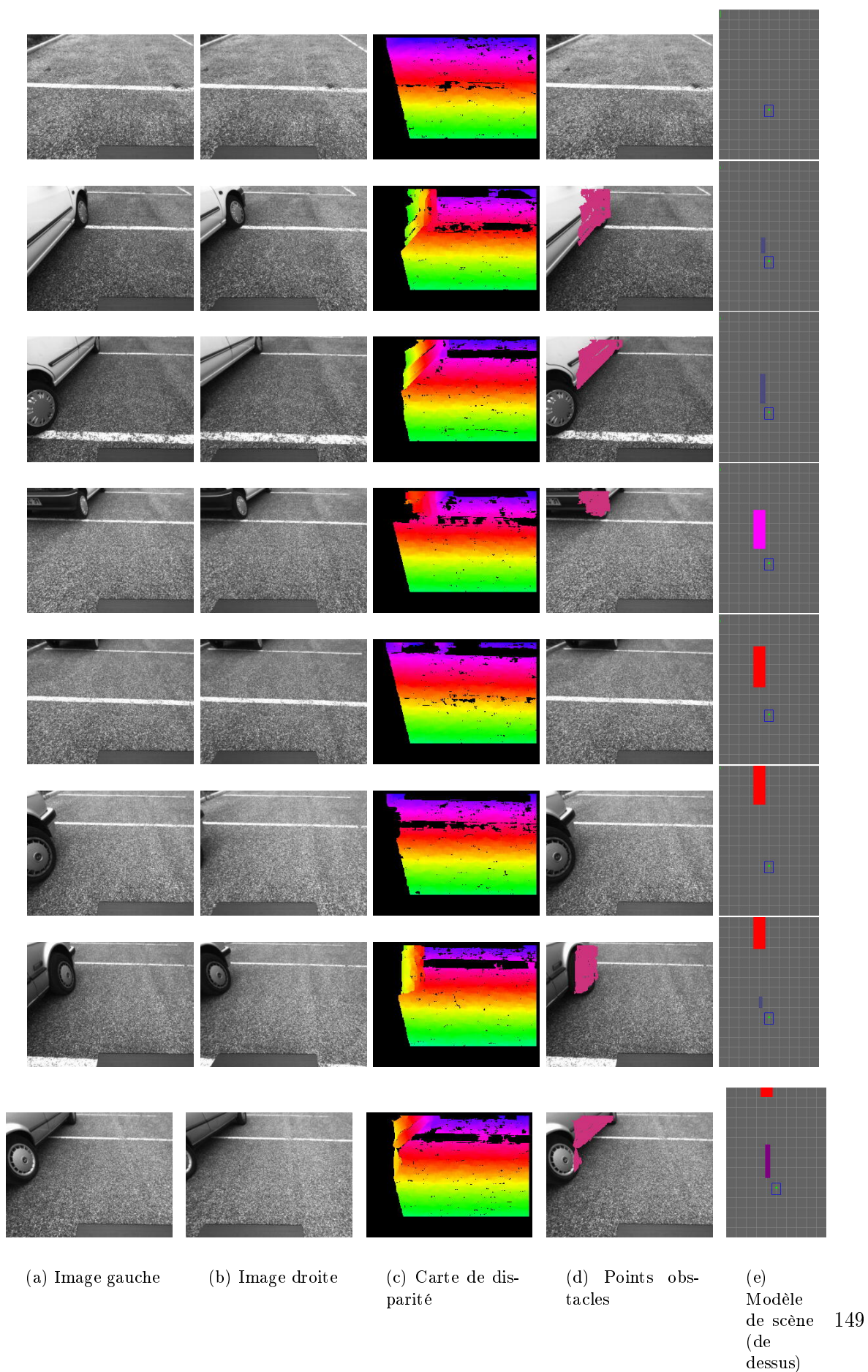


FIG. 5.23 – Résultat de modélisation de place de parking (banc stéréo orienté dans l’axe du véhicule)

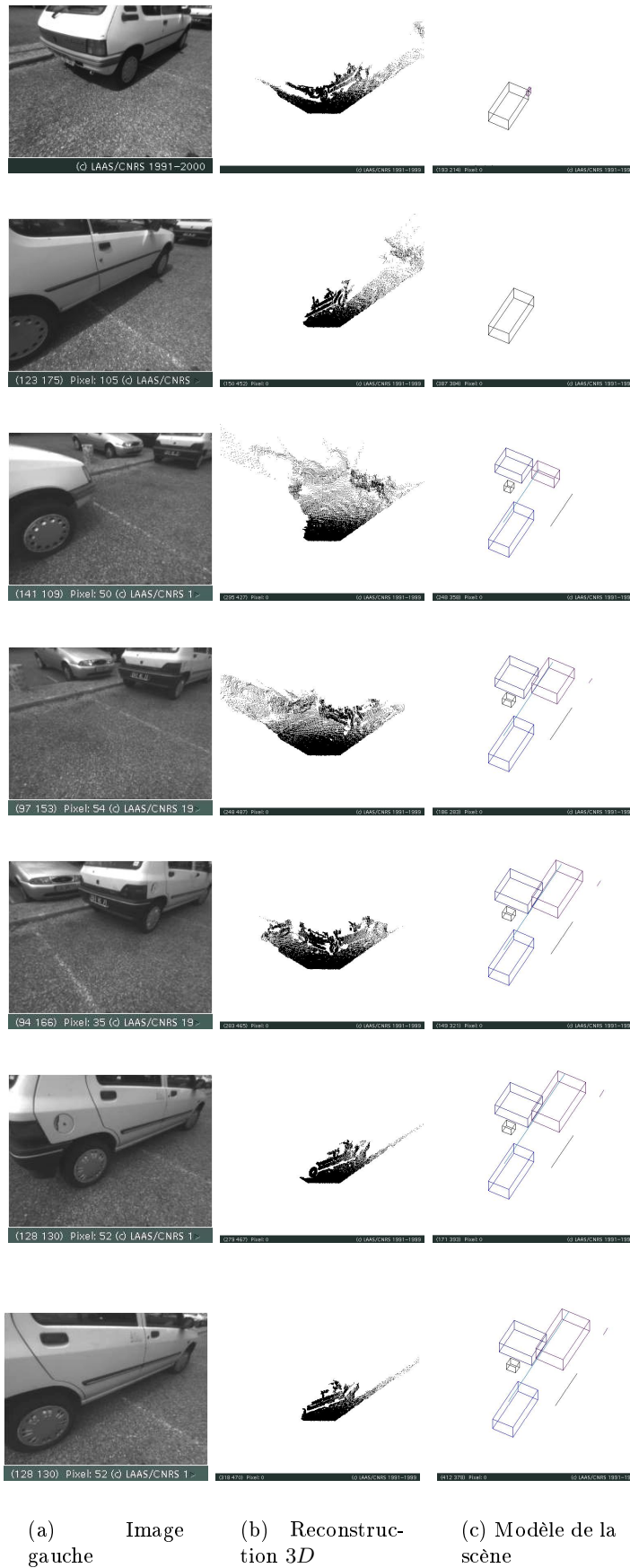


FIG. 5.24 – Résultat de modélisation de place de parking (banc stéréo orienté vers un coté de la chaussée)

à cette application d'aide à la conduite. L'ensemble des algorithmes décrits dans cette thèse a été développé sur la plate-forme RT-MAPS et intégré sur des véhicules et le robot DALA. Les performances du système atteignent les 10 Hz sur un pentium centrino 1.7 GHz pour des images de 320×240 pixels.

Afin d'améliorer ces performances et pour diminuer la charge du processeur, des tests ont été effectués à l'aide du capteur *3D* à temps de vol optique de la société CSEM. Les résultats sont concluants, mais des progrès du capteur sont à attendre en terme de temps d'intégration des photosites et de robustesse aux fortes illuminations. Par ailleurs, le coût de ce capteur rend très improbable son intégration dans des voitures de série.

Conclusions et Perspectives

1 Conclusions

La perception de l'état d'un véhicule et de son environnement reste un problème crucial dans le développement de l'automobile dite *intelligente*. Du fait de la complexité de ce problème, essentiellement dû à la complexité des situations observées, les constructeurs et équipementiers tendent, depuis ces dernières décennies, à multiplier les études et les développements de dispositifs de perception. Ceci bien que leur intégration sur les véhicules de série se limite au mieux aux véhicules haut de gamme et reste souvent réservé au domaine de la recherche et du développement de prototypes.

L'objectif de cette thèse était d'analyser les applications possibles de la stéréovision embarquée sur un véhicule. Nous nous sommes intéressés à deux applications d'aide à la conduite ainsi qu'à un système permettant le fonctionnement ininterrompu de la stéréovision malgré les problèmes inhérents à la calibration.

Un système stéréoscopique ne peut être utilisé longuement sur un engin mobile soumis à des chocs et vibrations. Habituellement, la calibration du capteur de stéréovision est obtenue à partir d'une mire placée dans le champ de perception de chacune des caméras. L'impossibilité d'utiliser cette technique sur un véhicule de série explique des études sur l'application en ligne des méthodes automatiques de calibration. Nous nous sommes intéressés à la mise à jour en ligne de la calibration d'un capteur stéréoscopique embarqué sur un véhicule, avec l'hypothèse que nous pouvons détecter toute décalibration, c'est-à-dire toute variation de la géométrie du capteur. La méthode présentée dans cette thèse ne se base plus sur des points d'une mire mais sur des points appariés dans les images sans tenir compte des paramètres courants de calibration. Les résultats présentent la capacité à réestimer les orientations relatives des deux caméras ainsi que la distance inter-caméra à un facteur d'échelle près. Une application dérivée de cette méthode est l'odométrie optique, permettant d'évaluer le déplacement d'une caméra (et du véhicule sur lequel elle est rigidement montée) entre deux vues.

En ce qui concerne la détection d'obstacles en milieu routier, nous avons proposé une méthode originale basée sur la reconstruction **3D** de l'environnement. Le traitement d'image par stéréocorrélation permet la détection de la géométrie de la route afin d'en extraire les obstacles : pour ce faire, nous avons au préalable implémenté une méthode classique de détection de la voie de circulation à partir des marquages. Les résultats obtenus permettent de s'affranchir des variations de pente et du dévers/roulis pour détecter les obstacles de jour comme de nuit. Nous avons comparé notre approche avec la méthode dite de *v-disparité*, considérée comme la plus performante à ce jour. Cette approche modélise la route et détecte les obstacles uniquement à partir de la disparité : notre méthode est plus complexe, puisqu'elle nécessite une reconstruction

3D explicite, mais nos résultats préliminaires montrent que l'exploitation des données 3D permet de prendre en compte des géométries plus complexes de la route.

En ce qui concerne la modélisation de la place de parking, nous avons présenté une méthode de segmentation de l'environnement 3D permettant d'extraire l'espace libre au stationnement. La méthode utilise les résultats de la stéréocorrélation pour extraire simplement les obstacles de l'image 3D. Cette image, privée des obstacles, permet de détecter d'éventuels trottoirs. L'ensemble des informations extraites sont suivies dans le temps afin de valider la présence ou non d'un obstacle après plusieurs perceptions et afin de détecter un espace libre adapté pour le stationnement. La solution retenue devrait permettre de détecter l'espace libre dans un grand nombre de configurations : crèneau, bataille ou épi, chaussée limitée par un trottoir, un marquage, un obstacle ... Notons cependant que l'approche a été validée uniquement pour une place en crèneau le long d'un trottoir. Ces travaux ont été menés dans le cadre de coopérations avec des partenaires industriels.

2 Perspectives

De manière générale, l'ensemble des algorithmes présentés dans cette thèse doivent être évalués sur un grand nombre de séquence sur le véhicule expérimental du LAAS-CNRS. Les évaluations sur les bases de données accessibles par Internet, notamment celles du projet ARCOS, sont nécessaires mais pas suffisantes. L'évaluation du taux de détection et de fausses alarmes permettrait de mieux classer les situations difficiles pour nos algorithmes afin de les améliorer.

