THÈSE

préparée à

L'INRIA Sophia Antipolis

et présentée à

L'UNIVERSITÉ de NICE - SOPHIA ANTIPOLIS

pour obtenir le grade de

DOCTEUR EN SCIENCES

Spécialité

Sciences de l'Ingénieur

soutenue publiquement par

François-Xavier ESPIAU

Sujet de la thèse :

Métrologie 3D par vision active sur des objets naturels sous-marins

le 26 Février 2002 devant le jury composé de :

M.	**Pierre**	**BERNHARD**	Président
MM.	**Michel**	**DHOME**	Rapporteurs
	Radu	**HORAUD**	
MM.	**Patrick**	**RIVES**	Examinateurs
	Vincent	**RIGAUD**	
	Boguslaw	**LORECKI**	

Remerciements

Je remercie tout d'abord M. le Professeur Pierre Bernhard d'avoir accepté la présidence du jury, MM. Radu Horaud et Michel Dhome d'avoir été mes rapporteurs et M. Lorecki de la société Cybernétix, d'avoir pris de son temps pour examiner ma thèse.
Je remercie tout particulièrement M. Patrick Rives, le directeur de cette thèse, qui a su se montrer attentif et patient durant ces trois années. Je remercie chaleureusement M. Vincent Rigaud mon responsable scientifique à l'Ifremer de Toulon qui m'a donné l'opportunité de faire ce travail de recherche et avec qui mes relations furent toujours enrichissantes.

Je tiens également à remercier les membres du projet Icare que j'ai pu cotoyer durant ces années, Claude, Pascal, Alessandro, Guillaume, Agnès, David, Hellal, et les nouveaux arrivants, Guillaume, Nicolas, Alexandre et Matthieu, avec qui nous avons eu de nombreux échanges constructifs. Je remercier particulièrement Jean-Jacques pour sa disponibilité et pour avoir pris le temps de répondre à mes questions singulières en matière de programmation. Une mention spéciale à Ezio, jeune papa, mon co-bureau, avec qui, nos échanges sur les plans professionnels et amicaux furent toujours constructifs et j'espère qu'ils continueront à l'être. Je souhaite remercier aussi les nombreux autres thésards des différents projets et plus particulièrement Fred le robotvisien pour son important soutien logistique à la fin de cette thèse.

Je remercie également tous mes amis des montagnes, de Grenoble à Valmorel, pour les soirées et les moments inoubliables et qui m'ont permis de décompresser et de prendre le recul nécessaire, quand il le fallait. Il n'est pas concevable de ne pas citer ici, les compagnons de mes nombreuses nuits de rédaction, qui, par leur musique m'ont tenu éveillé : Mlle Björk, Mr Thompson, Mr Jarvis, Mr David et les autres.
Je veux remercier ici mes parents et ma petite soeur Camille pour leur soutien infaillible durant toutes ces années.

Enfin, je remercie celle qui est devenue ma femme, Audrey.

Notations

Typographie

- les vecteurs sont représentés par des minuscules et notés en **gras** ($\mathbf{x}$, $\mathbf{b}$)
- les matrices sont représentées par des majuscules et notés en **gras** ($\mathbf{A}$, $\mathbf{K}$, $\mathbf{H}$)

Opérateurs mathématiques

- $\mathbf{x} \wedge \mathbf{y}$ représente le produit vectoriel des vecteurs $\mathbf{x}$ et $\mathbf{y}$
- $\otimes$ représente l'opérateur de convolution
- $\mathbf{I}$ représente la matrice identité
- $\mathbf{A}^{-\mathbf{1}}$ représente l'inverse de la matrice $\mathbf{A}$
- $\mathbf{A}^{\mathbf{T}}$ représente la transposée de la matrice $\mathbf{A}$

Symboles réservés

- $I(\mathbf{x})$ représente l'intensité de l'image au point $\mathbf{x} = [u\ v]^T$
- I_u représente la dérivée de I par rapport à u
- $G(x, \sigma)$ représente une gaussienne en 2D

Table des matières

Table des figures

Introduction

Le formalisme de la vision par ordinateur est-il réellement transposable aux applications concrètes ? Si oui, quel niveau de difficulté pouvons-nous atteindre ? Se basant sur ces intérogations, nous allons montrer dans cette thèse que cette question est de plus en plus légitime car la vision s'intègre de plus en plus dans des processus de traitement d'images d'environnements réels. Néanmoins le monde qui nous entoure est difficile à modéliser et les environnements naturels nous posent encore de nombreux problèmes.

1 Vers l'application de la théorie de la vision par ordinateur

Les progrès de l'informatique en terme de puissance de calcul, de réduction des volumes et de l'énergie consommée, permettent d'envisager aujourd'hui d'embarquer sur des engins robotiques des fonctions sophistiquées de traitement d'images avec des performances proches du temps réel vidéo. Cette capacité de traitement ouvre de larges perspectives en terme d'applications tant dans le domaine de l'analyse de scènes et de la modélisation d'objets que dans le domaine de l'utilisation d'informations vidéos dans des boucles de commande. Elle va permettre également de confronter les avancées théoriques et les efforts de formalisation qui ont eu lieu dans les dix dernières années à la dure réalité de la complexité du monde réel.

En effet, la théorie de la vision artificielle n'a réellement trouvé ses fondements que récemment avec en parallèle la compréhension de la nature même de l'image en terme de signal et la formalisation de la géométrie sous-jacente à l'acquisition des images.

L'enjeu des années qui viennent va être de montrer que ces modèles mathématiques et photométriques sont effectivement à même d'appréhender la complexité d'images réelles parfois fortement dégradées.

Toujours pour les mêmes raisons liées au progrès de l'informatique et du fait de la "maturité" de la théorie de la vision, on assiste aujourd'hui à une explosion des domaines applicatifs dont les plus connus du grand public sont, bien sûr, la création audiovisuelle. Il en existe bien d'autres, comme par exemple, l'aide à la conduite ou la conduite automatique de véhicule, la construction de modèle tridimensionnel comme dans le cas d'images géologiques ou médicales, la détection et la caractérisation du mouvement d'objets ou de véhicules, l'indexation d'images ou de séquences vidéos dans des bases de données, ...

Clairement, autant sur un plan théorique que sur le plan des applications, la vision par ordinateur se situe au carrefour de nombreuses sciences de l'Ingénieur, telles que les mathématiques fondamentales et appliquées, l'intelligence artificielle, le traitement de signal, l'automatique et l'informatique. De plus, l'expansion des technologies de l'information va amener des nouvelles problématiques, comme par exemple, les problèmes liés à la videoconférence via Internet.

2 Cadre applicatif et but de ce travail

Malgré les avancées, les grands problèmes de la vision par ordinateur ne sont toujours qu'en partie résolus. Si la théorie a énormement progressé, il n'en reste pas moins que le passage à la pratique peut devenir très rapidement délicat, voire impossible.

Celà est dû en grande partie à la nature projective de la vision qui conduit le plus souvent à des problèmes mal conditionnés (au sens d'Hadamard) dont la résolution se révèle instable. Parmi les problèmes canoniques de la vision par ordinateur connus pour être mal conditionnés, il faut citer le problème de la reconstruction d'un modèle tridimensionnel à partir d'une séquence d'images prises par une caméra mobile. La difficulté de ce type de problème dépend de façon directe des contraintes imposées par l'application : type de scène, connaissance ou non du mouvement de la caméra, connaissance ou non des paramètres de calibration de la caméra. Ce problème a déja suscité de nombreux travaux de la part de la communauté scientifique, peu cependant abordant le problème dans sa totale complexité.

Cette thèse soutenue par l'Ifremer[1] se place dans une problématique très pragmatique et aux retombées applicatives directes : extraire des caractéristiques robustes et pouvoir reconstruire un modèle tridimensionnel de structures naturelles à partir d'une séquence d'images acquises au cours de mission d'exploration.

Dans notre cas, les images à traiter sont soumises à un certain nombre de contraintes importantes ; en effet, elles sont issues de séquences vidéos acquises avec une seule caméra, les scènes observées sont inconnues, les objets contenus dans celles-ci sont globalement verticaux, à texture aléatoire et de géométrie inconnue. De plus les paramètres intrinsèques et extrinsèques de la caméra sont également inconnus.

Le but de ce travail est donc de présenter à la fois une étude sur les méthodes permettant d'effectuer la reconstruction de ce type de structures, d'identifier les limites de ces méthodes et de proposer des approches originales visant à robustifier les différentes étapes menant à la reconstruction.

3 Contributions

Ce travail de thèse est surtout orienté vers les problèmes liés au transfert technologique vers des applications et des situations réelles. La principale contribution de cette thèse est d'avoir fourni un travail important sur la structuration de la méthodologie employée et d'intégration de méthodes robustes en vision par ordinateur dans le cas d'images réelles complexes. Plus précisemment, les principales contributions portent sur les points suivants :

– Apports conceptuels : nous avons étudié, comparé et classé différentes méthodes de traitement d'images sur les images de type Ifremer. A partir de cette classification, nous avons choisi et développé une méthodologie pour ce type d'images. A chaque étape de celle-ci, notre objectif a été de définir des approches robustes susceptibles de satisfaire les contraintes de l'application.

[1]Institut Francais de Recherche et d'Exploitation de la Mer

- Réalisation d'un outil logiciel : il s'agissait d'intégrer notre approche dans un outil logiciel puissant, évolutif et demeurant cependant facile à utiliser par un non spécialiste du traitement d'image et de la vision par ordinateur.
- Validation expérimentale : nous avons évalué le domaine de validité de notre méthodologie sur un ensemble représentatif d'images et montré sa supériorité en terme de robustesse vis à vis des méthodes de référence du domaine.

4 Contenu des chapitres

Outre cette introduction qui fait office à la fois de motivation et de présentation générale du problème, le manuscrit se compose de six chapitres organisés comme suit :

Chapitre 0 : Cette première partie replace cette thèse dans le contexte actuel de la recherche en vision par ordinateur. Nous décrirons brièvement les tenants et aboutissants de ce travail.

Chapitre 1 : Dans ce chapitre, nous présenterons les différentes techniques classiques pour extraire des informations dans des images en vue d'une reconstruction. Notre choix se portera sur un détecteur de points, celui de Harris et Stephens, que nous avons comparé avec d'autres détecteurs récents. Nous verrons qu'il est possible de l'optimiser au niveau de l'implémentation. Nous montrerons l'apport essentiel d'une approche multi-échelles en terme de stabilité et de robustesse au niveau de l'extraction et de la caractérisation des points d'intérêts. Ces deux notions de robustesse et de stabilité seront précisées sous l'éclairage particulier du type d'images que nous avons à traiter.

Chapitre 2 : Cette seconde partie mettra en avant plusieurs méthodes d'appariement de points entre images, étape indispensable et primordiale lors de la reconstruction de scènes. Nous détaillerons notre approche, et montrerons comment la structure multi-échelles que nous avons introduite au chapitre précédent permet d'améliorer la qualité et la robustesse de l'appariement, problème difficile dans le cas d'images complexes naturelles.