De manière plus détaillée, un module d'autocalibration doit être implémenté sur les robots du LAAS afin de rendre les manipulations de calibration transparentes aux utilisateurs de la stéréovision. De plus, l'utilisation des appariements de points d'intérêt étant le facteur limitatif aux performances de notre système, il serait intéressant de trouver une méthode plus légère et plus robuste aux différences de luminosité entre les deux caméras.

La détection d'obstacles en milieu routier présente encore des problèmes de boîtes englobantes, soit élargies par rapport à la vérité soit limitées en hauteur. Il faudra donc améliorer le critère de corrélation de la stéréovision afin de l'adapter à notre application pour améliorer le taux de corrélation et limiter le *grossissement des bords*. De plus, des problèmes de différenciation d'un même obstacle apparaissent lorsque seuls les bords droit et gauche d'un obstacle sont détectés sans la zone centrale aux deux bords. L'algorithme actuel ne prend pas en compte la redondance temporelle très forte dans cette application. Sa prise en compte permettrait de réduire les temps d'exécution et de valider une cible détectée dans le temps avec sa caractérisation dynamique. Les résultats présentés dans cette thèse montrent une bonne détection des obstacles proches ($< 30m$) mais la méthode est très dépendante du bruit pour des obstacles éloignés. Les conclusions du projet ARCOS ont bien montré qu'une méthode de fusion d'un télémètre laser et d'un capteur stéréoscopique permettrait la réalisation d'un système complet de détection d'obstacles.

La modélisation de la place de parking fait l'hypothèse d'un monde statique. Il serait intéressant d'extraire les obstacles dynamiques. Une première approche pourrait consister à considérer les points rejetés par RANSAC lors de l'estimation de l'odométrie optique ; en regroupant ces points avec la reconstruction 3D de l'obstacle correspondant, cela permettrait d'identifier un obstacle dont la dynamique est différente de l'environnement statique. L'hypothèse du monde

statique pourrait alors être levée. Notons que J.Sola dans notre équipe, étudie la détection d'obstacles dans un environnement dynamique, en vision monoculaire. Il propose une approche de type SLAM, donc la mise à jour incrémentale d'une carte stochastique par un filtre considérant plusieurs hypothèses sur la nature des obstacles (statique ou dynamique) et sur leur profondeur.

De manière générale, les fonctions de perception dans le domaine des transports, doivent satisfaire des contraintes temps réel très sévères : 100Hz par exemple pour la détection d'obstacles en milieu routier. Avec une implémentation purement logicielle, la stéréovision ne peut pas pour l'instant, rivaliser avec les capteurs radar ou lidar pour les applications en milieu routier, ou avec les capteurs ultra-sons en milieu urbain : il sera donc indispensable d'étudier des systèmes intégrés, fondés sur la vision, capables de mettre à jour à 100Hz des modèles d'obstacles ou d'espace libre, dans des conditions d'illumination variables. Le projet PICAS\$O en cours au LAAS-CNRS, poursuit cet objectif ; une version simplifiée de l'algorithme de stéréovision décrit dans ce mémoire, est implantée sur FPGA, et produit une image de disparité toutes les 10ms.

Enfin, à plus long terme, il faudra étudier les problèmes liés à la coopération entre véhicules et infrastructure : c'est le thème des *Véhicules communicants* et de la *Route Automatisée*. Actuellement, il n'y a aucune coopération entre les différents acteurs sur la route autre que les règles de conduite du code de la route. Chaque usager évalue lui-même l'état de son véhicule et de l'environnement afin de prendre les décisions qui lui semblent les plus adaptées. L'information est donc indépendante entre tous les acteurs. Si les véhicules et la route étaient dotés de moyens de communication courte portée (*Bluetooth*, *Zigbee*...), alors, tous les acteurs pourraient collaborer. Chaque véhicule pourrait alors évaluer son état sur la base de ses capteurs, mais aussi des informations reçues des autres acteurs ; il transmettrait cet état aux acteurs les plus proches. De plus, l'infrastructure elle-même pourrait communiquer les dangers propres à la configuration courante de la route. L'information serait alors partagée entre les acteurs et permettrait une amélioration des performances des systèmes d'aide à la conduite.

Annexes

Annexe A

Ajustement de faisceaux

La modélisation 3D d'un environnement à partir d'un système stéréoscopique non calibré, ou d'une caméra mobile (*Structure From Motion* ou *Motion Stereo*), a été l'objet de nombreux travaux dans la communauté Vision Artificielle, en particulier le célèbre algorithme des 8 points proposé par Longuet-Higgins en 1981, source des travaux de Hartley, Faugeras and co sur la géométrie projective à partir de 1992. Le but est d'estimer le mouvement ou la configuration d'un capteur (mono ou stéréo) et de reconstruire un modèle 3D de l'environnement sous la forme d'un ensemble de points 3D. En fait, ces travaux exploitent une technique connue depuis fort longtemps dans la communauté Photogrammétrie, notamment l'ajustement de faisceaux²⁹ (noté BA ci-après), méthode globale d'optimisation non linéaire pour estimer tous les paramètres inconnus à partir de toutes les mesures extraites des images.

Dans le contexte du calibrage d'un capteur de vision (mono ou stéréo), l'algorithme BA consiste en une estimation simultanée des paramètres intrinsèques $k = (\alpha_u, \alpha_v, u_0, v_0)^t$ et des paramètres de distorsion $d = (r_1, r_2, r_3)^t$ des caméras, des paramètres extrinsèques T_i (position et orientation de la mire de calibrage dans les différentes images) et en une réestimation des coordonnées des points de calibrage M_i de la mire, à partir des pixels correspondants extraits des images. Ces paramètres sont estimés à l'aide de l'algorithme d'estimation non linéaire de *Levenberg-Marquardt* (noté LM ci après) [Levenberg, 1944]. La photogrammétrie a été la principale application utilisatrice de cette méthode pour la reconstruction de terrain depuis des images aériennes acquises depuis une seule caméra en mouvement. De nos jours, l'algorithme BA est de plus en plus répandu dans le domaine de la vision pour la robotique et est fortement utilisé pour des techniques d'auto-calibrage, comme nous l'avons illustré dans cette thèse.

La méthode présentée dans cette annexe est appelée *Sparse Bundle Adjustment* (SBA). C'est une variation sur l'algorithme LM qui prend en compte la structure éparse des matrices mises en jeu dans le processus d'estimation. La description de la méthode et les notations sont largement inspirées de [Hartley et al., 2000]. Le terme de *faisceau* fait référence aux faisceaux de rayons optiques entre le centre optique de la caméra et les points 3D de la scène. Ces faisceaux sont alors *ajustés* de manière à optimiser les coordonnées des points 3D appelés **structure** et les paramètres de la caméra. Cette optimisation vise à minimiser les erreurs entre les reprojections des points 3D observés et les points qui leur sont appariés dans les images ; ces erreurs peuvent être exprimées comme la somme des carrés de fonctions non linéaires. C'est pourquoi ce problème est résolu par des techniques d'estimation non linéaires telles que l'algorithme LM qui a prouvé son

²⁹ *Bundle adjustment* en anglais

efficacité de convergence même avec des estimées initiales fortement bruitées [Hiebert, 1981]. La fonction à minimiser est linéarisée itérativement dans le voisinage de l'estimée courante permettant de résoudre le système linéaire appelé **équation normale**. La résolution répétée de cette équation normale peut être très longue au vu de sa complexité en $O(N^3)$, fonction du nombre de paramètres. Heureusement, l'écriture détaillée de la matrice du système à résoudre, révèle une structure fortement éparse due à la forte indépendance des points *3D* entre eux. Il est donc important de prendre en compte la structure des matrices pour améliorer les performances du système.

La première section de cette annexe pose les notations en présentant l'algorithme de *Levenberg-Marquardt*. La section suivante détaillera alors les structures des différentes matrices mises en jeu.

A.1 Algorithme de *Levenberg-Marquardt*

L'algorithme LM est une technique itérative de recherche du minimum d'une fonction multivariable, exprimée comme la somme des carrés de fonctions non linéaires [Levenberg, 1944, Marquardt, 1963]. Cet algorithme est devenu un standard pour les problèmes d'estimation non linéaire dans la communauté Vision Artificielle. Il peut être vu comme une combinaison entre les méthodes de descente de gradient et de *Gauss-Newton*. C'est à dire que lorsque la solution est loin de l'estimée, l'algorithme converge lentement mais avec une garantie de converger. Lorsque la solution est proche, l'algorithme se comporte comme la méthode de *Gauss-Newton* afin d'éviter les minima locaux.

Posons f la fonction liant le vecteur de paramètres $\mathbf{p} \in \mathcal{R}^m$ avec le vecteur de mesures $\hat{\mathbf{x}} = f(\mathbf{p}), \hat{\mathbf{x}} \in \mathcal{R}^n$. Le vecteur d'estimées initiales est noté $\mathbf{p}_0$. Le vecteur optimal recherché est noté $\mathbf{p}^+$ et doit satisfaire au mieux la fonction f , c'est à dire qu'il doit minimiser la distance au carré $\epsilon^t \epsilon$ avec $\epsilon = \mathbf{x} - \hat{\mathbf{x}}$. Le principe de LM est de faire une approximation linéaire de f au voisinage de $\mathbf{p}$. Pour un petit $\|\delta_{\mathbf{p}}\|$, l'approximation selon une série de Taylor s'écrit :

$$f(\mathbf{p} + \delta_{\mathbf{p}}) \approx f(\mathbf{p}) + \mathbf{J}\delta_{\mathbf{p}} \quad (\text{A.1})$$

avec $\mathbf{J}$ la matrice jacobienne $\frac{\partial f(\mathbf{p})}{\partial \mathbf{p}}$. Comme toute méthode d'optimisation non linéaire, l'algorithme LM est une méthode itérative : l'initialisation se fait en $\mathbf{p}_0$ et produit les vecteurs $\mathbf{p}_1, \mathbf{p}_2, \dots$ et converge vers $\mathbf{p}^+$ correspondant à un minimum local de f . A chaque itération, $\delta_{\mathbf{p}}$ est recherché qui minimise $\|\mathbf{x} - f(\mathbf{p} + \delta_{\mathbf{p}})\| \approx \|\mathbf{x} - f(\mathbf{p}) - \mathbf{J}\delta_{\mathbf{p}}\| = \|\epsilon - \mathbf{J}\delta_{\mathbf{p}}\|$. $\delta_{\mathbf{p}}$ est la solution du système linéaire des moindres carrés : le minimum est atteint lorsque $\mathbf{J}\delta_{\mathbf{p}} - \epsilon$ est orthogonale à $\mathbf{J}$, c'est à dire lorsque $\mathbf{J}^t(\mathbf{J}\delta_{\mathbf{p}} - \epsilon) = \mathbf{0}$ et que $\delta_{\mathbf{p}}$ est solution de **l'équation normale** décrite par :

$$\mathbf{J}^t \mathbf{J} \delta_{\mathbf{p}} = \mathbf{J}^t \epsilon \quad (\text{A.2})$$

La méthode LM résoud une variante de A.2 appelée **équation normale augmentée** donnée par :

$$\mathbf{N} \delta_{\mathbf{p}} = \mathbf{J}^t \epsilon \quad (\text{A.3})$$

où les termes non diagonaux de $\mathbf{N}$ sont identiques à $\mathbf{J}^t \mathbf{J}$ et les termes diagonaux sont données par $\mathbf{N}_{ii} = \mu + [\mathbf{J}^t \mathbf{J}]_{ii}$ pour $\mu > 0$. Cette technique d'altération de la diagonale est appelée le **damping** et μ est appelé le **damping term**. Si le vecteur $\mathbf{p} + \delta_{\mathbf{p}}$ avec le $\delta_{\mathbf{p}}$ calculé en A.3

conduit à une réduction de l'erreur ϵ , alors la mise à jour $\delta_{\mathbf{p}}$ est validée et l'itération suivante utilisera un *damping term* plus faible. Sinon, μ est augmenté jusqu'à trouver un $\delta_{\mathbf{p}}$ qui réduise ϵ .

L'algorithme s'arrête lorsque au moins une des situations suivantes est atteinte :

- $\mathbf{J}^t \epsilon$, terme de droite dans l'équation A.3, est inférieur à un seuil ϵ_1 ,
- la variation de $\delta_{\mathbf{p}}$ est inférieure à un seuil ϵ_2 ,
- un nombre d'itérations maximal k_{max} est atteint.

Il est possible d'introduire une matrice de covariance $\Sigma_{\mathbf{x}}$ du vecteur de mesures $\mathbf{x}$, si celle-ci est disponible, en minimisant $\epsilon^t \Sigma_{\mathbf{x}}^{-1} \epsilon$ au lieu de $\epsilon^t \epsilon$. L'équation A.3 s'écrit alors :

$$\mathbf{J}^t \Sigma_{\mathbf{x}}^{-1} \mathbf{J} \delta_{\mathbf{p}} = \mathbf{J}^t \Sigma_{\mathbf{x}}^{-1} \epsilon \quad (\text{A.4})$$

et s'appelle l'équation normale pondérée. Le déroulement de l'algorithme reste inchangé.

A.2 Ajustement de faisceaux

Cette section présente comment appliquer l'algorithme LM efficacement au problème de l'ajustement de faisceaux. Supposons que n points 3D sont visibles dans m vues et notons $\mathbf{x}_{ij}$ la projection du i -ème point dans la j -ème image ; les vues peuvent être acquises par une seule caméra mobile, ou par plusieurs caméras fixes, ou par un banc stéréo (deux caméras rigidement liées) mobile. L'algorithme BA a pour vocation de réestimer l'ensemble des paramètres décrivant le système visuel et la structure de la scène observée, décrite par les n points 3D ; la méthode trouve l'ensemble des paramètres qui prédit au mieux les mesures des n points dans les m images.

Notons $\mathbf{a}_j$ le vecteur des paramètres de la caméra d'où provient l'image j , et $\mathbf{b}_i$ les coordonnées du point 3D i exprimées dans un repère unique. Supposons, pour l'instant et pour simplifier les écritures, que les n points sont visibles dans toutes les images (nous verrons plus tard que cette hypothèse n'est pas nécessaire). L'algorithme BA produit l'ensemble des paramètres qui permet de minimiser l'erreur de reprojection de l'ensemble des points 3D :

$$\min_{\mathbf{a}_j, \mathbf{b}_i} \sum_{i=1}^n \sum_{j=1}^m d(\mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i), \mathbf{x}_{i,j})^2 \quad (\text{A.5})$$

avec $\mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i)$ la prédiction de la projection du point 3D i sur l'image j et $d(\mathbf{x}, \mathbf{y})$ représente la distance euclidienne dans l'image des deux points $\mathbf{x}$ et $\mathbf{y}$. Par définition, A.5 est tolérant à des points manquants dans les vues puisqu'il minimise un critère de distance physique. La définition de $\mathbf{Q}()$ est générique à n'importe quelle paramétrisation de la caméra et de la structure. Notons k et λ les dimensions respectives de $\mathbf{a}_j$ et $\mathbf{b}_i$. Le nombre total de paramètres à estimer est alors $mk + n\lambda$, nombre qui peut devenir très grand dans des problèmes d'ajustement de faisceaux.

Notons $\mathbf{P} \in \mathcal{R}^M$ le vecteur global des paramètres décrit par $\mathbf{P} = (\mathbf{a}_1^t, \dots, \mathbf{a}_m^t, \mathbf{b}_1^t, \dots, \mathbf{b}_n^t)^t$ et $\mathbf{X} = (\mathbf{x}_{11}^t, \dots, \mathbf{x}_{1m}^t, \mathbf{x}_{21}^t, \dots, \mathbf{x}_{2m}^t, \dots, \mathbf{x}_{n1}^t, \dots, \mathbf{x}_{nm}^t)^t$, $\mathbf{X} \in \mathcal{R}^N$, le vecteur des mesures des points images. $\mathbf{P}_0$ désigne le vecteur des estimées initiales des paramètres et $\Sigma_{\mathbf{X}}$ est la matrice de covariance associée aux mesures $\mathbf{X}$ (en absence de connaissance sur les covariances, $\Sigma_{\mathbf{X}}$ sera toujours, la matrice identité). Pour chaque vecteur de paramètres, un vecteur de prédiction des mesures $\hat{\mathbf{X}}$ est généré avec $\hat{\mathbf{X}} = f(\mathbf{P})$, défini par $\hat{\mathbf{X}} = (\hat{\mathbf{x}}_{11}^t, \dots, \hat{\mathbf{x}}_{1m}^t, \hat{\mathbf{x}}_{21}^t, \dots, \hat{\mathbf{x}}_{2m}^t, \dots, \hat{\mathbf{x}}_{n1}^t, \dots, \hat{\mathbf{x}}_{nm}^t)^t$ avec $\hat{\mathbf{x}}_{ij} = \mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i)$.

L'algorithme BA est la minimisation de la distance au carré de *Mahalanobis* $\epsilon^t \Sigma_{\mathbf{X}}^{-1} \epsilon$, $\epsilon = \mathbf{X} - \hat{\mathbf{X}}$ sur $\mathbf{P}$. Cette minimisation non linéaire peut être résolue directement avec la méthode LM en utilisant l'équation normale pondérée suivante :

$$\mathbf{J}^t \Sigma_{\mathbf{X}}^{-1} \mathbf{J} \delta = \mathbf{J}^t \Sigma_{\mathbf{X}}^{-1} \epsilon \quad (\text{A.6})$$

où $\mathbf{J}$ est la jacobienne de f et δ la mise à jour recherchée pour le vecteur de paramètres $\mathbf{P}$. Comme nous le verrons un peu plus tard, l'équation normale A.6 est fortement éparse par bloc du fait de la forte indépendance entre les paramètres des différentes caméras et des différents points 3D. Prenons l'exemple simple de $n = 4$ points visibles pour $m = 3$ vues. Le vecteur de mesures est donné par $\mathbf{X} = (\mathbf{x}_{11}^t, \mathbf{x}_{12}^t, \mathbf{x}_{13}^t, \mathbf{x}_{21}^t, \mathbf{x}_{22}^t, \mathbf{x}_{23}^t, \mathbf{x}_{31}^t, \mathbf{x}_{32}^t, \mathbf{x}_{33}^t, \mathbf{x}_{41}^t, \mathbf{x}_{42}^t, \mathbf{x}_{43}^t)^t$. $\mathbf{P}$ s'écrit $\mathbf{P} = (\mathbf{a}_1^t, \mathbf{a}_2^t, \mathbf{a}_3^t, \mathbf{b}_1^t, \mathbf{b}_2^t, \mathbf{b}_3^t, \mathbf{b}_4^t)^t$. Du fait de l'indépendance, remarquons dès à présent que $\frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_k} = \mathbf{0}$, $\forall j \neq k$ et $\frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_k} = \mathbf{0}$, $\forall i \neq k$. Notons $\mathbf{A}_{ij}$ et $\mathbf{B}_{ij}$ les dérivées partielles respectives $\frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_j}$ et $\frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i}$. Le vecteur δ peut se décomposer selon les paramètres des caméras et de la scène 3D $(\delta_{\mathbf{a}}^t, \delta_{\mathbf{b}}^t)^t$ et, de manière plus détaillée $(\delta_{\mathbf{a}_1}^t, \delta_{\mathbf{a}_2}^t, \delta_{\mathbf{a}_3}^t, \delta_{\mathbf{b}_1}^t, \delta_{\mathbf{b}_2}^t, \delta_{\mathbf{b}_3}^t, \delta_{\mathbf{b}_4}^t)^t$. Nous allons maintenant voir comment résoudre efficacement l'équation normale de LM en tenant compte des blocs égaux à zéro.

Avec la notation des dérivées partielles du paragraphe précédent, nous pouvons écrire la matrice jacobienne $\mathbf{J}$ de la manière suivante :

$$\mathbf{J} = \frac{\partial \mathbf{X}}{\partial \mathbf{P}} = \begin{pmatrix} A_{11} & 0 & 0 & B_{11} & 0 & 0 & 0 \\ 0 & A_{12} & 0 & B_{12} & 0 & 0 & 0 \\ 0 & 0 & A_{13} & B_{13} & 0 & 0 & 0 \\ A_{21} & 0 & 0 & 0 & B_{21} & 0 & 0 \\ 0 & A_{22} & 0 & 0 & B_{22} & 0 & 0 \\ 0 & 0 & A_{23} & 0 & B_{23} & 0 & 0 \\ A_{31} & 0 & 0 & 0 & 0 & B_{31} & 0 \\ 0 & A_{32} & 0 & 0 & 0 & B_{32} & 0 \\ 0 & 0 & A_{33} & 0 & 0 & B_{33} & 0 \\ A_{41} & 0 & 0 & 0 & 0 & 0 & B_{41} \\ 0 & A_{42} & 0 & 0 & 0 & 0 & B_{42} \\ 0 & 0 & A_{43} & 0 & 0 & 0 & B_{43} \end{pmatrix} \quad (\text{A.7})$$

L'équation A.7 est clairement éparse. Ceci est dû au fait que l'équation normale est elle-même éparse. Considérons la matrice de covariance du vecteur des mesures par la matrice diagonale suivante :

$$\Sigma_{\mathbf{X}} = \text{diag}(\Sigma_{\mathbf{X}_{11}}, \Sigma_{\mathbf{X}_{12}}, \Sigma_{\mathbf{X}_{13}}, \Sigma_{\mathbf{X}_{21}}, \Sigma_{\mathbf{X}_{22}}, \Sigma_{\mathbf{X}_{23}}, \Sigma_{\mathbf{X}_{31}}, \Sigma_{\mathbf{X}_{32}}, \Sigma_{\mathbf{X}_{33}}, \Sigma_{\mathbf{X}_{41}}, \Sigma_{\mathbf{X}_{42}}, \Sigma_{\mathbf{X}_{43}}) \quad (\text{A.8})$$

En remplaçant $\mathbf{J}$ et $\Sigma_{\mathbf{X}}^{-1}$ de l'équation A.7, le produit de matrice à gauche de A.6 devient :

$$\begin{pmatrix}
 \sum_{i=1}^4 (\mathbf{A}_{i1}^t \Sigma_{x_{i1}}^{-1} \mathbf{A}_{i1}) & 0 & 0 & \mathbf{A}_{11}^t \Sigma_{11}^{-1} \mathbf{B}_{11} & \mathbf{A}_{21}^t \Sigma_{21}^{-1} \mathbf{B}_{21} & \mathbf{A}_{31}^t \Sigma_{31}^{-1} \mathbf{B}_{31} & \mathbf{A}_{41}^t \Sigma_{41}^{-1} \mathbf{B}_{41} \\
 0 & \sum_{i=1}^4 (\mathbf{A}_{i2}^t \Sigma_{x_{i2}}^{-1} \mathbf{A}_{i2}) & 0 & \mathbf{A}_{12}^t \Sigma_{12}^{-1} \mathbf{B}_{12} & \mathbf{A}_{22}^t \Sigma_{22}^{-1} \mathbf{B}_{22} & \mathbf{A}_{32}^t \Sigma_{32}^{-1} \mathbf{B}_{32} & \mathbf{A}_{42}^t \Sigma_{42}^{-1} \mathbf{B}_{42} \\
 0 & 0 & \sum_{i=1}^4 (\mathbf{A}_{i3}^t \Sigma_{x_{i3}}^{-1} \mathbf{A}_{i3}) & \mathbf{A}_{13}^t \Sigma_{13}^{-1} \mathbf{B}_{13} & \mathbf{A}_{23}^t \Sigma_{23}^{-1} \mathbf{B}_{23} & \mathbf{A}_{33}^t \Sigma_{33}^{-1} \mathbf{B}_{33} & \mathbf{A}_{43}^t \Sigma_{43}^{-1} \mathbf{B}_{43} \\
 \mathbf{B}_{11}^t \Sigma_{11}^{-1} \mathbf{A}_{11} & \mathbf{B}_{21}^t \Sigma_{21}^{-1} \mathbf{A}_{21} & \mathbf{B}_{31}^t \Sigma_{31}^{-1} \mathbf{A}_{31} & \sum_{i=1}^4 (\mathbf{B}_{1j}^t \Sigma_{x_{1j}}^{-1} \mathbf{B}_{1j}) & 0 & 0 & 0 \\
 \mathbf{B}_{12}^t \Sigma_{12}^{-1} \mathbf{A}_{12} & \mathbf{B}_{22}^t \Sigma_{22}^{-1} \mathbf{A}_{22} & \mathbf{B}_{32}^t \Sigma_{32}^{-1} \mathbf{A}_{32} & 0 & \sum_{i=1}^4 (\mathbf{B}_{2j}^t \Sigma_{x_{2j}}^{-1} \mathbf{B}_{2j}) & 0 & 0 \\
 \mathbf{B}_{13}^t \Sigma_{13}^{-1} \mathbf{A}_{13} & \mathbf{B}_{23}^t \Sigma_{23}^{-1} \mathbf{A}_{23} & \mathbf{B}_{33}^t \Sigma_{33}^{-1} \mathbf{A}_{33} & 0 & 0 & \sum_{i=1}^4 (\mathbf{B}_{3j}^t \Sigma_{x_{3j}}^{-1} \mathbf{B}_{3j}) & 0 \\
 \mathbf{B}_{14}^t \Sigma_{14}^{-1} \mathbf{A}_{14} & \mathbf{B}_{24}^t \Sigma_{24}^{-1} \mathbf{A}_{24} & \mathbf{B}_{34}^t \Sigma_{34}^{-1} \mathbf{A}_{34} & 0 & 0 & 0 & \sum_{i=1}^4 (\mathbf{B}_{4j}^t \Sigma_{x_{4j}}^{-1} \mathbf{B}_{4j})
 \end{pmatrix} \quad (\text{A.9})$$