Chapitre 3 : Dans ce chapitre nous aborderons le problème de la reconstruction d'un modèle tridimensionnel. L'utilisation de caméras non calibrées et le fait que le mouvement de la caméra soit inconnu, nous amènera tout naturellement à parler de reconstruction projective. Nous décrirons plusieurs méthodes de reconstruction projective robuste qui ne requièrent pas des temps de calcul élevés. Puis, nous introduirons la méthode que nous avons retenue qui doit d'une part être rapide et d'autre part, être robuste et assurer une bonne estimation du modèle 3D calculé. Afin de compléter cette technique, nous discuterons de la représentation de la rigidité de la scène en tant que contrainte pour la reconstruction.

Chapitre 4 : En pratique, il n'est pas rare d'avoir des images sous-marines d'une qualité très médiocre (faibles gradients, images très bruitées) et notre approche montre alors ses limites. Nous montrerons que la robustesse introduite à chaque étape de nos traitements permettra, moyennant un traitement peu coûteux d'égalisation d'histogramme, d'obtenir toutefois des résultats très intéressants sur ce type d'images. Tout au long de ce travail de thèse, nous avons eu le souci constant de viser une implémentation aussi proche que possible du temps-réel de nos algorithmes. Nous présentons également des résultats sur les temps de calculs de toute la chaîne de traitement montrant que cette optique du temps-réel est envisageable.

Chapitre 5 : Enfin, nous présenterons l'outil logiciel que nous avons développé et dans lequel nous avons intégré nos algorithmes. Nous expliquerons l'utilité d'un tel logiciel et en quoi il est « innovant ». En effet, après une étude des logiciels déjà existants, nous avons pris le parti de développer un tel outil afin qu'il soit possible par la suite de l'étendre, de le manipuler simplement et surtout que nous ayons la possibilité de prototyper des applications.

Chapitre 1

Contexte et Positionnement du problème

La recherche en vision par ordinateur, bien que récente par rapport à d'autres sciences, s'est rapidement formalisée et développée au cours des deux dernières décennies et a même pour certains, déjà atteint ses limites. Cependant, dans la pratique, nombre de problèmes subsistent tant en termes de robustesse que de stabilité des algorithmes et donc rendent impossible l'exploitation directe de la théorie dans le cas d'images naturelles peu structurées.

1.1 La vision par ordinateur aujourd'hui

Qu'en est-il aujourd'hui de la recherche appliquée en vision par ordinateur ? Par cette question, nous souhaitons tout d'abord montrer qu'il existe encore bon nombre de problèmes ouverts donnant lieu à de nombreux travaux de recherche, puis repositionner ce travail de thèse par rapport aux résultats obtenus en vision par ordinateur ces dernières années. Nous montrerons que la problématique contenue dans l'application proposée par l'Ifremer dépasse largement le simple cadre applicatif de l'imagerie sous-marine et que sa résolution reste un point incontournable au développement d'autres thèmes de recherches comme par exemple, la navigation d'engins mobiles en environnement naturel inconnu par asservissement visuel.

S'il est incontestable que la vision par ordinateur a beaucoup mûri sur le plan théorique, avec entre autre le formalisme et l'exploitation de la géométrie projective et de toute la théorie qui en découle, le passage à la pratique n'en reste pas moins délicat. En effet, même si en théorie, on sait très bien exprimer la relation géométrique entre deux images d'une même scène prises avec des points de vue différents, il s'avère que le calcul des matrices fondamentales ou essentielles liant ces images est très difficile à réaliser dans un grand nombre de cas. Ainsi, reconstruire un modèle 3D d'une scène inconnue, même à partir d'une paire de caméras calibrées, reste délicat et requiert le plus souvent une intervention manuelle de la part d'un utilisateur qualifié. Dans certain cas, il peut même s'avérer impossible à faire ou pire, conduire à des résultats erronnés. A l'heure actuelle, dans le cas de scènes d'intérieurs, où généralement il y a des structures facilement identifiables comme des coins ou des contours, en utilisant de bonnes caméras calibrées, voire non calibrées, éventuellement avec des modèles plus ou moins approximatifs des dites scènes, dans des conditions d'éclairage normales, sans perturbations majeures, sans occultations, avec des temps de calcul pouvant être longs, il est tout à fait possible de faire de la reconstruction robuste euclidienne et d'obtenir des résultats de bonne qualité. A noter, toutefois, que la plupart des systèmes qui réalisent ces fonctionnalités, le font rarement de façon complètement automatique. Néanmoins, ces cas sont considérés comme des cas plutôt simples et maîtrisés. Malheureusement, nous savons aujourd'hui qu'il n'existe pas de méthode générique pour faire de la reconstruction robuste ou encore du suivi de caractéristiques quand on s'écarte de ces conditions quasi-idéales. L'explication en est simple : il s'agit de problèmes inverses mal conditionnés lorsqu'ils sont pris dans leur généralité. Une façon de les stabiliser sera d'introduire des contraintes supplémentaires comme par exemple dans le cas de l'utilisation d'un système stéréo calibré où les algorithmes sont nettement plus robustes et permettront ainsi de s'affranchir de beaucoup de situations délicates. Dans un grand nombre de situations, la mise en oeuvre d'un tel système se révèlera néanmoins impossible. Il faudra alors s'orienter vers des méthodes de régularisation dont on sait qu'elles sont très dépendantes de la validité des hypothèses sur lesquelles les fonctions de régularisation sont définies. Par exemple, le cas d'une caméra mobile se déplaçant dans la scène illustre parfaitement ces problèmes. Si le déplacement est faible entre deux images, on saura relativement bien apparier des caractéristiques mais l'évaluation de la matrice fondamentale sera de mauvaise qualité, du fait du mauvais conditionnement de l'étape de triangulation. On saura donc faire du suivi robuste de caractéristiques mais dans ce cas, on ne saura pas reconstruire la scène. On peut aussi donner comme exemple une scène où des points sont bien appariés, où le déplacement entre les images est suffisamment grand pour ne pas fausser le calcul de la matrice fondamentale, mais où hélas, les points sont quasiment sur un plan. Du point de vue théorique, il n'est pas possible de calculer cette matrice, pourtant dans la pratique, à cause des bruits contenus dans les images, une estimation, quoique complètement fausse, sera fournie par les algorithmes. La logique, dans ce cas, serait plutôt d'essayer de calculer une matrice d'homographie reposant sur un modèle différent.

Ces cas ne sont pas isolés, loin de là : ce sont des situations récurrentes qui posent problème. En résumé et en forçant un peu le trait, il y a presque autant de techniques que d'images. Certaines sont plus génériques que d'autres, certaines sont davantage orientées « temps-réel », d'autres sont très robustes sous certaines conditions (tête stéreo calibrée et modèle 3D de la scène), etc.

Mais il reste encore beaucoup à faire dans certains cas, qui pourtant peuvent paraître simples de prime abord : par exemple reconstruire des scènes et objets naturels avec une seule caméra ou encore estimer le mouvement entre deux images dans des conditions d'acquisition extrêmes.

Cete thèse se situe dans ce contexte où plusieurs ingrédients facilitant et stabilisant d'ordinaire la reconstruction, ne sont pas présents et où nous sommes contraints pratiquement à traiter le problème dans toute sa généralité. En effet, dans notre problématique, nous nous plaçons dans le contexte suivant :

1. les images sont acquises avec une seule caméra non calibrée et dont nous ne connaissons pas le mouvement,

2. les objets que nous devons reconstruire sont inconnus et non structurés,

3. le rapport signal/bruit dans l'image est en général très faible,

4. les caméras utilisées pour la prise de vue sont les caméras standards utilisées pour l'observation directe : résolution de l'image moyenne, validité du modèle projectif perturbée par le milieu de propagation, ...

1.2 Exemples d'images

Pour illustrer ces contraintes, nous présentons dans cette partie les images fournies par l'Ifremer qui serviront de fil conducteur tout au long de ce manuscrit. Celles-ci ont également servi comme base de référence pour les tests et expérimentations lors de cette thèse. Les échantillons ci-dessous sont tous issus de séquence vidéo sous-marines et ont leur caractère propre que nous allons détailler.

1.2.1 Rocher et fumerolle

Dans le cas de la figure 1.1 (de taille 256x224 pixels), nous sommes en présence d'une fumerolle sous-marine, dont la fumée perturbe les algorithmes classiques de segmentation au sens du mouvement. Le mouvement de la fumée devient le mouvement dominant dans l'image, il est alors difficile d'estimer celui de la caméra. Nous savons que seules des approches basées sur des champs de Markov permettent d'estimer un mouvement correct au prix d'un coût de calcul incompatible avec les objectifs temps réel que nous poursuivons. Dans cette série d'images, les problèmes sont assez simples à cerner : tout d'abord, la fumée s'échappe verticalement alors que le mouvement de la caméra est plutôt horizontal dans cette courte séquence (19 images), ce qui induit en erreur les techniques d'estimation de mouvement ; ensuite, le manque de structure géométrique rend très difficile l'extraction et l'appariement des contours de cette structure pourtant immobile et verticale ; enfin, la résolution moyenne des images et les bruits parasites compliquent énormément la tâche.

Fig. 1.1 – Image extraite de la séquence « Fumerolle »

Cette série d'images sera utilisée dans le cadre de cette thèse pour valider les performances de nos algorithmes d'extraction de caractéristiques robustes et de leur suivi. En revanche, du fait du peu de déplacement de la caméra le long de la séquence dont nous disposons, il ne nous sera pas possible d'aborder le problème de la reconstruction 3D.

1.2.2 Amphores

L'image de la figure 1.2 (de taille 256x256 pixels) est issue d'une longue séquence de plus de trois cents images, montrant un banc de sable sur lequel reposent des amphores. Cette série d'images se caractérise essentiellement par deux aspects : d'une part, les amphores occupent peu d'espace dans l'image, et d'autre part, la structure engendrée par l'ensemble des amphores est assez plane et ne contient finalement que peu d'informations 3D. Notons que cette séquence d'images a déja été exploitée au sein d'IFREMER pour faire du « mosaicing » en s'appuyant sur des techniques à base de résolution d'équations aux dérivées partielles, afin d'établir des cartes sous-marines sur de grandes distances.

Notons également que cette séquence est assez longue, ce qui est intéressant pour effectuer un suivi de points, que le mouvement de la caméra subit quelques soubresauts vraisemblablement dus aux courants et enfin que l'éclairage n'est ni constant ni uniforme.

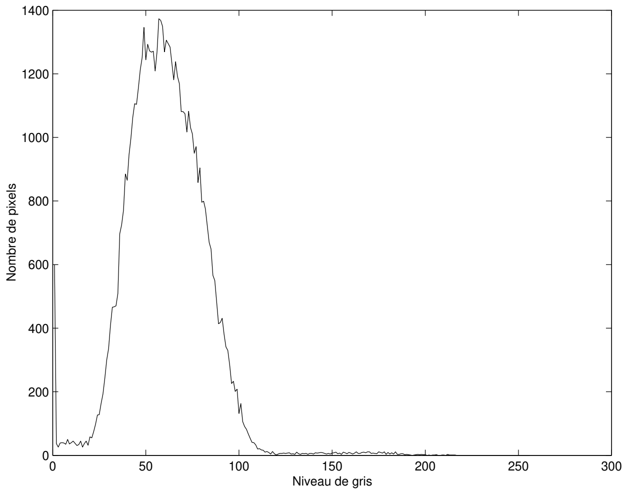
FIG. 1.2 – Image extraite de la séquence « Amphores »

Que pouvons-nous attendre du traitement de cette séquence ? En fait, un peu comme précédemment, nous montrerons que nous sommes en mesure d'extraire et de suivre des points robustes. Il sera difficile également de reconstruire un modèle 3D de cette scène du fait de son caractère planaire. En revanche, nous présenterons des résultats sur l'estimation robuste de matrices d'homographie et nous verrons que celle-ci est de très bonne qualité.