Notons $\sum_{i=1}^4 (\mathbf{A}_{ij}^t \Sigma_{x_{ij}}^{-1} \mathbf{A}_{ij})$ par U_j , $\sum_{i=1}^4 (\mathbf{B}_{ij}^t \Sigma_{x_{ij}}^{-1} \mathbf{B}_{ij})$ par V_i , et $\mathbf{A}_{ij}^t \Sigma_{ij}^{-1} \mathbf{B}_{ij}$ par W_{ij} . La matrice précédente devient :

$$\begin{pmatrix}
 U_1 & 0 & 0 & W_{11} & W_{21} & W_{31} & W_{41} \\
 0 & U_2 & 0 & W_{12} & W_{22} & W_{32} & W_{42} \\
 0 & 0 & U_3 & W_{13} & W_{23} & W_{33} & W_{43} \\
 W_{11}^t & W_{12}^t & W_{13}^t & V_1 & 0 & 0 & 0 \\
 W_{21}^t & W_{22}^t & W_{23}^t & 0 & V_2 & 0 & 0 \\
 W_{31}^t & W_{32}^t & W_{33}^t & 0 & 0 & V_3 & 0 \\
 W_{41}^t & W_{42}^t & W_{43}^t & 0 & 0 & 0 & V_4
 \end{pmatrix} \quad (\text{A.10})$$

A partir de A.7 et A.8, le produit de matrice à droite de A.6 s'écrit :

$$\begin{pmatrix}
 \sum_{i=1}^4 (\mathbf{A}_{i1}^t \Sigma_{x_{i1}}^{-1} \epsilon_{i1}) \\
 \sum_{i=1}^4 (\mathbf{A}_{i2}^t \Sigma_{x_{i2}}^{-1} \epsilon_{i2}) \\
 \sum_{i=1}^4 (\mathbf{A}_{i3}^t \Sigma_{x_{i3}}^{-1} \epsilon_{i3}) \\
 \sum_{i=1}^4 (\mathbf{B}_{1j}^t \Sigma_{x_{1j}}^{-1} \epsilon_{1j}) \\
 \sum_{i=1}^4 (\mathbf{B}_{2j}^t \Sigma_{x_{2j}}^{-1} \epsilon_{2j}) \\
 \sum_{i=1}^4 (\mathbf{B}_{3j}^t \Sigma_{x_{3j}}^{-1} \epsilon_{3j}) \\
 \sum_{i=1}^4 (\mathbf{B}_{4j}^t \Sigma_{x_{4j}}^{-1} \epsilon_{4j})
 \end{pmatrix} \quad (\text{A.11})$$

Notons $\sum_{i=1}^4 (\mathbf{A}_{ij}^t \Sigma_{x_{ij}}^{-1} \epsilon_{ij})$ et $\sum_{i=1}^4 (\mathbf{B}_{ij}^t \Sigma_{x_{ij}}^{-1} \epsilon_{ij})$ respectivement $\epsilon_{\mathbf{a}_j}$ et $\epsilon_{\mathbf{b}_i}$, A.11 s'écrit alors :

$$(\epsilon_{\mathbf{a}_1}^t \quad \epsilon_{\mathbf{a}_2}^t \quad \epsilon_{\mathbf{a}_3}^t \quad \epsilon_{\mathbf{b}_1}^t \quad \epsilon_{\mathbf{b}_2}^t \quad \epsilon_{\mathbf{b}_3}^t \quad \epsilon_{\mathbf{b}_4}^t)^t \quad (\text{A.12})$$

A.10 et A.11 donne l'écriture de l'équation normale A.6 suivante :

$$\begin{pmatrix}
 U_1 & 0 & 0 & W_{11} & W_{21} & W_{31} & W_{41} \\
 0 & U_2 & 0 & W_{12} & W_{22} & W_{32} & W_{42} \\
 0 & 0 & U_3 & W_{13} & W_{23} & W_{33} & W_{43} \\
 W_{11}^t & W_{12}^t & W_{13}^t & V_1 & 0 & 0 & 0 \\
 W_{21}^t & W_{22}^t & W_{23}^t & 0 & V_2 & 0 & 0 \\
 W_{31}^t & W_{32}^t & W_{33}^t & 0 & 0 & V_3 & 0 \\
 W_{41}^t & W_{42}^t & W_{43}^t & 0 & 0 & 0 & V_4
 \end{pmatrix}
 \begin{pmatrix}
 \delta_{\mathbf{a}_1} \\
 \delta_{\mathbf{a}_2} \\
 \delta_{\mathbf{a}_3} \\
 \delta_{\mathbf{b}_1} \\
 \delta_{\mathbf{b}_2} \\
 \delta_{\mathbf{b}_3} \\
 \delta_{\mathbf{b}_4}
 \end{pmatrix}
 =
 \begin{pmatrix}
 \epsilon_{\mathbf{a}_1} \\
 \epsilon_{\mathbf{a}_2} \\
 \epsilon_{\mathbf{a}_3} \\
 \epsilon_{\mathbf{b}_1} \\
 \epsilon_{\mathbf{b}_2} \\
 \epsilon_{\mathbf{b}_3} \\
 \epsilon_{\mathbf{b}_4}
 \end{pmatrix} \quad (\text{A.13})$$

Notons :

$$\mathbf{U}^* = \begin{pmatrix} \mathbf{U}_1^* & 0 & 0 \\ 0 & \mathbf{U}_2^* & 0 \\ 0 & 0 & \mathbf{U}_3^* \end{pmatrix} \quad (\text{A.14})$$

$$\mathbf{V}^* = \begin{pmatrix} \mathbf{V}_1^* & 0 & 0 & 0 \\ 0 & \mathbf{V}_2^* & 0 & 0 \\ 0 & 0 & \mathbf{V}_3^* & 0 \\ 0 & 0 & 0 & \mathbf{V}_4^* \end{pmatrix} \quad (\text{A.15})$$

$$\mathbf{W} = \begin{pmatrix} \mathbf{W}_{11} & \mathbf{W}_{21} & \mathbf{W}_{31} & \mathbf{W}_{41} \\ \mathbf{W}_{12} & \mathbf{W}_{22} & \mathbf{W}_{32} & \mathbf{W}_{42} \\ \mathbf{W}_{13} & \mathbf{W}_{23} & \mathbf{W}_{33} & \mathbf{W}_{43} \\ \mathbf{W}_{14} & \mathbf{W}_{24} & \mathbf{W}_{34} & \mathbf{W}_{44} \end{pmatrix} \quad (\text{A.16})$$

où * représente les éléments de la diagonale augmentée permettant d'écrire l'équation normale sous la forme compactée suivante :

$$\begin{pmatrix} \mathbf{U}^* & \mathbf{W} \\ \mathbf{W}^t & \mathbf{V}^t \end{pmatrix} \begin{pmatrix} \delta_{\mathbf{a}} \\ \delta_{\mathbf{b}} \end{pmatrix} = \begin{pmatrix} \epsilon_{\mathbf{a}} \\ \epsilon_{\mathbf{b}} \end{pmatrix} \quad (\text{A.17})$$

La multiplication de la partie gauche de A.17 par :

$$\begin{pmatrix} \mathbf{I} & -\mathbf{WV}^{*-1} \\ 0 & \mathbf{I} \end{pmatrix} \quad (\text{A.18})$$

donne :

$$\begin{pmatrix} \mathbf{U}^* - \mathbf{WV}^{*-1}\mathbf{W}^t & 0 \\ \mathbf{W}^t & \mathbf{V}^* \end{pmatrix} \begin{pmatrix} \delta_{\mathbf{a}} \\ \delta_{\mathbf{b}} \end{pmatrix} = \begin{pmatrix} \epsilon_{\mathbf{a}} - \mathbf{WV}^{*-1}\epsilon_{\mathbf{b}} \\ \epsilon_{\mathbf{b}} \end{pmatrix} \quad (\text{A.19})$$

Remarquons que le bloc haut droit de la matrice gauche est 0, ce qui permet de déterminer $\delta_{\mathbf{a}}$ à partir de :

$$(\mathbf{U}^* - \mathbf{WV}^{*-1}\mathbf{W}^t) \delta_{\mathbf{a}} = \epsilon_{\mathbf{a}} - \mathbf{WV}^{*-1}\epsilon_{\mathbf{b}} \quad (\text{A.20})$$

Connaissant $\delta_{\mathbf{a}}$, $\delta_{\mathbf{b}}$ peut être calculé en utilisant A.19 avec l'équation :

$$\mathbf{V}^* \delta_{\mathbf{b}} = \epsilon_{\mathbf{b}} - \mathbf{W}^t \delta_{\mathbf{a}} \quad (\text{A.21})$$

En substituant $\mathbf{U}^*$, $\mathbf{V}^*$ et $\mathbf{W}^*$ de l'équation A.16 dans la partie gauche de l'équation A.20 :

$$\begin{pmatrix} \mathbf{U}_1^* & 0 & 0 \\ 0 & \mathbf{U}_2^* & 0 \\ 0 & 0 & \mathbf{U}_3^* \end{pmatrix} - \begin{pmatrix} \mathbf{W}_{11} & \mathbf{W}_{21} & \mathbf{W}_{31} & \mathbf{W}_{41} \\ \mathbf{W}_{12} & \mathbf{W}_{22} & \mathbf{W}_{32} & \mathbf{W}_{42} \\ \mathbf{W}_{13} & \mathbf{W}_{23} & \mathbf{W}_{33} & \mathbf{W}_{43} \\ \mathbf{W}_{14} & \mathbf{W}_{24} & \mathbf{W}_{34} & \mathbf{W}_{44} \end{pmatrix} \begin{pmatrix} \mathbf{V}_1^{*-1} & 0 & 0 & 0 \\ 0 & \mathbf{V}_2^{*-1} & 0 & 0 \\ 0 & 0 & \mathbf{V}_3^{*-1} & 0 \\ 0 & 0 & 0 & \mathbf{V}_4^{*-1} \end{pmatrix} \begin{pmatrix} \mathbf{W}_{11}^t & \mathbf{W}_{12}^t & \mathbf{W}_{13}^t \\ \mathbf{W}_{21}^t & \mathbf{W}_{22}^t & \mathbf{W}_{23}^t \\ \mathbf{W}_{31}^t & \mathbf{W}_{32}^t & \mathbf{W}_{33}^t \\ \mathbf{W}_{41}^t & \mathbf{W}_{42}^t & \mathbf{W}_{43}^t \end{pmatrix} \quad (\text{A.22})$$

En notant $\mathbf{Y}_{ij} = \mathbf{W}_{ij} \mathbf{V}_i^{*-1}$, cette dernière équation devient :

$$\begin{pmatrix} \mathbf{U}_1^* - \sum_{i=1}^4 \mathbf{Y}_{i1} \mathbf{W}_{i1}^t & -\sum_{i=1}^4 \mathbf{Y}_{i1} \mathbf{W}_{i2}^t & -\sum_{i=1}^4 \mathbf{Y}_{i1} \mathbf{W}_{i3}^t \\ -\sum_{i=1}^4 \mathbf{Y}_{i2} \mathbf{W}_{i1}^t & \mathbf{U}_2^* - \sum_{i=1}^4 \mathbf{Y}_{i2} \mathbf{W}_{i2}^t & -\sum_{i=1}^4 \mathbf{Y}_{i2} \mathbf{W}_{i3}^t \\ -\sum_{i=1}^4 \mathbf{Y}_{i3} \mathbf{W}_{i1}^t & -\sum_{i=1}^4 \mathbf{Y}_{i3} \mathbf{W}_{i2}^t & \mathbf{U}_3^* - \sum_{i=1}^4 \mathbf{Y}_{i3} \mathbf{W}_{i3}^t \end{pmatrix} \quad (\text{A.23})$$

La partie droite de l'équation A.20 s'écrit donc :

$$\epsilon_{\mathbf{a}} - \begin{pmatrix} \sum_{i=1}^4 \mathbf{Y}_{i1} \epsilon_{\mathbf{b}_i} \\ \sum_{i=1}^4 \mathbf{Y}_{i2} \epsilon_{\mathbf{b}_i} \\ \sum_{i=1}^4 \mathbf{Y}_{i3} \epsilon_{\mathbf{b}_i} \end{pmatrix} \quad (\text{A.24})$$

En combinant les deux dernières équations, on peut extraire $\delta_{\mathbf{a}}$ de :

$$\begin{pmatrix} \mathbf{U}_1^* - \sum_{i=1}^4 \mathbf{Y}_{i1} \mathbf{W}_{i1}^t & - \sum_{i=1}^4 \mathbf{Y}_{i1} \mathbf{W}_{i2}^t & - \sum_{i=1}^4 \mathbf{Y}_{i1} \mathbf{W}_{i3}^t \\ - \sum_{i=1}^4 \mathbf{Y}_{i2} \mathbf{W}_{i1}^t & \mathbf{U}_2^* - \sum_{i=1}^4 \mathbf{Y}_{i2} \mathbf{W}_{i2}^t & - \sum_{i=1}^4 \mathbf{Y}_{i2} \mathbf{W}_{i3}^t \\ - \sum_{i=1}^4 \mathbf{Y}_{i3} \mathbf{W}_{i1}^t & - \sum_{i=1}^4 \mathbf{Y}_{i3} \mathbf{W}_{i2}^t & \mathbf{U}_3^* - \sum_{i=1}^4 \mathbf{Y}_{i3} \mathbf{W}_{i3}^t \end{pmatrix} \begin{pmatrix} \delta_{\mathbf{a}_1} \\ \delta_{\mathbf{a}_2} \\ \delta_{\mathbf{a}_3} \end{pmatrix} = \begin{pmatrix} \epsilon_{\mathbf{a}_1} - \sum_{i=1}^4 \mathbf{Y}_{i1} \epsilon_{\mathbf{b}_i} \\ \epsilon_{\mathbf{a}_2} - \sum_{i=1}^4 \mathbf{Y}_{i2} \epsilon_{\mathbf{b}_i} \\ \epsilon_{\mathbf{a}_3} - \sum_{i=1}^4 \mathbf{Y}_{i3} \epsilon_{\mathbf{b}_i} \end{pmatrix} \quad (\text{A.25})$$

$\delta_{\mathbf{b}}$ se déduit de A.21 :

$$\delta_{\mathbf{b}} = \begin{pmatrix} \mathbf{V}_1^{*-1} \left(\epsilon_{\mathbf{b}_1} - \sum_{j=1}^3 \mathbf{W}_{1j}^t \delta_{\mathbf{a}_j} \right) \\ \mathbf{V}_2^{*-1} \left(\epsilon_{\mathbf{b}_2} - \sum_{j=1}^3 \mathbf{W}_{2j}^t \delta_{\mathbf{a}_j} \right) \\ \mathbf{V}_3^{*-1} \left(\epsilon_{\mathbf{b}_3} - \sum_{j=1}^3 \mathbf{W}_{3j}^t \delta_{\mathbf{a}_j} \right) \\ \mathbf{V}_4^{*-1} \left(\epsilon_{\mathbf{b}_4} - \sum_{j=1}^3 \mathbf{W}_{4j}^t \delta_{\mathbf{a}_j} \right) \end{pmatrix} \quad (\text{A.26})$$

A ce stade du développement, il est évident que cette solution de l'équation normale, présentée ici pour un problème de taille réduite (3 vues, 4 points), peut être généralisée à n points et m vues. Cette méthode peut traiter de nombreux cas :

- si un point k n'apparaît pas dans une image l , on a $\mathbf{A}_{kl} = 0$ et $\mathbf{B}_{kl} = 0$ sans perturber l'algorithme. Un point n'est pas obligatoirement vu dans toutes les images.
- le vecteur a_i donne les paramètres de la caméra depuis laquelle la vue i a été acquise. Si cette caméra a une focale fixe, ce vecteur se décompose en intrinsèques (identiques pour toutes les vues acquises depuis cette caméra) et extrinsèques (position de la caméra par rapport au repère de l'environnement).
- Si le système visuel est un banc stéréo en mouvement comme dans nos applications, les vues sont associées deux par deux : les paramètres extrinsèques de la caméra gauche (par exemple) donnent la position du banc dans l'environnement ; les paramètres extrinsèques de la caméra droite sont exprimés en composant ceux de la caméra gauche avec une transformation rigide (identique pour toutes les vues) décrivant la configuration du banc.

Annexe B

Localisation de route

B.1 Présentation de l'approche choisie

Localiser la route sur laquelle évolue un véhicule, est évidemment un problème fondamental pour les applications de la vision dans les transports : les applications sont très nombreuses, notamment la prévention des sorties de route, causes de très nombreux accidents. De nombreuses approches ont déjà été développées ; il existe aujourd'hui des systèmes dédiés commercialisés par des équipementiers, systèmes uniquement exploitables sur des routes avec marquages.

La localisation de la route est également indispensable pour traiter de détection d'obstacles, afin d'éviter des fausses alarmes dues à la présence d'objets sur les bords de la chaussée. Nous devons donc disposer d'une méthode de détection de la route pour valider nos travaux en ce domaine. Nous avons analysé une approche image développée par [Chapuis et al., 2000], qui exploite un modèle de la route dans l'image afin de grouper les indices préalablement détectés ; ces indices sont exploités pour remettre à jour le modèle dans chaque image. Le développement d'une telle approche nécessite ensuite l'utilisation d'un modèle caractérisant l'ensemble route/véhicule afin de localiser le véhicule dans sa voie de circulation ; des extensions permettent de traiter aussi les changements de voie, ou de considérer des routes sans marquage. La solution développée³⁰ est une version simplifiée de l'approche initiale issue des travaux de R.Chapuis. Elle se décompose en 4 étapes : apprentissage, reconnaissance, localisation et suivi (voir figure B.1).

L'étape d'apprentissage permet de définir la position de la route sur l'image à partir des paramètres du modèle, c'est-à-dire les zones où se trouveront potentiellement les marquages de la route. Une fois les zones de recherche définies, l'étape de reconnaissance permet de déterminer précisément la position de la route dans l'image. Ces images sont typiquement acquises depuis une caméra embarquée au niveau du rétroviseur intérieur du véhicule expérimental. A partir de cette étape, la phase de localisation permet de calculer les paramètres du modèle de l'ensemble route/véhicule, à savoir :

- la position latérale du véhicule x_0 ,
- l'orientation du véhicule sur sa voie ϕ ,
- l'angle d'inclinaison de la caméra α ,
- la largeur de la route L ,
- la courbure latérale de la route C_l .

³⁰Travail issu du stage de Damien Quinton dans le cadre de son stage de 4ème année INSA Toulouse Génie Informatique

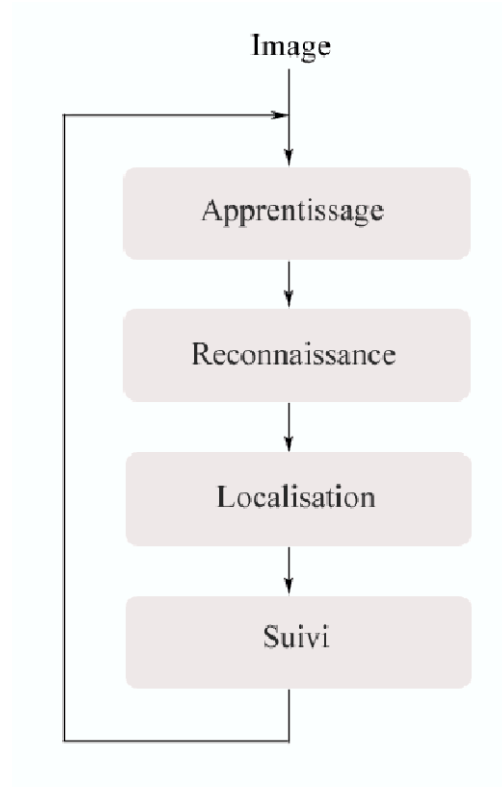


FIG. B.1 – Organigramme de l'approche

Enfin, la phase de suivi permet de faire évoluer les paramètres du modèle déterminés précédemment en fonction des variables dynamiques du véhicule et de la fréquence de capture des images. Cette étape gère également les changements de voies ainsi que les pertes de détection qui peuvent être les conséquences par exemple d'un manque de marquage au sol, d'une déformation de la chaussée ou bien du passage d'un véhicule dans le champ de la caméra.

B.2 Apprentissage et reconnaissance

B.2.1 Modèle de la route dans l'image

Le modèle exploité dans cette approche, doit permettre de positionner la route dans le repère image. Pour cela, il est composé d'un vecteur de position $\underline{X}_d$ et d'une matrice de covariance C_{Xd} associée à $\underline{X}_d$ définis de la façon suivante :

$$\underline{X}_d = \begin{pmatrix} u_{1g} \\ \cdot \\ u_{ng} \\ u_{1d} \\ \cdot \\ u_{nd} \end{pmatrix}, C_{Xd} = \begin{pmatrix} \sigma_{u_{1g}}^2 & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \sigma_{u_{ng}}^2 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \sigma_{u_{1d}}^2 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \sigma_{u_{nd}}^2 \end{pmatrix}$$

$(u_{1g} \dots u_{ng})$ et $(u_{1d} \dots u_{nd})$ représentent les coordonnées horizontales (colonnes) dans l'image pour chacun des deux bords de la route ou de la voie de circulation, pour les différentes lignes v_i (voir figure B.3). Pour limiter le temps de détection de la route, les n points caractérisant chaque bord de la route seront recherchés uniquement sur Nb_{ech} lignes échantillonnées dans l'image. Ce nombre Nb_{ech} peut varier en fonction des besoins de l'application ; il définit également la portée de la détection (voir figure B.2). Par exemple, pour les routes sinueuses ou encombrées par le trafic, il conviendra de limiter la portée de la détection afin d'éviter les perturbations. Au contraire, sur les voies relativement bien marquées et régulières comme les autoroutes, une grande portée sera préférable afin de profiter d'un maximum d'information sur la position de la route. Il est évident que plus le nombre de points Nb_{ech} est élevé, plus la détection des marquages et la modélisation de la route est précis, mais plus les temps d'exécution sont élevés. La portée de la détection est donnée par la formule suivante :

$$D_{max} = \frac{-z_0}{\frac{v_1}{e_v} - \alpha}$$

avec α et z_0 respectivement inclinaison et hauteur de la caméra, v_1 indice de la ligne traitée la plus haute dans l'image et e_v paramètre de discrétisation en ligne de la caméra³¹. Ainsi la portée est de 13.64 m pour $Nb_{ech} = 6$, de 26.76m pour $Nb_{ech} = 8$ et de 36.29 m pour $Nb_{ech} = 10$. L'augmentation de cette dernière portée reste difficile en raison des déformations de la route, de la courbure des virages, et de la faible apparition du marquage sur les images capturées à cette distance. Toutefois, l'application permet de changer dynamiquement le nombre de points échantillonnés sur les marquages, ce qui permettra de changer la configuration de la détection en fonction du type de route.