1.2.3 Titanic

Enfin, nous présenterons des résultats sur des images du Titanic. Les images de taille 256x256 pixels proviennent d'une séquence de 50 images. Comme on peut le voir sur la figure 1.3(a), l'image est vraiment dégradée et son histogramme sur la figure 1.3(b) en atteste. Evidemment, la principale difficulté de cette série vient du manque de dynamique des niveaux de gris. Par contre, on constate la présence de quelques structures rigides, comme par exemple, les rembardes du pont. Là aussi, les déplacements de la caméra dans cette séquence sont marqués par des soubresauts, créant un mouvement saccadé.

(a) Image originale



(b) Histogramme associé

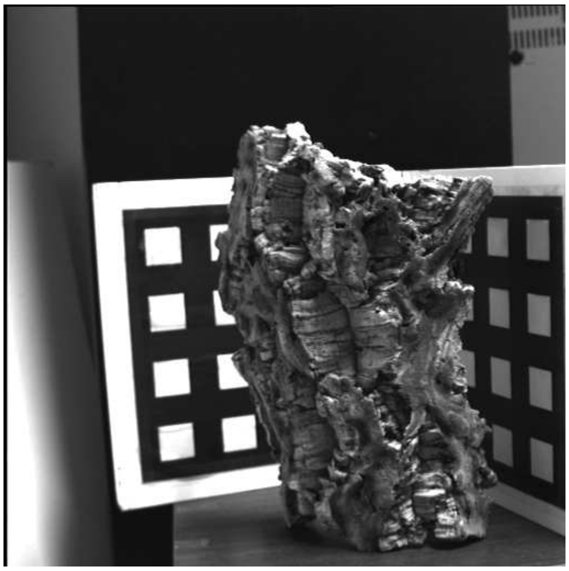
Fig. 1.3 – Image extraite de la séquence « Titanic »

Sur ce type d'images, quasiment toutes les méthodes robustes sont vouées à l'échec, à l'exception de certaines à base de résolutions d'équations aux dérivées partielles qui permettent d'apparier un certain nombre d'informations mais avec des temps de calculs de plusieurs dizaines de secondes pour un couple d'images. Nous proposons alors dans ces cas extrêmes d'images faibles dynamiquement un processus de pré-traitement en temps réel, qui nous permet ensuite d'extraire et de suivre des points robustes.

1.2.4 Validations expérimentales

Afin de valider nos algorithmes et approches, et surtout comme nous ne disposons pas de la vérité terrain sur les images naturelles, il nous faut utiliser des images acquises lors d'expérimentations où l'on connait les différents paramètres comme les paramètres intrinsèques de la, ou des caméras, les conditions d'éclairage, le type d'objet observé, etc.

Dans la mesure où l'application porte essentiellement sur des objets sous-marins et à terme sur des cheminées hydrothermales (cf. figure 1.1) ou des rochers, nous avons choisi de prendre des morceaux de liège ou de tronc d'arbre pour nos expérimentations en laboratoire. Ceux-ci ont en effet les caractéristiques essentielles des rochers sous marins : la structure est rigide et relativement verticale, la texture est, au sens du signal image, de même nature, c'est-à-dire répétitive et inconnue. Egalement, nous utiliserons soit une caméra calibrée ou non, soit une paire stéreo calibrée, pour valider la partie concernant la reconstruction de notre approche. La figure 1.4 présente un exemple d'une paire d'images stereo avec un morceau de liège. Ces mêmes images sans le liège, ne contenant que la mire, nous serviront pour effectuer les calibrations des caméras. Nous montrerons que notre approche est parfaitement validée avec des images d'objets naturels complexes acquises en laboratoire où nous connaissons et nous maîtrisons les conditions d'expérimentations.



(a) Liège : image gauche

(b) Liège : image droite

Fig. 1.4 – Exemple d'images d'un morceau de liège acquises en laboratoire par une tête stereo calibrée

1.3 Que peut-on attendre de cette thèse ?

Comme le montrent les précédents exemples, les images spécifiquement sous-marines traitées dans cette thèse ne sont pas aisées à traiter essentiellement à cause de leur relative mauvaise qualité intrinsèque (c'est-à-dire du point de vue du signal image). Mais à cela, il faut rajouter un certain nombre de facteurs qui ne simplifient pas la tâche, bien au contraire.

Tout d'abord, rappellons que nous sommes dans un cas mono-caméra non calibrée. Ensuite, les séquences fournies par l'Ifremer n'ont pas été acquises explicitement en vue d'une reconstruction ultérieure et donc il s'ensuit que le mouvement est totalement inconnu et le plus souvent inadapté à la reconstruction. En particulier les mouvements de la caméra sont le plus souvent des translations quasi-pures, sans la moindre rotation autour de l'objet observé, ce qui ne permet d'avoir suffisamment d'informations 3D. Enfin, aucune connaissance a priori n'est fournie sur la nature et les dimensions des objets dans la scène qui aurait pu permettre d'accéder à une reconstruction euclidienne de celle-ci.

Il y a donc plusieurs enjeux à ce travail de thèse. Tout d'abord, il s'agit de fournir une étude sur les différentes possibilités d'extraction d'informations robustes dans des images de scènes naturelles complexes. Ensuite, il faut assurer de façon robuste le suivi et l'appariement de ces points. Il faut alors fournir une méthodologie en vue d'une reconstruction partielle ou complète (si cela est possible), utilisant des méthodes robustes. Enfin, le dernier aspect de ce travail concerne son application : il faut que les logiciels développés soient fonctionnels et que les algorithmes puissent être utilisables dans des conditions réelles et surtout avec des temps de calcul très faibles, si possible en temps réel.

Il faut néammoins être réaliste sur les limites de ce travail : la reconstruction de scènes naturelles est déjà par nature un problème complexe non encore résolu de façon générique. Les résultats présentés doivent être analysés en gardant présent à l'esprit les conditions d'acquisition de la séquence, notamment la nature du mouvement de la caméra. A ces difficultés s'ajoutent celles plus spécifiques du milieu sous-marin dont on donne ici une liste probablement incomplète mais qui fournira au lecteur une idée suffisamment précise de celles-ci :

- les conditions d'éclairage sont très souvent délicates, essentiellement dues à la dispersion et à la diffusion dans l'eau des spots installés sur les robots sous-marins ;
- les mouvements des sous-marins ne sont pas fluides à cause des courants que l'on ne peut pas toujours compenser, ce qui a des répercussions sur les mouvements de la caméra ;
- la présence de poussières de sédiments due aux mouvements des hélices vient ajouter du bruit parasite dans les images ;
- enfin, il n'est pas rare d'avoir des occultations des objets dues à des poissons passant devant la caméra.

Néanmoins, les résultats que nous présenterons sur des images obtenues en laboratoire permettront de se faire une idée des performances de notre approche dans le cas de conditions d'acquisition mieux contrôlées. Les scènes que nous utiliserons seront certes différentes, mais contenant tout de même des objets qui, dans leur structure, conservent les hypothèses que nous nous sommes fixées dans le cadre de ce travail.

Chapitre 2

Extraction de caractéristiques

L'image en deux dimensions n'est qu'une représentation d'un monde en trois dimensions. S'il est naturel pour le cerveau humain de passer de cette information d'intensité lumineuse à une représentation sur laquelle on puisse raisonner, ce cheminement n'a rien d'évident pour une machine qui va, le plus souvent, devoir créer et manipuler des représentations intermédiaires. La première étape d'un processus d'analyse d'images va consister à structurer l'information contenue dans les pixels de l'image afin d'éliminer d'une part, l'information non utile à la tâche de vision et, d'autre part, d'extraire et de représenter l'information nécessaire à la poursuite du processus d'analyse. Cette information utile dépend, bien sûr, de la finalité de la tâche de vision.

2.1 Introduction

En vision par ordinateur, le premier et difficile problème est celui de l'extraction d'informations caractéristiques contenues dans une ou plusieurs images. En effet, la représentation que notre oeil se fait d'une droite ou d'un coin n'est pas nécessairement facile à interpréter du point de vue du signal image. Depuis de nombreuses années maintenant, les chercheurs se penchent sur cet aspect de traitement « bas-niveau ». Si les progrès sont sans conteste indéniables, il n'en reste pas moins que certains cas posent encore des problèmes tant théoriques que pratiques. Nous évaluerons différentes méthodes d'extraction de caractéristiques sur les images sous-marines que nous a fournies l'Ifremer.

Dans ce chapitre, nous présentons un état de l'art, nécessairement incomplet tant la littérature est fournie, mais nous essaierons de nous restreindre à des méthodes récentes et robustes orientées vers le traitement d'images naturelles complexes. Nous renvoyons le lecteur aux références proposées dans ce chapitre pour des compléments de bibliographie.

2.2 Etat de l'art

Parmi les éléments caractéristiques que l'on cherche à extraire, on a longtemps cherché des méthodes robustes pour extraire des contours ou des lignes, puis des coins et des points d'intérêt. D'une part, ce sont des formes faciles à interpréter pour l'oeil humain et d'autre part, elles représentent une description physique réelle de la scène contenue dans l'image.

Historiquement, les chercheurs ont commencé à s'intéresser sérieusement à la vision par ordinateur à la fin des années 70. En effet, les ordinateurs et les « nouvelles » technologies de l'époque permettaient de s'atteler à des domaines de recherches encore peu ou pas répandus. D. Marr a exposé au tout début des années 80 le paradigme qui depuis porte son nom. Le principe est assez simple et pour lui, le traitement d'une ou plusieurs images se découpe en trois étapes : segmentation, reconstruction et reconnaissance. Sans détailler ici les recherches allant jusqu'aux début des années 80, citons un ouvrage de référence retraçant les cheminements des recherches et l'état de l'art de l'époque, le livre de D.H. Ballard et C.M. Brown (Ballard et Brown, 1982). On y trouve une partie intéressante et fournie sur les techniques dites "bas-niveau" lors des débuts de la vision par ordinateur.

Plus récemment, dans les premiers chapitres du livre de R. Horaud et O. Monga (Horaud et Monga, 1995), le lecteur aura un inventaire précis des méthodes d'extraction et de segmentation de contours. Les auteurs y décrivent les nombreuses méthodes des années 90 et mettent l'accent sur les problèmes qui y sont liés, comme le calcul des dérivées de l'image ou encore le chaînage des contours, par exemple.

2.2.1 Contours et droites

Les contours et par déclinaison les droites, ont été les premières informations que l'on a cherché à modéliser et extraire des images. L'extraction de contours rigides a été beaucoup étudiée dans les années 80. Les méthodes à base de convolution et de calcul de masques de dérivation ont été très répandues, car rapides et faciles à implémenter : on peut citer par exemple les masques et opérateurs de Sobel, Roberts, Kirsch ou Prewitt. Néanmoins, ces approches, basées sur une approximation du gradient spatial par des différences finies, ne sont ni précises ni robustes aux bruits et ne fonctionnent à peu près bien que dans des cas très simples.

-1	0	1
-2	0	2
-1	0	1

1	2	1
0	0	0
-1	-2	-1

Opérateur de Sobel : calcul des gradients en X et en Y.

Par la suite, les recherches se sont portées sur les techniques d'estimation de dérivées premières de l'image en utilisant des bases mathématiques plus solides pour formaliser la notion de dérivée. Ce problème, bien que largement traité, demeure encore difficile à résoudre dans des images naturelles, l'estimation de la dérivée étant très sensible aux bruits. Là encore, on trouve un large panel de détecteurs de contours dont le plus connu et répandu est certainement celui de J. Canny (Canny, 1986) et amélioré par R. Deriche (Deriche, 1987). O. Faugeras présente dans le chapitre 4 de son livre (Faugeras, 1993) une partie intéressante et complète sur différentes approches pour extraire des contours. La particularité et l'intérêt majeur du détecteur de Canny-Deriche, outre sa bonne qualité, réside dans le fait qu'une implémentation optimale est possible. Rappelons brièvement le principe de ce détecteur. Celui-ci doit impérativement répondre à trois critères :

- détection : il doit y avoir une réponse de l'opérateur dans le voisinage du contour
- localisation : le contour doit être localisé avec une grande précision
- unicité de la réponse : le contour ne provoque qu'une seule réponse à l'opérateur

Habituellement, on effectue une convolution du contour (bruité) avec une fonction antisymétrique. Ainsi, J. Canny définit une formulation de ces trois critères qui mène à une équation différentielle dont la solution est le filtre f permettant alors la détection du contour. La position du contour est alors donnée par : $max(I * f)(x)$. J. Canny propose alors une solution, mais qui n'est pas optimale. R. Deriche étend les précédents travaux et propose un filtre dont la dérivée est la solution exacte à l'équation de Canny étendue à des filtres infinis.

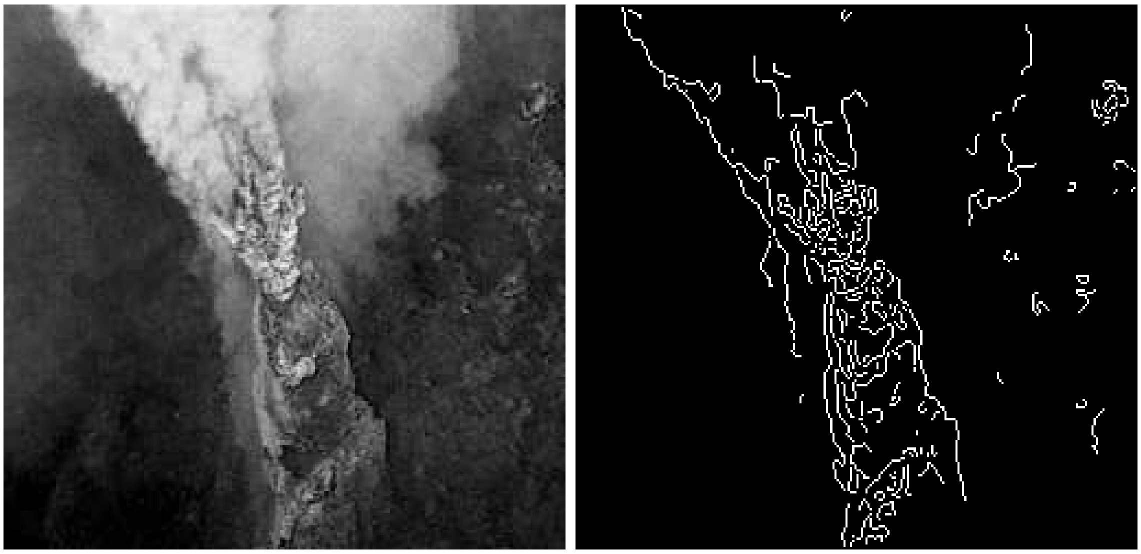
A la fin des années 80, M. Kass *et al.* (Kass et al., 1988) ont étendu ces travaux à l'extraction de contours déformables aussi appelés « snakes ». Le principe est le suivant : le *snake* peut être vu comme l'énergie minimale d'une spline, contrainte par des forces internes et externes, pour "coller" au mieux des lignes et contours dans l'image. De façon plus formelle, la position d'un *snake* de façon paramétrique étant $\mathbf{v}(s) = (x(s), y(s))$, alors sa fonction d'énergie s'écrit :

$$\begin{aligned} E^*_{snake} &= \int_0^1 E_{snake}(\mathbf{v}(s))ds \\ &= \int_0^1 \left(E_{int}(\mathbf{v}(s)) + E_{image}(\mathbf{v}(s)) + E_{con}(\mathbf{v}(s))\right) ds \end{aligned}$$

où E_{int} représente l'énergie interne du contour, qui le contraint à être régulier et lisse, E_{image} « attire » le contour vers la position recherchée et E_{con} est le terme de contraintes sur l'espace des solutions. Ces travaux ont été largement repris et developpés, par exemple dans le domaine de l'imagerie médicale. Ces approches ont amené de nouvelles problématiques de recherche, communément appelées « contours actifs », dont nous parlerons dans la section suivante.

Nous avons donc testé un détecteur de Canny-Deriche sur les images du Titanic, de la fumerolle, des amphores et du liège. Les résultats sont présentés sur les figures 2.1(b), 2.2(b), 2.3(b) et 2.4(b). Comme on peut le constater, les contours extraits correspondant aux objets d'intérêt dans l'image sont difficilement identifiables et de plus, peu stables d'une image à l'autre dans la séquence. On note même que, dans le cas de la fumerolle, les véritables contours du rocher sont mal extraits, alors que visuellement, l'oeil arrive à les distinguer relativement clairement. C'est également le cas pour l'image du Titanic où le contour de la rambarde n'est pas extrait en totalité alors que là encore, l'oeil l'identifie clairement. En fait on peut obtenir la totalité de cette rambarde en modifiant le seuillage, mais on va rajouter beaucoup plus d'informations dans l'image, compliquant alors la phase d'appariement. De plus, si l'on se place dans une optique d'un traitement « tout-automatique », force est de constater que, d'une part le réglage des paramètres n'est ni facile ni générique pour les exemples présentés ici et d'autre part, les contours extraits sont très sensibles aux dits réglages.

Il est assez évident que chercher des contours ou segments de droites n'a un sens que si les scènes observées contiennent des objets structurés, avec certaines « bonnes » propriétés géométriques. Or dans le cas de scènes naturelles, cet aspect géométrique est « rarement » présent. Dans la mesure où nous souhaitons à terme faire de la reconstruction, nous allons être amenés à effectuer des appariements entre images naturelles et il est clair que les contours ne sont pas les meilleures informations que l'on peut extraire dans notre cas. Notons aussi que le seuillage pour la détection des contours n'est pas si simple et si, sur certaines images, on veut à tout prix conserver tel ou tel contour, cela ne peut se faire qu'au prix d'un ajout

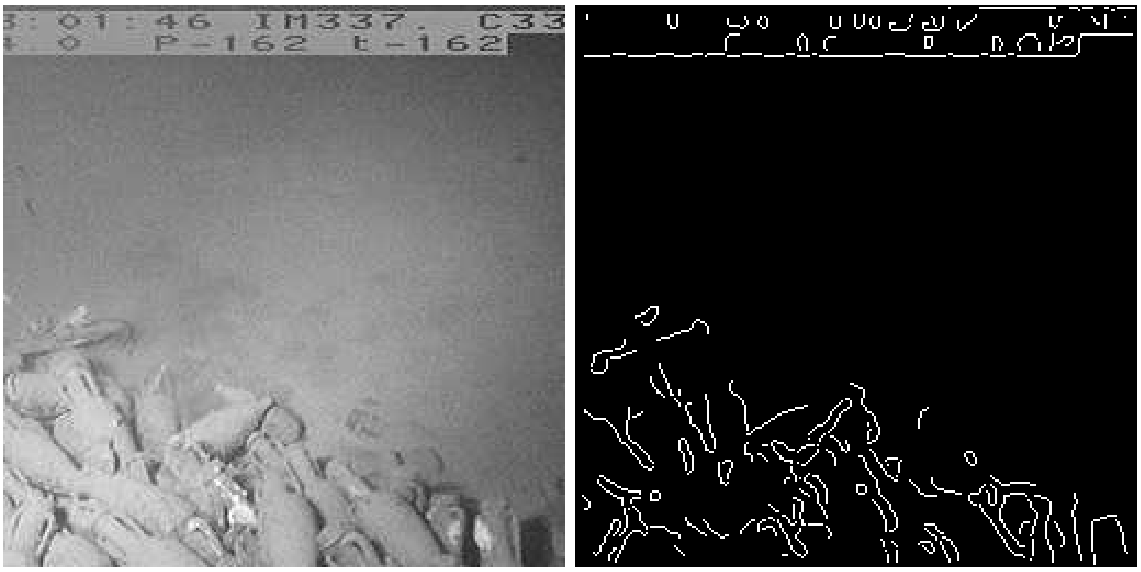


(a) Fumerolle (b) Contours extraits

FIG. 2.1 – Extraction de contours sur une image de la séquence « Fumerolle » avec l'algorithme de Canny-Deriche

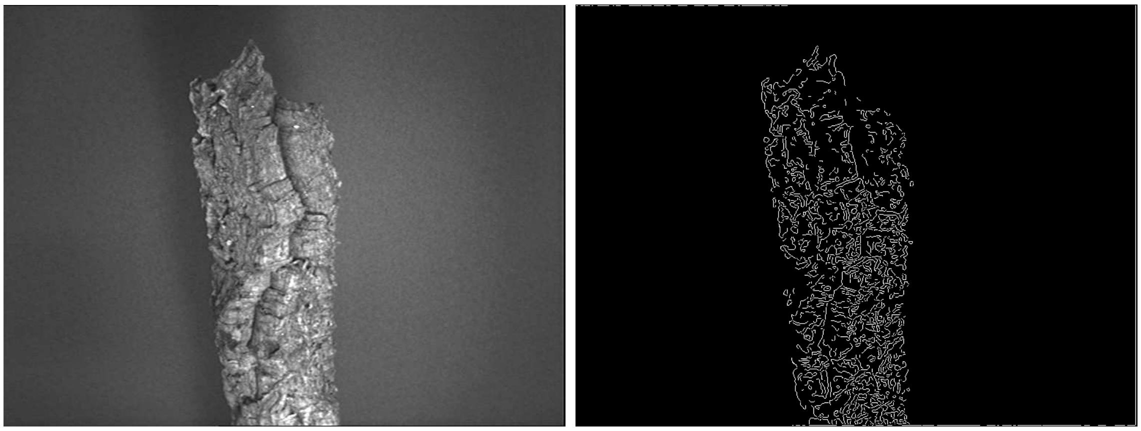
(a) Titanic (b) Contours extraits

FIG. 2.2 – Extraction de contours sur une image de la séquence « Titanic » avec l'algorithme de Canny-Deriche