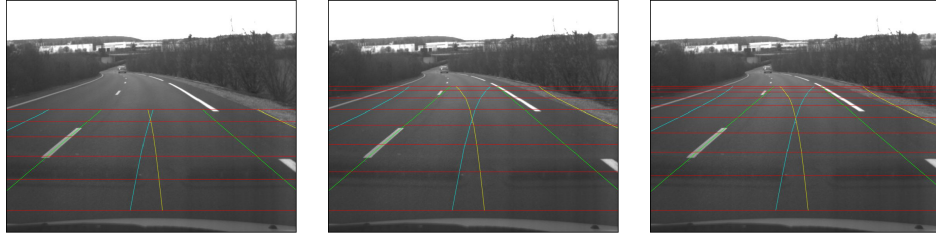


FIG. B.2 – Modèle pour différentes valeurs du nombre d'échantillons

La matrice C_{Xd} définit les variations possibles de u_{ig} et u_{id} . La valeur initiale du modèle $(\underline{X}_d, C_{Xd})$ est obtenue lors de la phase d'apprentissage.

Pour chaque bord de la route, un intervalle de confiance centré sur $\underline{X}_d$ est représenté, et dont la largeur est déduite de C_{Xd} . Les lignes v_i sont également sélectionnées (voir figure B.3).

B.2.2 Apprentissage

Contrairement au modèle défini en [Aufrere et al., 2000], la phase d'apprentissage intervient toujours au début de chaque phase de détection, ce qui permet de générer $\underline{X}_d(k+1)$ et $C_{Xd}(k+1)$ à partir des paramètres $(x_0, \phi, C_l, \alpha, L)$ calculés lors de la phase de suivi pour l'image k . Pour cela, on utilise les équations qui expriment les coordonnées u_i en fonction de $(v_i, z_0, x_0, \phi, \alpha, C_l, L)$ dans [Chapuis et al., 2000] :

³¹Paramètre noté α_u dans ce mémoire ; nous avons préféré garder les notations de les articles de R.Chapuis pour simplifier la lecture de cette annexe.

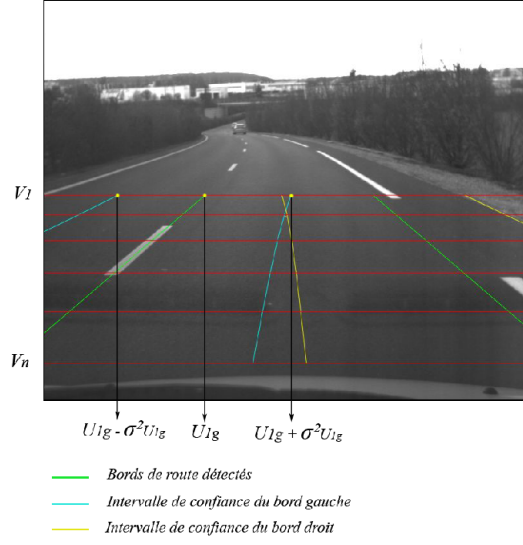


FIG. B.3 – Représentation du modèle

$$\begin{cases} u_{ig} = e_u \left(-\frac{e_v z_0}{2(v_i - e_v \alpha)} C_l + \frac{v_i - e_v \alpha}{e_v z_0} x_0 - \phi \right) \\ u_{id} = e_u \left(-\frac{e_v z_0}{2(v_i - e_v \alpha)} C_l + \frac{v_i - e_v \alpha}{e_v z_0} (x_0 - L) - \phi \right) \end{cases}$$

où :

$$e_u = \frac{f}{d_u} \text{ et } e_v = \frac{f}{d_v}$$

- f : focale de la caméra,
- d_u, d_v : largeur et hauteur d'un pixel,
- z_0 : hauteur de la caméra.

Il est à noter que les approximations faites en [Chapuis et al., 2000] sur les petits angles $\sin(\theta) = \theta$ ne sont pas acceptables pour $6 \leq \alpha \leq 10$ et $0 \leq |\phi| \leq 8$. Les approximations que nous faisons sur les angles, sont donc les suivantes :

$$\sin(\alpha) \equiv \frac{\alpha}{57.4}, \sin(\phi) \equiv \frac{\phi}{57.3}$$

Pour ne pas surcharger les notations dans cette annexe, nous conviendrons que $\alpha = \frac{\alpha}{57.4}$ et $\phi = \frac{\phi}{57.3}$. Pour le calcul de $\underline{X}_d(0)$ et $C_{Xd}(0)$, on utilise les deux équations précédentes avec le vecteur d'état suivant comme exemple :

$$\underline{X}_{l_init} = \begin{pmatrix} x_0 \\ \phi \\ C_l \\ \alpha \\ L \end{pmatrix} = \begin{pmatrix} 1.75 \\ 0 \\ 0 \\ 8 \\ 3.5 \end{pmatrix} \text{ et } C_p = \begin{pmatrix} \sigma_{x_0}^2 & & & & \\ & \sigma_\phi^2 & & & \\ & & \sigma_{C_l}^2 & & \\ & & & \sigma_\alpha^2 & \\ & & & & \sigma_L^2 \end{pmatrix}$$

De plus, comme $C_{Xd} = J_g C_p J_g$ dans [Aufrere et al., 2000] avec J_g la matrice jacobienne, il suffit de définir des valeurs pour σ_i^2 dans C_p suffisamment grandes pour obtenir une matrice de covariance qui permettra de détecter la route sans connaître ses paramètres a priori en proposant une grande tolérance sur les estimées initiales des paramètres.

$$J_g = \begin{pmatrix} e_u \frac{v_1 - e_v \alpha}{e_v z_0} & -e_u & -\frac{e_v e_u z_0}{2(v_1 - e_v \alpha)} & -\frac{e_u e_v^2 z_0 C_l}{2(\alpha e_v - v_1)^2} - \frac{e_u x_0}{z_0} & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ e_u \frac{v_n - e_v \alpha}{e_v z_0} & -e_u & -\frac{e_v e_u z_0}{2(v_n - e_v \alpha)} & -\frac{e_u e_v^2 z_0 C_l}{2(\alpha e_v - v_n)^2} - \frac{e_u x_0}{z_0} & 0 \\ e_u \frac{v_{n+1} - e_v \alpha}{e_v z_0} & -e_u & -\frac{e_v e_u z_0}{2(v_{n+1} - e_v \alpha)} & -\frac{e_u e_v^2 z_0 C_l}{2(\alpha e_v - v_{n+1})^2} - \frac{e_u (x_0 - L)}{z_0} & -e_u \frac{v_{n+1} - e_v \alpha}{e_v z_0} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ e_u \frac{v_{2n} - e_v \alpha}{e_v z_0} & -e_u & -\frac{e_v e_u z_0}{2(v_{2n} - e_v \alpha)} & -\frac{e_u e_v^2 z_0 C_l}{2(\alpha e_v - v_{2n})^2} - \frac{e_u (x_0 - L)}{z_0} & -e_u \frac{v_{2n} - e_v \alpha}{e_v z_0} \end{pmatrix}$$

Lorsque les paramètres du modèle $(x_0, \phi, C_l, \alpha, L)$ issus de l'étape de localisation sont connus, il est possible de restreindre l'espace de recherche caractérisé par C_{Xd} , en diminuant les valeurs σ_i^2 . On obtient ainsi :

$$C_{p_init} = \begin{pmatrix} 1.75^2 & & & \\ & 5^2 & & 0 \\ & & 0.004^2 & \\ & 0 & & 2^2 \\ & & & & 0.6^2 \end{pmatrix}, C_{p_suivi} = \begin{pmatrix} 0.20^2 & & & \\ & 2^2 & & 0 \\ & & 0.002^2 & \\ & 0 & & 0.5^2 \\ & & & & 0.20^2 \end{pmatrix}$$

D'autres valeurs sont attribuées à C_p lorsque le véhicule change de voie ou bien lorsque la route n'est plus détectée car dans ces deux cas, l'intervalle de recherche doit être élargi.

B.2.3 Procédure de recherche

L'objectif de cette procédure est de rechercher une valeur optimale de $\underline{X}_d$ représentant la route dans l'image. Pour cela, on parcourt un arbre de recherche basé sur l'exploration de différentes zones caractérisées par quatre points $P1(u_i - \sigma_i, v_i)$, $P2(u_i + \sigma_i, v_i)$, $P3(u_{i+1} - \sigma_{i+1}, v_{i+1})$, et $P4(u_{i+1} + \sigma_{i+1}, v_{i+1})$; c'est dans ces zones que seront successivement recherchés les marquages ou les bords de la route, à droite ou à gauche, la n_{ieme} recherche exploitant les résultats des $(n - 1)$ zones traitées précédemment. Ainsi pour $2n$ points définissant la route, il y a $2n - 2$ zones. La solution optimale sera le plus souvent obtenue par une maximisation du nombre de zones dans lesquelles le marquage ou le bord de la route a été détecté.

Dans [Aufrere et al., 2000], c'est la zone de plus petite variance de la diagonale de C_{Xd} qui définit la prochaine zone d'intérêt. Dans notre application, les zones d'intérêt sont classées dans une liste fixée au départ mais évoluant au cours de la recherche (voir figure B.4). Ainsi la détection débutera toujours par les zones situées dans le bas de l'image privilégiant les lignes proches et supposées plus faciles à détecter que les lignes lointaines. Les zones d'intérêt suivantes seront celles restantes sur le bord gauche puis enfin sur le bord droit. Le côté gauche est donc privilégié par rapport au côté droit car généralement, celui-ci correspond au milieu de la chaussée qui est souvent mieux marqué et surtout indépendant des déformations situées sur le bord de la route.

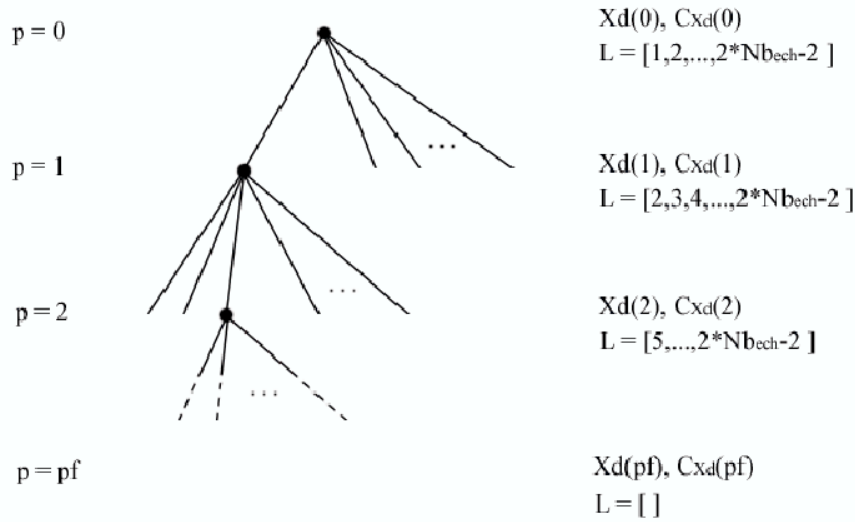


FIG. B.4 – Exemple d'évolution de l'algorithme

L'algorithme consiste donc à analyser de façon séquentielle les zones de la liste. A chaque étape, il est nécessaire de déterminer la présence ou non d'un bord de route dans la zone d'intérêt, et de le caractériser. Dans la version courante, la détection des bords de route est basée uniquement sur la recherche d'un marquage au sol, qui correspond le plus souvent à des lignes ou courbes blanches qui limitent une voie de circulation. C'est dans ce contexte que se place l'application. Cette hypothèse restreint les cas possibles de détection comme les routes temporaires avec marquage jaune et les routes non marquées. L'évolution vers ces types de détection reste cependant facile, notamment en exploitant les travaux de G.Avina sur la couleur et la texture [Avina-Cervantes, 2005].