(a) Amphore (b) Contours extraits

Fig. 2.3 – Extraction de contours sur une image de la séquence « Amphores » avec l'algorithme de Canny-Deriche



(a) Liège (b) Contours extraits

Fig. 2.4 – Extraction de contours sur une image de la séquence « Liège » avec l'algorithme de Canny-Deriche

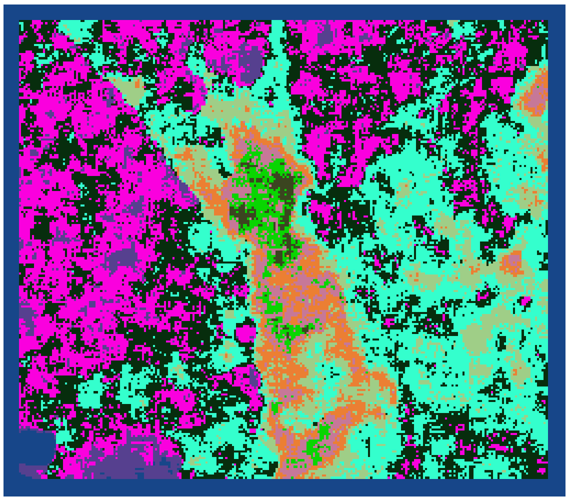
ou d'une perte d'un certain nombre d'autres contours. De plus, l'étape de chaînage peut également devenir réellement problématique, comme dans le cas du liège et l'intervention d'un opérateur spécialisé peut s'avérer indispensable.

2.2.2 Segmentation dans une image

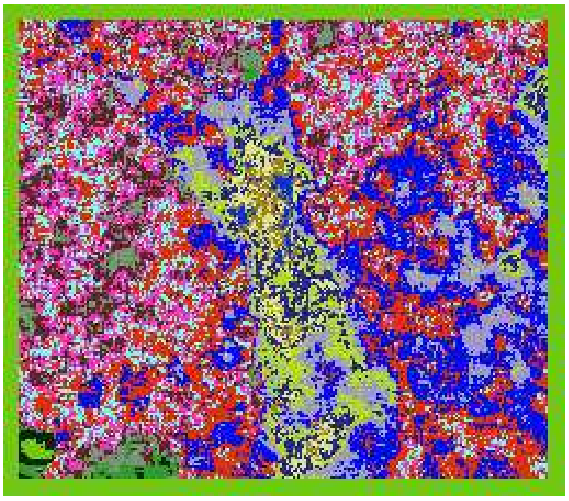
Dans la littérature, on trouve également des approches basées sur le partitionnement en régions, comme par exemple la segmentation des différentes textures. Le lecteur trouvera dans les chapitres 7 et 8 de (Cocquerez et Philipp, 1995), une bonne introduction sur les problèmes de segmentation et de filtrage. Ces méthodes sont très utilisées en imagerie satellitaire pour différencier les zones urbaines des zones péri-urbaines. Des techniques basées sur l'analyse de texture par des méthodes Markoviennes (on essaie alors de modéliser les textures via des champs de Markov) ont été mises en oeuvre avec un certain succès. Le lecteur pourra trouver plus d'informations dans l'état de l'art de la thèse d'A. Lorette (Lorette, 1999). Egalement, les travaux de thèse de C. Samson (Samson, 2000) portent sur le traitement d'images satellitaires par approche variationnelle en utilisant des équations aux dérivées partielles. Dans ce type d'images, il est nécessaire de se baser sur des méthodes d'apprentissage pour faciliter l'étape de classification des différentes textures (segmentation supervisée). Ces classifications peuvent se faire aussi avec les réseaux Bayésiens comme le font V.P. Kumar et U.B. Desai dans (Kumar et Desai, 1996). Des exemples de segmentation de textures avec les méthodes Markoviennes sont présentés sur les figures 2.5(b), 2.5(c), 2.6(b) et 2.6(c). Nous avons sélectionné « a priori » un nombre de classes de textures et essayé de classifier les pixels en fonction de ces classes. Ainsi il est possible de regrouper les pixels appartenant à une même classe et d'en déterminer les contours dans l'image. Mais, on le constate, ces résultats sont très difficiles à interpréter, et finalement, les contours d'occultations que l'on cherchait à mettre en évidence sont difficilement interprétables. Accessoirement, le réglage des paramètres reste à la fois délicat et non générique. Les temps de calcul sont en général longs, surtout si l'on utilise des méthodes de recuit simulé, habituellement utilisées pour l'optimisation.

Plus récemment, on s'est interessé à des techniques de restauration d'images dégradées en utilisant des méthodes faisant appel à la théorie des EDP (équations aux dérivées partielles) dont on a adapté le formalisme pour qu'il soit plus approprié au monde de la vision. R. Deriche et O. Faugeras en font une description poussée dans (Deriche et Faugeras, 1995). Si les résultats sont encourageants et ces techniques promises à un grand avenir, il nous est difficile dans notre cas de les utiliser car nos images ne s'y prêtent pas et les temps de calcul sont trop longs. On le constate aisément sur les figures 2.5(a) et 2.6(a) pour des images issues de la séquence « Fumerolle » et celle du « Liège ». Dans le premier exemple, dans la mesure

(a) Fumerolle

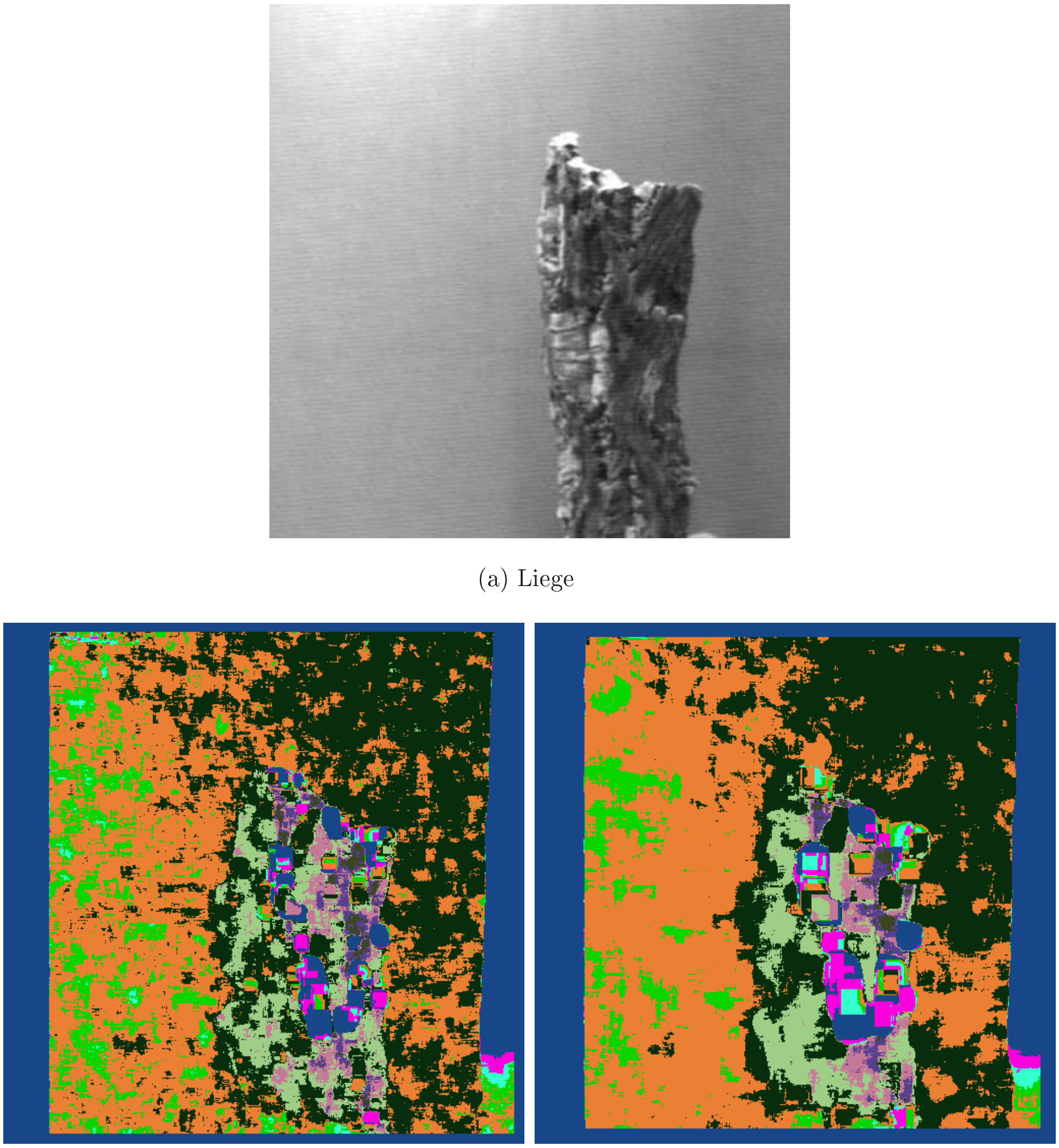


(b) 10 classes de textures



(c) 15 classes de textures

FIG. 2.5 – Segmentation de textures de la Fumerolle en 10 et 15 classes de textures à l'aide d'approches Markoviennes



(a) Liege

(b) Liege : 10 classes

(c) Liege : 25 classes

FIG. 2.6 – Segmentation de textures du Liège en 10 et 15 classes de textures à l'aide d'approches Markoviennes

où il est clairement difficile de délimiter à l'oeil nu les zones contenant la fumée et le sol, les algorithmes de segmentation sont mis en échec et les contours des différentes zones d'intérêt ne sont ni précis ni stables par rapport à une variation des paramètres, comme on l'observe sur les figures 2.5(b) et 2.5(c). Dans le second exemple, on s'attend d'une part à ce que la segmentation délimite correctement les contours car le fond est uniforme et d'autre part, à ce que peut de classes suffisent pour caractériser le morceau de liège. En réalité, les résultats obtenus sont assez moyens comme le montrent les figures 2.6(b) et 2.6(c). Il est clair que les objets que nous traitons dans cette thèse ne sont pas adaptés à ce type d'approche. Notons également qu'il a fallu une dizaine de secondes par image sur un PC à 200Mhz pour effectuer ces segmentations.

R. Deriche et N. Paragios ont utilisé les contours actifs géodésiques pour faire de la segmentation de textures (Paragios et Deriche, 1998). Ces travaux sont une extension de ceux de M. Kass *et al.* (Kass et al., 1988). Le lecteur trouvera aussi une bonne synthèse sur les travaux de contours actifs dans le livre d'A. Blake et M. Isard (Blake et Isard, 1998). N. Paragios a étendu le concept à des régions actives géodésiques (Paragios, 2000). Cette technique est une approche variationnelle, implémentée par des « level sets ». Le principe étant de minimiser des énergies, composées d'une part des termes d'attache aux données et d'autre part d'un terme de régularisation. Cependant, de nombreux problèmes subsistent avec ce type d'approche. Tout d'abord, il faut être capable de calculer de bonnes valeurs d'attaches aux données et de bien choisir le terme de régularisation, éventuellement en utilisant des a priori markoviens. Ensuite, dans la mesure où ce sont des méthodes itératives, il est nécessaire de bien initialiser les courbes qui vont évoluer. N. Paragios impose par exemple le fait qu'il doit y avoir au moins une partie de l'objet à segmenter dans une des courbes initiales. Enfin, rien n'assure de converger vers le minimum global et bien souvent, la solution trouvée correspond à un minimum local : dans certains cas, le résultat peut convenir mais très souvent, celui-ci est loin de la solution désirée.

Ces méthodes de plus en plus répandues sont essentiellement robustes (à la convergence des algorithmes, on a un résultat stable) et donnent de bons résultats dans les cas où les différentes textures sont clairement différentiables les unes des autres (par exemple, une texture rayée sur une texture assez uniforme). Donc dans notre cas, ces méthodes sont vouées à l'échec d'une part sur la qualité de la segmentation et d'autre part sur les temps de calcul trop élevés.

2.2.3 Points d'intérêt et coins

Nous nous intéressons maintenant aux coins et aux points d'intérêt. Dès 1977, H. Moravec (Moravec, 1977) introduit la notion de points d'intérêt. Pour lui, certains points dans une image peuvent avoir des caractéristiques plus significatives que les autres et ont donc un intérêt plus important. Puis, P.R. Beaudet (Beaudet, 1978) cherche à formaliser les coins dans une image et est le premier à proposer un détecteur.

Depuis, de nombreux travaux ont été effectués que l'on peut classer en deux grandes catégories :

(a) les détecteurs utilisant les contours et leurs fortes courbures

(b) les détecteurs utilisant une représentation directe des coins.

Les premières techniques ont été très étudiées depuis une vingtaine d'années (Asada et Brady, 1986), (Deriche et Faugeras, 1990), (Mokhtarian et Suomela, 1998). L'idée est d'extraire les contours d'une image et de chercher sur ces derniers, les points de forte courbure. Les deux principaux problèmes d'une telle approche viennent d'une part de l'extraction des contours, très sensible aux bruits et d'autre part, du chaînage des contours qui peut s'avérer délicat dans le cas des occultations, par exemple.

Dans les secondes approches, on ne passe pas par une phase explicite d'extraction de contours, mais on essaie de caractériser directement les pixels comme étant des coins. Pour cela, deux sortes de travaux se retrouvent dans la littérature : soit on cherche une représentation de coins via un modèle paramétrique (Rohr, 1992), (Blaszka et Deriche, 1994), soit on utilise une mesure calculée directement sur un ensemble de pixels du voisinage du point considéré (Dreschler et Nagel, 1982), (Kitchen et Rosenfeld, 1982), (Noble, 1988), (Harris et Stephens, 1988) et plus récemment (Smith et Brady, 1997).

2.2.3.1 Utilisation d'un modèle de coins

Utiliser un modèle paramétrique de coins revient en fait à chercher des jonctions bien déterminées comme des « T » ou encore des « L » comme le fait K. Rohr (Rohr, 1992), en utilisant un modèle paramétrique permettant de différencier des classes caractéristiques du signal, dont les coins. Il effectue alors une minimisation robuste pour ajuster au mieux les paramètres de son modèle et cela lui permet d'obtenir des localisations très précises des coins, puisque subpixelliques. Néanmoins, il est clair que l'utilisation d'un tel modèle sous-entend fortement qu'il doit y avoir la présence de structures géométriques dans l'image, ce qui n'est pas le cas dans les images sous-marines. T. Blaszka and R. Deriche (Blaszka et Deriche, 1994) ont étendu ces travaux, essentiellement pour la partie concernant l'initialisation des modèles paramétriques.

2.2.3.2 Utilisation de mesures calculées directement à partir du signal luminance

Plusieurs travaux de recherche sont issus des travaux initiaux de P.R. Beaudet (Beaudet, 1978) qui cherchait des points d'intérêts en calculant la mesure suivante appelée DET, invariante aux rotations dans l'image :

$$DET = I_{x^2} I_{y^2} - I_{xy}^2 \tag{2.1}$$

où I est le signal image et I_i ses dérivées. Les maxima locaux de la fonction DET correspondent alors aux points d'intérêt. Notons que DET est lié à la courbure gaussienne de la surface définie par le signal image. Cependant, on rencontre deux problèmes majeurs avec cette mesure : d'une part, cela nécessite le calcul des dérivées secondes de l'image et d'autre part, il a été montré (Deriche et Giraudon, 1993) que les points d'intérêt ne sont pas bien localisés puisqu'ils sont situés à l'intérieur des coins cherchés. Les chercheurs ont donc travaillé sur la bonne localisation de ces points d'intérêt sous l'hypothèse de la présence de coins. Par exemple, L. Dreschler et H. Nagel (Dreschler et Nagel, 1982) proposent un détecteur utilisant directement la courbure gaussienne.

L. Kitchen et A. Rosenfeld (Kitchen et Rosenfeld, 1982) proposent quant à eux la mesure suivante :

$$K = \frac{I_{x^2} I_y^2 + I_{y^2} I_x^2 - 2 I_{xy} I_x I_y}{I_x^2 + I_y^2} \tag{2.2}$$

Il a été montré par H.H. Nagel (Nagel, 1983) que ces différents détecteurs et celui de O.A. Xuniga et R.M. Haralick (Zuniga et Haralick, 1983) sont équivalents. Une étude précise et complète sur plusieurs détecteurs et leur comportement a été effectuée par R. Deriche et G. Giraudon dans (Deriche et Giraudon, 1991).

Se basant alors sur les travaux de H. Moravec (Moravec, 1977) qui a été le premier à introduire la fonction d'auto-corrélation, Förstner (Förstner et Gülch, 1987), puis, C. Harris et M. Stephens (Harris et Stephens, 1988) ont proposé un détecteur de coins, que nous détaillons dans le paragraphe 2.3.1.3. On trouvera dans (Schmid et al., 1998) une comparaison de plusieurs détecteurs de coins dont ceux de Harris et Förstner.

2.3 Extraction de points dans des images naturelles

Comme décrit dans la partie précédente, la détection de coins et de points d'intérêt est un domaine de recherche actif. Encore récemment, des travaux font preuve d'originalité, comme ceux par exemple de :

- R. Laganière qui utilise un ensemble d'opérateurs morphologiques pour déterminer la position réelle du coin, même sur des contours à 45 degrés. Pour cela, il utilise des masques de pixels prédéfinis, chacun ayant une direction privilégiée, ce qui lui permet de détecter des coins autres que ceux en « T, L » ou « Y ». Les résultats obtenus sont robustes aux bruits et de bonne facture (Laganière, 1998).
- F. Chabat qui propose quant à lui un détecteur qui estime la vraie position du coin mais aussi son orientation. Sa particularité est de permettre également la détection des jonctions. Cela sous-entend la présence de structures dans les images considérées. La technique employée pour estimer l'orientation se base sur une analyse des pixels appartenant au contour sur lequel le coin se situe. Les résultats sont également de bonne qualité et robustes aux bruits (Chabat et al., 1999)

2.3.1 Comparaison de détecteurs de points

Dans notre problématique de reconstruction de scènes naturelles sous-marines inconnues, il est évident que rechercher des structures géométriques telles que des lignes ou des droites ne semble pas approprié. Les points d'intérêt, tout en ayant une « structure » géométrique minimale, permettent d'appréhender toutefois des propriétés physiques telles que la rigidité ou le maillage d'objet. En celà, ils sont particulièrement intéressants et adaptés aux objets naturels non structurés. Le problème majeur est que la notion de point d'intérêt est difficile à définir de manière formelle. Les chercheurs se sont surtout concentrés sur les représentations des coins et leurs structures géométriques. De nombreux détecteurs ont été étudiés durant les dernières années et nous avons testé et comparé trois détecteurs de coins considérés comme robustes : Susan, Css et Harris. Nous rappelons dans les paragraphes suivants brièvement leur fonctionnement et présentons des résultats d'extraction de coins dans le cas d'images naturelles.

2.3.1.1 Susan

S.M. Smith et J.M. Brady (Smith et Brady, 1997) proposent un détecteur de coins et de contours, robuste aux bruits, qui ne nécessite pas le calcul des dérivées de l'image et aucun traitement pour réduire le bruit contenu dans celle-ci. Cette approche originale est basée sur l'idée de considérer des masques de pixels de forme circulaire dont les centres sont appelés des "nucleus". On compare alors le niveau de gris de chaque pixel contenu dans le masque avec celui du "nucleus", ce qui définit une zone appelée "USAN" ("Univalue Segment Assimilating Nucleus"). On associe alors à chaque point de l'image une zone locale de même niveau de gris. A partir du barycentre, de la taille et des moments de second ordre de l' "USAN", on évalue à la fois les contours et les coins : plus cette zone est petite, plus la présence d'un coin est probable.

2.3.1.2 Css

Pour leur part, F. Mokhtarian et R. Suomela (Mokhtarian et Suomela, 1998) se basent sur un processus plus classique dans lequel la détection de coins nécessite une étape d'extraction de contours. Pour ce faire, ils utilisent le détecteur de Canny qu'ils ont optimisé pour les contours à 45 degrés et à 135 degrés, connus pour poser des problèmes. Leur approche utilise une représentation multi-échelle des contours préalablement extraits. Les coins étant définis comme les maxima de courbure des contours, ils construisent donc l'espace multi-échelles des courbures. Les coins sont alors détectés au niveau le plus élevé de l'espace échelle et suivis dans les différentes échelles jusqu'à l'échelle initiale. Cela assure d'une part une bonne localisation des coins et d'autre part une bonne robustesse vis-à-vis du bruit. Le principal problème réside dans le fait qu'il faut tout de même un minimum de structures pour trouver des coins. En revanche, leur détecteur est très rapide.

2.3.1.3 Harris

Le détecteur de C. Harris et M. Stephens (Harris et Stephens, 1988), aussi appelé détecteur de Plessey, est certainement le plus connu et utilisé ; il est considéré comme robuste et fiable et a un coté universel, dans le sens où il fonctionne sur un large spectre d'images. Celui-ci est basé sur les travaux de H. Moravec (Moravec, 1977) qui a eu l'idée d'utiliser la fonction d'auto-corrélation afin de déterminer la meilleure position que doit avoir une fenêtre, de façon à ce que toute position voisine contienne moins d'informations. Cela signifie que si l'on se déplace par rapport au centre de la fenêtre, il doit être aisé de distinguer la position courante de la position précédente. Il y a donc trois cas qui se présentent :

(a) si la fenêtre se trouve dans une zone homogène, l'auto-corrélation donnera une réponse faible dans toutes les directions

(b) si la réponse est forte dans une direction prédominante, cela signifie que la localisation ne peut être distinguée dans les autres directions, on se trouve alors sur une arête

(c) si la réponse est forte dans toutes les directions, on se trouve dans une zone que l'on peut caractériser : motifs texturés, coins et « tâches »[1].