La détection d'une ligne blanche dans une zone d'intérêt revient à rechercher les zones de fort contraste. Pour cela, pour chaque ligne de l'image appartenant à cette zone, le premier point de gradient maximum est localisé à l'aide d'un filtre linéaire semblable à un filtre de Sobel (voir figure B.5), mais adapté à l'orientation des lignes recherchées à droite ou à gauche. Le fait de garder seulement le premier point de gradient maximum permet de ne prendre en compte que les lignes les plus proches de la voiture, puisque le sens de calcul va du centre de l'image vers les bords. Une amélioration pourrait porter sur la caractérisation de plusieurs lignes blanches à l'intérieur même d'une seule zone d'intérêt.

A l'ensemble des points ainsi constitués, un segment de droite est ajusté par la méthode des moindres carrés médians. Ce calcul est effectué à condition de disposer d'au moins un certain pourcentage de points afin de limiter les perturbations dues à une faible précision. On procède alors à un certain nombre de tirages aléatoires de 60% des points relevés afin de garder l'équation du segment qui présentera le plus faible résidu sur la totalité des points. De plus, cette méthode tient compte d'une contrainte sur la pente du segment détecté.



FIG. B.5 – Filtre utilisé pour la mesure du gradient

Dans le cas où aucun segment n'est détecté dans la zone d'intérêt, une détection est tentée dans la zone suivante de la liste. Au contraire, si une détection a été concluante, celle-ci fournit les deux points d'intersection qui seront utilisés par le filtre de Kalman dégénéré [Aufrere et al., 2000] afin de mettre à jour $\underline{X}_d$ et C_{Xd} .

$$\begin{cases} \underline{X}_d(p) = \underline{X}_d(p-1) + K_d[\underline{y} - \underline{x}(p-1)] \\ C_{Xd}(p) = C_{Xd}(p-1) - K_d H_d C_{Xd}(p-1) \end{cases}$$

avec :

- $K_d = C_{Xd}(p-1)H_d^t[H_d C_{Xd}(p-1)H_d^t + R]^{-1}$,
- $\underline{x} = (u_i, u_{i+1})^t$,
- H_d telle que $\underline{x} = H_d \underline{X}_d$,
- R : Matrice de covariance de l'erreur de détection, qui vaut pour une erreur de n pixels $R = n^2 I$ où I est une matrice identité (2x2).

Lorsque toutes les zones ont été explorées à une profondeur quelconque p de l'arbre de recherche, l'algorithme reprend à la profondeur $p=0$ en considérant la zone suivante, et non à la profondeur $p-1$ comme dans [Aufrere et al., 2000]. Toutes les branches de l'arbre ne sont donc pas explorées, ce qui signifie que certaines combinaisons traduisant un ordre de détection des zones d'intérêt sont ignorées. En procédant ainsi, la solution optimale est retrouvée ou du moins très approchée. Les temps de calcul sont ainsi très largement réduits dans les cas où aucun critère de fin n'est valide, car cela implique de parcourir la totalité de l'arbre de recherche.

Le processus de recherche s'arrête lorsqu'un critère est atteint et que l'algorithme revient à la profondeur $p=0$ car toutes les zones ont été explorées. Ce critère est atteint si au moins le

marquage a été détecté dans une zone de chaque bord et dans $\frac{2Nb_{ech}}{3}$ zones au total ou au moins dans deux zones de chaque bord et $Nb_{ech} - 1$ zones au total.

Une amélioration pourrait être de prendre en compte, dans le critère pour l'image k , le nombre de zones détectées à l'image $k-1$ afin d'avoir une approximation du résultat attendu.

B.3 Localisation et suivi

B.3.1 Localisation du véhicule

Un des objectifs de l'application est de pouvoir connaître la position du véhicule sur la route. Cela permet d'analyser le comportement du véhicule (détecter les sorties de route) et du conducteur (état de vigilance, type de conduite...); en effet, la connaissance de la position latérale du véhicule x_0 sur la route, couplée à des systèmes de surveillance d'hypovigilance, permettra d'estimer l'état de conscience du conducteur. Par ailleurs, nous verrons dans la section suivante, que la connaissance des paramètres du modèle route/véhicule va permettre de fournir un nouvel espace de recherche pour la détection des bords de la route ou des marquages dans l'image suivante.

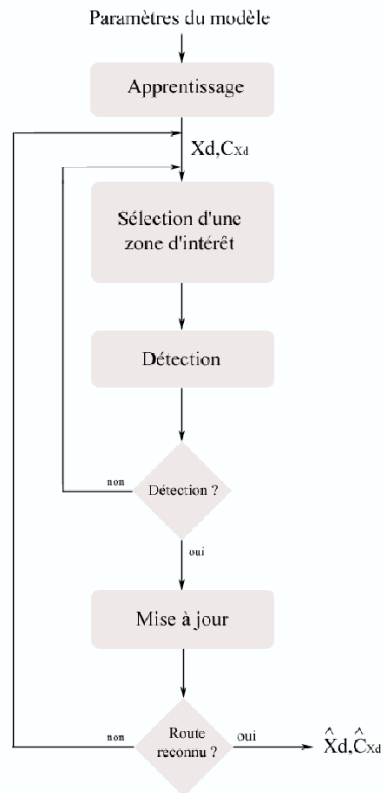


FIG. B.6 – Organigramme de l'étape de reconnaissance

La procédure de reconnaissance présentée en figure B.6 fournit, pour l'image k , un vecteur d'observation $\hat{X}_d(k)$ et sa matrice de covariance $\hat{C}_{Xd}(k)$. La mise à jour du vecteur de $\underline{X}_d(k)$ est

effectuée selon les équations d'un filtre de Kalman identique à [Aufrere et al., 2000] auquel est ajouté un gain $K_g < 1$ afin d'atténuer les variations brutales de la route.

$$\underline{X}_d(k) = \underline{X}_d(k-1) + K_g K [\hat{X}_d(k) - H \underline{X}_d(k-1)]$$

avec :

- $K = C_{Xd}(k-1)H^t[HC_{Xd}(k-1)H^t + C_{Xd}(k)]^{-1}$,
- H telle que $\hat{X}_d(k) = H \underline{X}(k) + \underline{w}$,
- $\underline{X}_d(k-1)$: Vecteur d'observation issu de l'étape d'apprentissage,
- $C_{Xd}(k-1)$: Matrice de covariance issue de l'étape d'apprentissage,
- $\hat{X}_d$: Vecteur d'observation issu de l'étape de reconnaissance,
- $C_{Xd}(k)$: $[ww^t]$.

Pour calculer les paramètres à partir de $X_d(k)$, on procède en trois étapes. Tout d'abord le calcul de l'angle d'inclinaison de la caméra α se fait de façon empirique. En connaissant la largeur en pixels de la route en bas (L_1) et au milieu (L_2) de l'image déduite du vecteur d'observation $\underline{X}_d(k)$, et en faisant le rapport $r = L_2/L_1$, il est possible de retrouver l'angle α puisque la largeur de la route est considérée constante sur toute la portée de la détection. Le lien entre ce rapport r et α se traduit par un polynôme $\alpha = f(r)$ d'ordre 3. Ce dernier dépend donc de la portée de la recherche. Il est possible de retrouver ce polynôme en mesurant les valeurs de r pour plusieurs valeurs de l'angle α . A partir de ces couples d'échantillons, l'inclinaison α se calcule par la méthode des moindres carrés à l'ordre désiré. L'approximation est très bonne et le temps de calcul très réduit par rapport à un calcul matriciel. Connaissant α , il est alors possible de calculer la largeur de la route L grâce à la formule suivante.

$$L = -\frac{e_v z_0 (u_{nd} - u_{ng})}{e_u (v_n - e_v \alpha)}$$

Enfin, il reste à calculer les trois derniers paramètres ϕ, x_0, C_l . Pour celà, on résoud le système linéaire à 3 inconnues suivant déduit des deux équations exprimant les coordonnées u_i en prenant trois points de $\underline{X}_d$.

$$\begin{pmatrix} \frac{v_i - e_v \alpha}{e_v z_0} & -1 & -\frac{e_v z_0}{2(v_i - e_v \alpha)} \\ \frac{v_j - e_v \alpha}{e_v z_0} & -1 & -\frac{e_v z_0}{2(v_j - e_v \alpha)} \\ \frac{v_k - e_v \alpha}{e_v z_0} & -1 & -\frac{e_v z_0}{2(v_k - e_v \alpha)} \end{pmatrix} \begin{pmatrix} x_0 \\ \phi \\ C_l \end{pmatrix} = \begin{pmatrix} \frac{u_{id}}{e_u} + \frac{v_i - e_v \alpha}{e_v z_0} L \\ \frac{u_{jd}}{e_u} + \frac{v_j - e_v \alpha}{e_v z_0} L \\ \frac{u_{kd}}{e_u} + \frac{v_k - e_v \alpha}{e_v z_0} L \end{pmatrix}$$

B.3.2 Suivi de la route

A partir de l'information présente à l'image k , nous cherchons, dans cette étape, une approximation des bords ou marquages dans l'image suivante $k+1$ afin de pouvoir restreindre l'intervalle de confiance pour l'étape de reconnaissance. Il s'agit de mettre à jour le vecteur de paramètres $\underline{X}_l = (x_0, \phi, C_l, \alpha, L)^t$ grâce à la matrice d'évolution M :

$$\underline{X}_l(k+1) = M \underline{X}_l(k) \text{ avec } M = \begin{pmatrix} 1 & \Delta_y & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Δ_y représente le déplacement du véhicule entre deux images, qui peut être calculé en connaissant la vitesse du véhicule et le temps entre deux acquisitions d'images par l'application. Ainsi on obtient :

$$\Delta_y = Vitesse \Delta_t$$

Lors de cette étape, on vérifie que les paramètres restent dans un domaine acceptable défini par des valeurs limites. Quand un paramètre sort de son domaine de validité, on réinitialise les paramètres avec des valeurs moyennes calculées à partir d'un certain nombre d'images déjà analysées dont les résultats sont consignés dans un historique. En cas de nouvel échec, on réinitialise la recherche $\underline{X}_l = \underline{X}_l_{init}$.

La gestion du changement de voie s'effectue en vérifiant la valeur de x_0 :

$$\begin{cases} \text{si } x_0(k) > L \Rightarrow x_0(k+1) = x_0(k) - L \\ \text{si } x_0(k) < 0 \Rightarrow x_0(k+1) = x_0(k) + L \end{cases}$$

B.4 Résultats

Afin de valider l'approche dans un contexte temps réel, l'algorithme a été implémenté dans le logiciel *RTmaps*. La méthode a fait l'objet d'une étude sur des milliers d'images issues du projet *Arcos* (voir figure B.7). Les figures B.8, B.9 et B.10 représentent les données relevées pour quelques évolutions du véhicule.