Dans son approche, H. Moravec calcule cette fonction d'auto-corrélation en utilisant des différences entre des fenêtres carrées définies dans le voisinage du point considéré. Le problème est que la réponse du détecteur est anisotrope. W. Förstner et E. Gülch (Förstner et Gülch, 1987) ainsi que C. Harris et M. Stephens ont repris les travaux de Moravec et ont montré que le calcul de la fonction d'auto-corrélation se ramenait à l'étude de valeurs

[1]que l'on trouve aussi dans la littérature sous le nom de *blobs*

propres de la matrice liée à cette fonction. Notons que J.A. Noble (Noble, 1988) a montré par ailleurs que le détecteur de Harris n'est optimal que pour les coins en forme de « L ». Le détecteur de Harris utilise donc comme matrice d'auto-corrélation la matrice M suivante :

$$\mathbf{M} = exp^{-\frac{x^2+y^2}{2\sigma^2}} \otimes \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \tag{2.3}$$

où $I_x = \frac{\partial I}{\partial x}$ et $I_y = \frac{\partial I}{\partial y}$. Le lecteur trouvera dans le chapitre 3 de la thèse de Y. Dufournaud (Dufournaud, 2001) une étude détaillée sur la fonction d'auto-corrélation et ses liens avec la matrice $\mathbf{M}$. Egalement, on trouvera dans (Allezard et al., 1999) une étude sur le détecteur de Harris en vue d'effectuer l'appariement de points lors de changements d'échelles.

Sachant que les valeurs propres de cette matrice $\mathbf{M}$ représentent les courbures principales de la fonction d'auto-corrélation, nous avons alors trois cas :

(a) si les deux valeurs propres sont faibles, la région considérée est homogène

(b) si une seule des valeurs propres est clairement dominante, on se trouve sur une arête

(c) enfin, si les deux valeurs propres sont élevées, il n'y pas de direction à privilégier, on se trouve en présence d'un point d'intérêt.

Le problème est donc l'évaluation de ces valeurs propres. Pour éviter un calcul explicite de celles-ci, C. Harris et M. Stephens proposent de calculer une mesure s'appuyant sur le déterminant et la trace de la matrice M. Notons que ces valeurs sont invariantes aux transformations isométriques et par conséquent, l'invariance de l'image aux rotations est préservée.

On évalue alors la mesure suivante :

$$K = det(\mathbf{M}) - \lambda Tr^2(\mathbf{M})$$

Si $K > 0$ alors il y a un point d'intérêt, λ étant généralement compris entre 0.04 et 0.06. On peut signaler que la puissance des ordinateurs récents n'empêche plus de calculer les valeurs propres avec des temps de calcul raisonnables. Néanmoins, d'une part dans le souci d'avoir des algorithmes aussi proches que possible du temps réel et d'autre part parce que nous réutilisons cette mesure de Harris plus tard, nous avons pris le parti d'en effectuer le calcul comme le font C. Harris et M. Stephens.

2.3.2 Implémentation robuste du détecteur de Harris

Il est bien connu que le calcul des dérivées n'est pas très bien conditionné, car il n'est pas robuste au bruit contenu dans le signal image. Pour s'en convaincre, il suffit de formaliser (en une dimension) le signal d'entrée comme suit :

$$f_1(x) = f(x) \pm \varepsilon sin(\omega x)$$

Alors f_1 et f sont équivalentes si ε est petit. En revanche si ε est grand, les deux fonctions ne sont plus équivalentes et il est clair que les dérivées de f et f_1 vont être différentes. Comme il faut prendre en compte ce bruit contenu dans les images dans le calcul des dérivés pour le détecteur de Harris, nous lissons le signal avant de calculer effectivement sa dérivée : $g \otimes f_1$. Le calcul de la dérivée s'effectue alors de la façon suivante :

$$\partial_i(g \otimes f_1) = g \otimes \partial_i f_1 = \partial_i g \otimes f_1$$

Le fait de lisser la fonction g et non pas directement la fonction f_1, robustifie le détecteur de Harris, comme l'a montré C. Schmid dans (Schmid et al., 1998). Pour l'implémentation de cet algorithme, nous utiliserons comme fonction g la fonction gaussienne :

$$G(x, \sigma) = \frac{1}{2\pi\sigma^2} exp^{-\frac{x^2}{2\sigma^2}} \tag{2.4}$$

Nous avons aussi opté pour l'implémentation récursive des gaussiennes telle que décrite dans (Deriche, 1987), car le temps de calcul, indépendant de la taille du σ, est très faible.

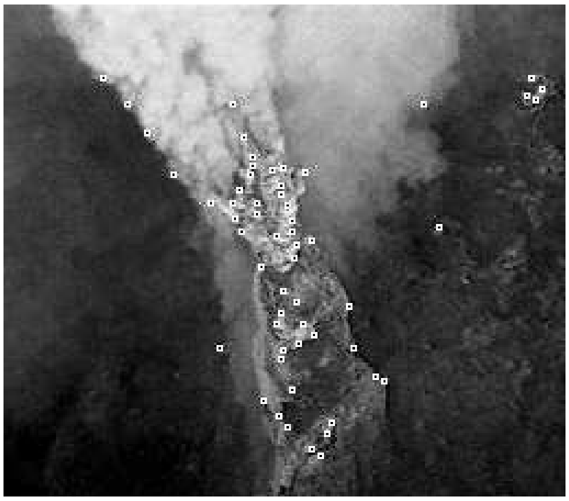
2.3.3 Comparaison des trois détecteurs

Dans la mesure où les images que nous devons traiter ne sont pas structurées, au sens où il n'existe pas de structures géométriques simples comme des droites, des cylindres, des coins, les critères de choix entre les différents détecteurs ne sont pas simples. Nous nous sommes basé sur la méthode décrite dans (Schmid et al., 1998) pour effectuer la comparaison. Celle-ci repose sur la notion de répétabilité des points extraits lors de changement de luminosité et de rotations dans l'image. Nous y avons aussi rajouté la robustesse et la pertinence des points en présence de bruit de type gaussien.

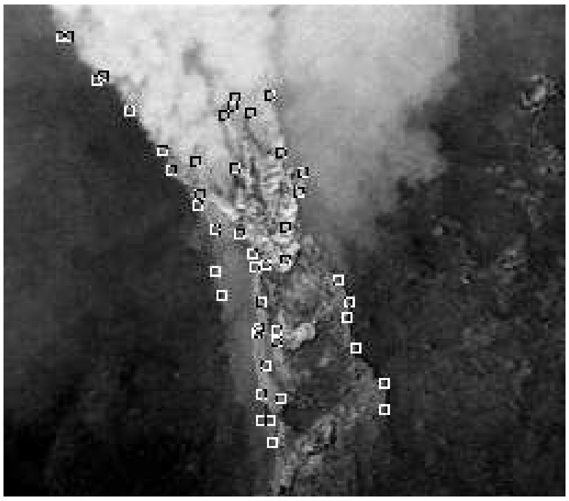
Sur les figures 2.7(b), 2.7(c) et 2.7(d), nous présentons un exemple des résultats obtenus. Si ceux-ci semblent très proches, en réalité, les détecteurs ne se comportent pas de la même façon. On remarque que sur le rocher proprement dit, les points semblent à peu près identiques, mais sur le fond et sur la fumée, on remarque des différences. En fait, pour avoir les mêmes points sur le rocher, il va falloir autoriser la détection de points sur le fond avec Harris par exemple. On comprend bien que le réglage des paramètres va être primordial.

Comme nous l'avons déjà constaté, l'une des caractéristiques fortes des images que nous devons traiter est la présence de bruits. Ceux-ci peuvent provenir de l'environnement lui-

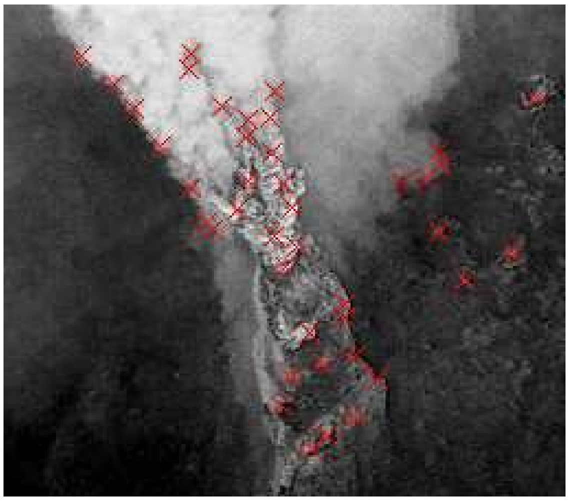
(a) Fumerolle : image initiale



(b) Fumerolle : détecteur Susan



(c) Fumerolle : détecteur Css



(d) Fumerolle : détecteur de Harris

Fig. 2.7 – Détecteurs de points appliqués à une image de la fumerolle sous-marine

même ou de la caméra par exemple. Comme cette donnée est primordiale, nous devons être sûrs qu'à la moindre perturbation dans l'image, les points d'intérêt pertinents détectés continueront de l'être. Nous avons donc testé la robustesse des trois détecteurs à l'ajout de bruit dans l'image. Notons que les résultats présentés ici sont issus d'une étude plus large effectuée sur différentes images, dont nous ne montrons qu'un aperçu ici.

Les figures 2.8(a) et 2.8(b) montrent le résultat du nombre de points détectés dans une même image avec ou sans la présence d'un bruit gaussien rajouté de variance égale à deux.

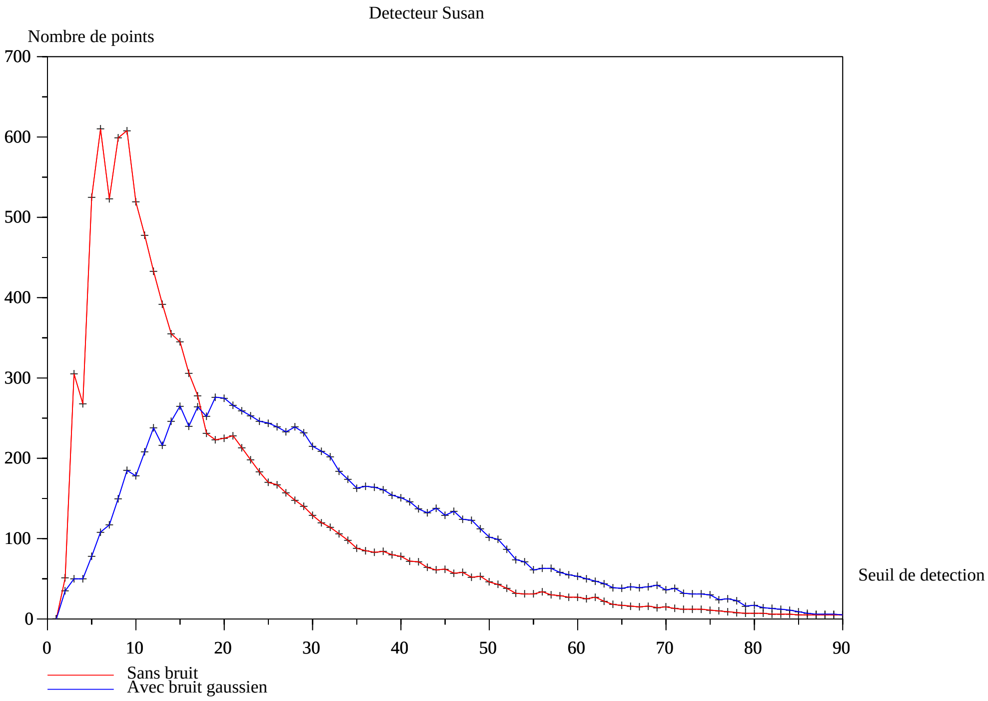
On le constate facilement sur ces figures, le détecteur de Susan se comporte nettement moins bien en présence de bruit. En effet, le détecteur de Susan doit calculer dans une zone la probabilité ou non d'une présence d'un coin. En théorie, Susan est robuste, car l'image comporte de bons contours, des coins précis et elle est binaire, donc du bruit gaussien ne va pas perturber le calcul. Mais dès que l'on se place dans le cas d'images naturelles, il est clair que cela devient extrêmement instable.