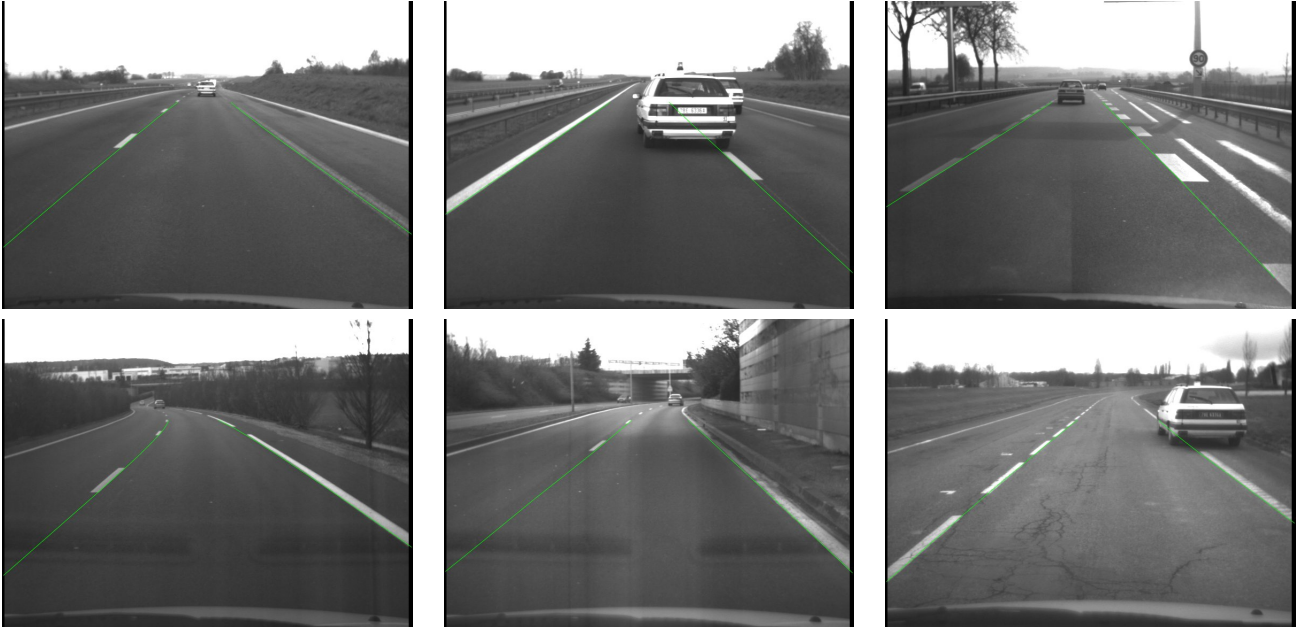


FIG. B.7 – Résultats sur les images Arcos

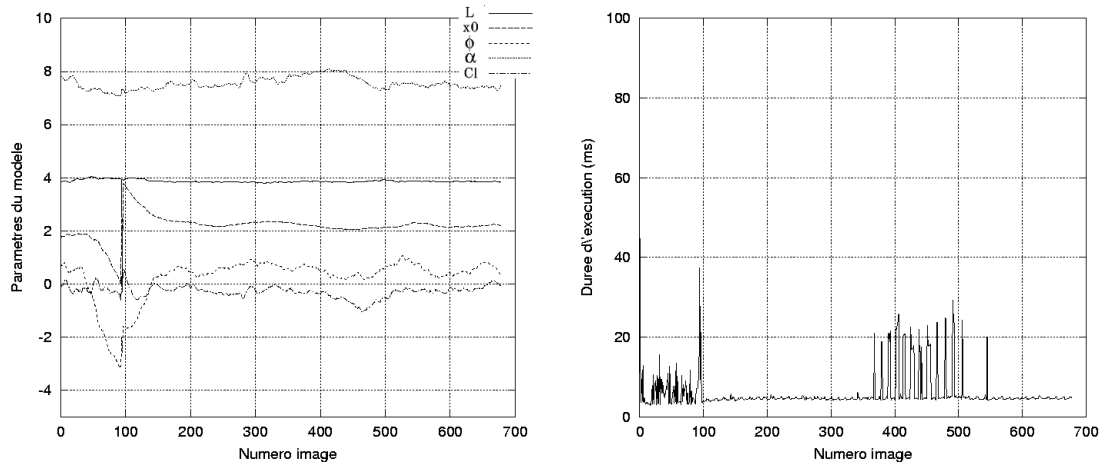


FIG. B.8 – Changement de voie

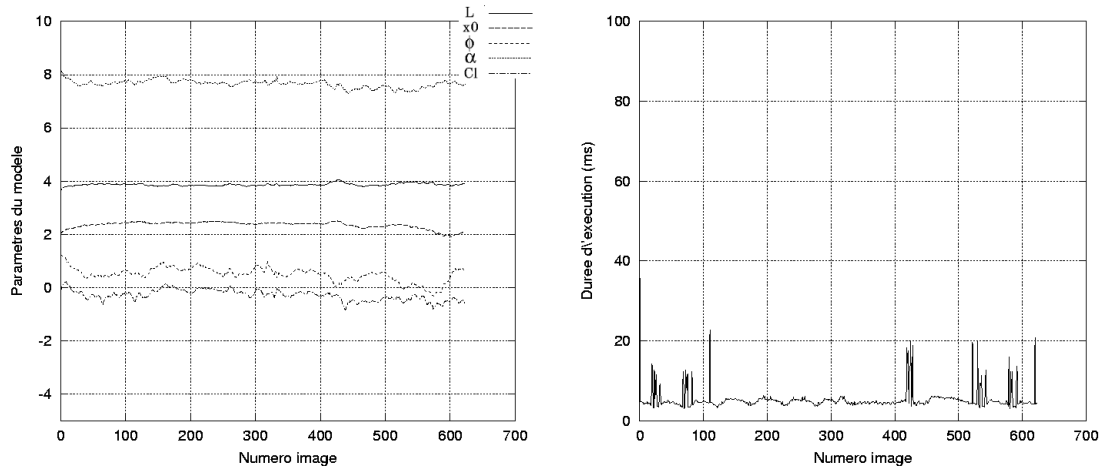


FIG. B.9 – Ligne droite

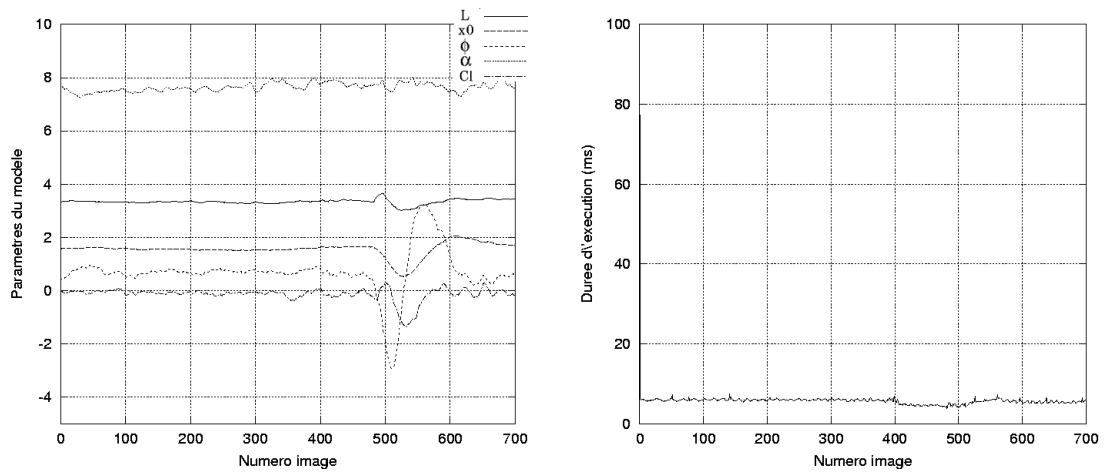


FIG. B.10 – Ecart brutal du véhicule

Bibliographie du chapitre

- [Aufrere et al., 2000] R. Aufrere, F. Marmoiton, R. Chapuis, F. Collange, et J. Derutin (2000). Road detection and vehicles tracking by vision for acc. *Traitement du signal*.
- [Avina-Cervantes, 2005] G. Avina-Cervantes (2005). *Navigation visuelle d'un robot mobile dans un environnement d'extérieur semi-structuré*. Thèse de doctorat, Institut National Polytechnique de Toulouse.
- [Chapuis et al., 2000] R. Chapuis, F. Marmoiton, et R. Aufrere (2000). Road detection and vehicles tracking by vision for an on-board acc system in the velac vehicle. Dans *3rd International Conference on Information Fusion*, volume 2.
- [Hartley et al., 2000] R. Hartley et A. Zisserman (2000). *Multiple view geometry in computer vision*.
- [Hiebert, 1981] K. Hiebert (1981). An evaluation os mathematical software that solves nonlinear least squares problems. Dans *ACM Trans. on Math Software*.
- [Levenberg, 1944] K. Levenberg (1944). A method for the solution of certain non-linear problems in least squares. *Quaterly of Applied Mathematics*, 2 :164–168.
- [Marquardt, 1963] D. Marquardt (1963). An algorithm for the least-squares estimation of non-linear parameters. *SIAM Journal of applied mathematics*, pages 431–441.

Résumé

La complexité croissante de l'environnement routier et le souci d'amélioration de la sécurité routière expliquent un nombre important de travaux sur l'aide à la conduite. De nombreux systèmes (ultrasons, laser, radar, ...) ont déjà été mis en œuvre pour percevoir l'environnement autour d'un véhicule. Tous se heurtent à la diversité des scènes observées, que ce soit en milieu urbain ou routier. Il est donc nécessaire d'exploiter des capteurs qui donnent une information riche pouvant être analysée dans plusieurs fonctions et quelle que soit la configuration. La vision semble tout à fait adéquate; elle est d'ailleurs une source sensorielle capitale dans la prise de décision chez l'humain. Les méthodes décrites dans cette thèse consistent donc à proposer un système complet et autonome de stéréovision passive pour la détection d'obstacles sur un véhicule, dit intelligent, évoluant en milieu urbain ou routier.

La problématique majeure relative à l'exploitation d'un banc de stéréovision sur un robot ou un véhicule concerne la qualité du calibrage qui tend à varier au cours du temps. Un tel système n'est en effet viable que si les paramètres qui décrivent la géométrie du capteur peuvent être correctement estimés, y compris en présence de chocs et vibrations. Nous présentons donc d'abord une méthode de recalibrage en ligne d'un système initialement calibré, consistant en une correction dynamique de ses paramètres. Affranchi des problèmes de calibrage, le deuxième problème abordé est la détection d'obstacles en milieu routier. Les méthodes décrites dans ce document proposent une construction du profil longitudinal de la route afin d'en extraire un modèle d'obstacle tenant compte des variations de pente et du dévers et/ou roulis. Notre approche exploite les données 3D fournies par la stéréo-vision. La dernière application concerne un système de perception pour l'aide au parking; nous présentons une méthode générique de construction incrémentale d'un modèle 3D de l'espace libre. Cette méthode suppose que le déplacement du véhicule est estimé avec une bonne précision. Le modèle est alors analysé pour détecter et caractériser une place de parking; le but est d'alimenter un planificateur de mouvement permettant la réalisation automatique de la manœuvre.

Toutes les méthodes ont été intégrées dans l'environnement RT-MAPS, et ont été validées soit en ligne sur le démonstrateur du LAAS, soit sur des séquences d'images acquises sur des véhicules.

Mots-clés: Stéréovision passive, recalibrage en ligne, aide à la conduite, détection d'obstacles en milieu routier, modélisation de la place de parking.

Abstract

The increasing complexity of road environments and the will to improve driving safety may explain the large number of studies (investigations / research programs) on driving assistance. Numerous systems (based on ultrasonic sensors, laser range, radar,...) have been set up to perceive the environment around a vehicle. All of them have to deal with the diversity of scenes observed, both in urban and road contexts. Therefore, one has to use sensors providing rich information that can be analysed in several functions and in any configuration. Vision seems to be a relevant choice. It is indeed an essential sensorial source for human's decision making process. Thus, the methods described in this thesis consist in proposing a complete and autonomous passive stereovision system for obstacles detection on a so-called "intelligent" vehicle, evolving in urban or road environments. The main issue about using a stereovision bench on board a robot or a vehicle concerns the calibration accuracy, which tends to vary with time. Indeed, such a system can only be reliable if the parameters describing the sensor's geometry can be correctly estimated, in spite of vibrations and possible shocks.

First, an on-line recalibration method for an initially calibrated system will be presented. It consists in a dynamic correction of its parameters. Once calibration issue is tackled, the second one considered is obstacles detection in road environment. The methods described in this document propose a building of the longitudinal profile of the road so as to extract an obstacle model from it, which takes into account the variations in pitch (superelevation) and/or roll. This approach processes 3D data provided by stereovision. The last application concerns a perception system for parking assistance; a generic method for an incremental building of a 3D model of free space will be presented. That method assumes that the vehicle movement is estimated with a good precision. The 3D model obtained is then analysed to detect a parking space and estimate its characteristics; the goal is to provide information to a motion planning system enabling an automatic achievement of the parking manoeuvring.

Every method has been integrated in the RT-MAPS environment, and validated on-line on board one of the LAAS robots or with images sequences taken on board vehicles.

Keywords: Stereovision, on-line recalibration, car assistance, on road obstacle detection, parking modeling.