En revanche, le détecteur de Harris, se comporte nettement mieux en présence de bruit, comme on peut le constater sur la figure 2.8(b). Il en est de même pour le détecteur Css, car il fait appel à une approche multi-échelle qui a pour but de lisser le bruit dans l'image avant l'extraction des points.

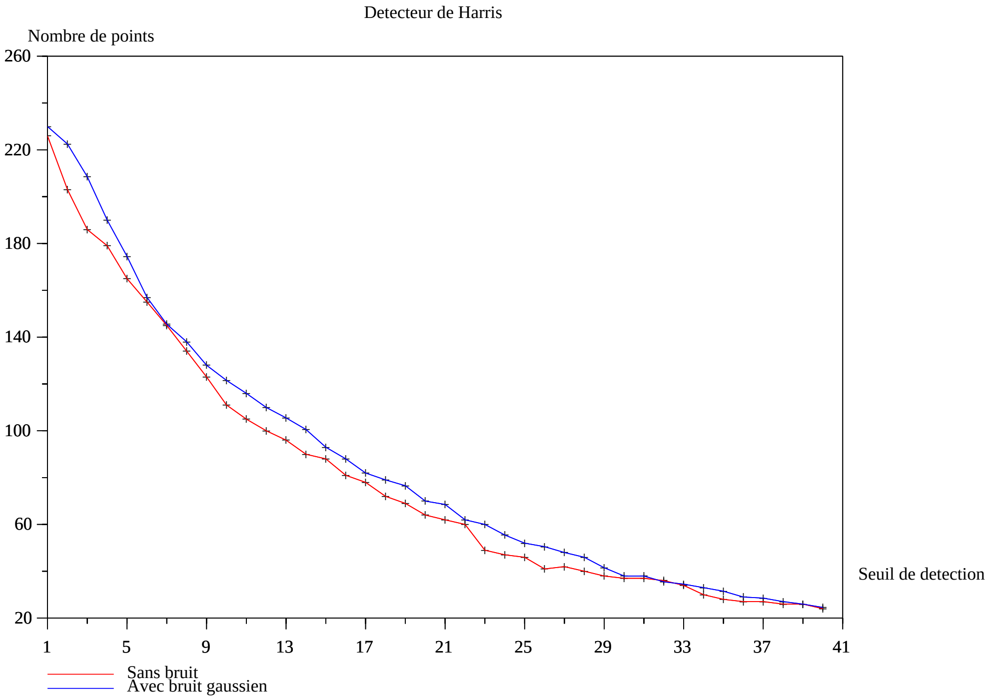
Un second test intéressant est d'observer l'évolution des points détectés lorsque l'on est proche de la convergence en faisant évoluer les paramètres des détecteurs. Nous avons pris comme critère de comparaison entre les détecteurs, le nombre de points détectés et leur emplacement (critère visuel). Là encore, Susan montre des faiblesses et l'on voit bien que la localisation des points stables est difficile. En effet, avant la « relative » convergence les points se déplacent autour du vrai point de façon aléatoire lorsqu'on fait varier les paramètres, comme le montrent les images de la figure 2.9. Cela est d'autant plus visible sur la partie du rocher.

En ce qui concerne le détecteur Css, sur la figure 2.10, on constate que la convergence atteinte, les points intéressants sont réellement stables, seuls certains points positionnés sur la fumée disparaissent. Egalement, le détecteur de Harris montre le même type de résultats (voir figure 2.11), à une exception notable près, qui est la détection de points stables sur le fond de la scène. Cela est dû au fait que dans le cas du détecteur de Css, il faut extraire des contours et donc d'utiliser des informations moins locales que pour Harris. Par conséquent, le contour n'a pas de signification. Notons bien que cela n'est pas gênant. En effet, le but de cette détection étant d'avoir des points stables, on peut même considérer que cela est un « plus » : le fait d'avoir des points éloignés dans l'image et à des distances 3D plus lointaines que celles des points sur le rocher, a pour effet de stabiliser le calcul des matrices de mouvements entre images (voir chapitre 4).

En définitive, nous avons constaté de manière générale, que le détecteur de Susan était

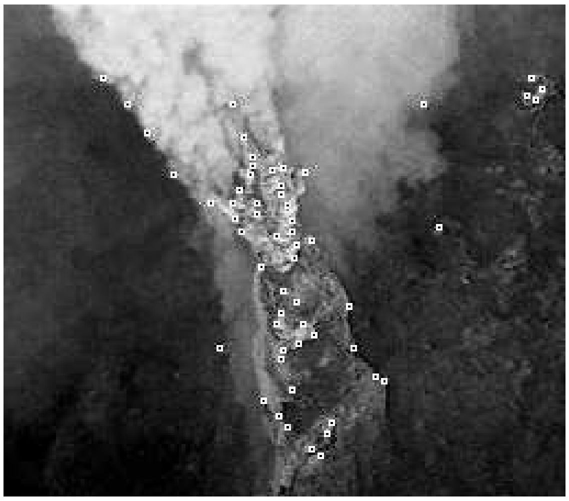


(a) Détecteur de Susan



(b) Détecteur de Harris

Fig. 2.8 – Robustesse des détecteurs en présence de bruit dans une image

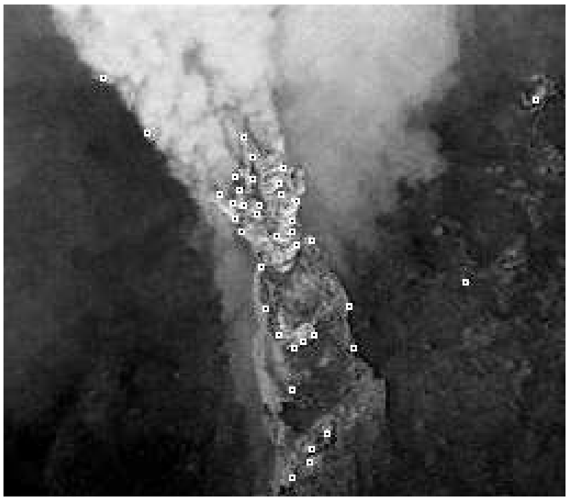


(a) Seuil 45 ; 57 points

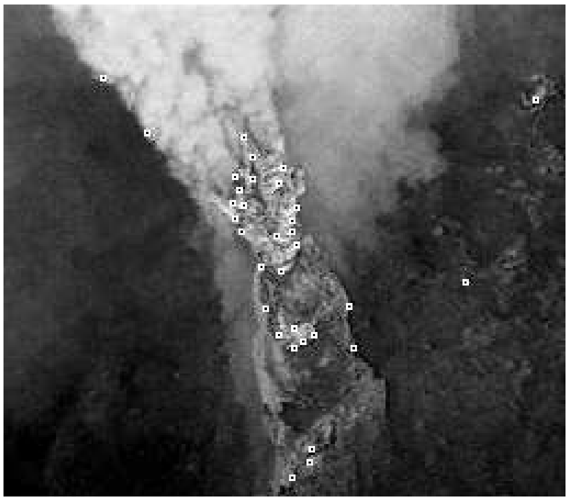
(b) Seuil 47 ; 54 points

(c) Seuil 49 ; 48 points

(d) Seuil 51 ; 43 points

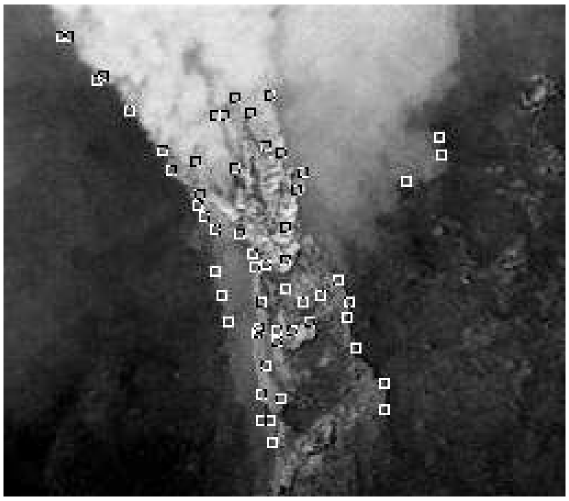


(e) Seuil 53 ; 38 points

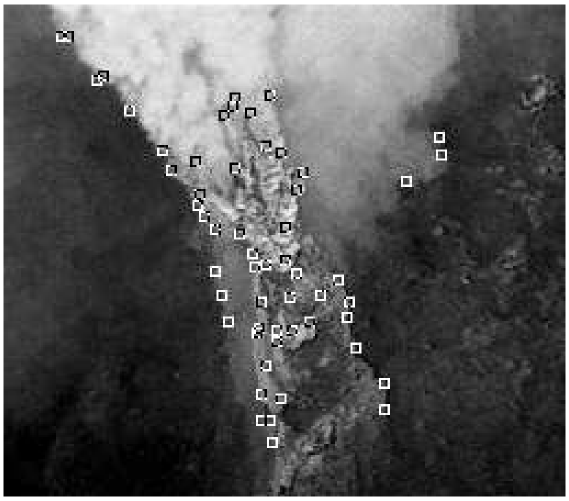


(f) Seuil 55 ; 33 points

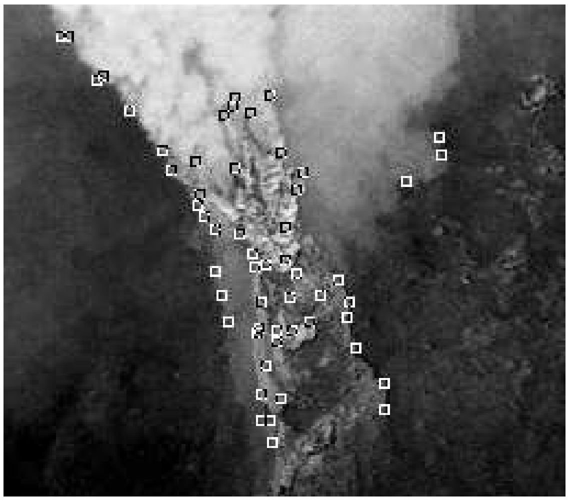
FIG. 2.9 – Effet du paramètre de seuillage du détecteur de SUSAN sur la détection et la localisation des points dans l'image de la fumerolle



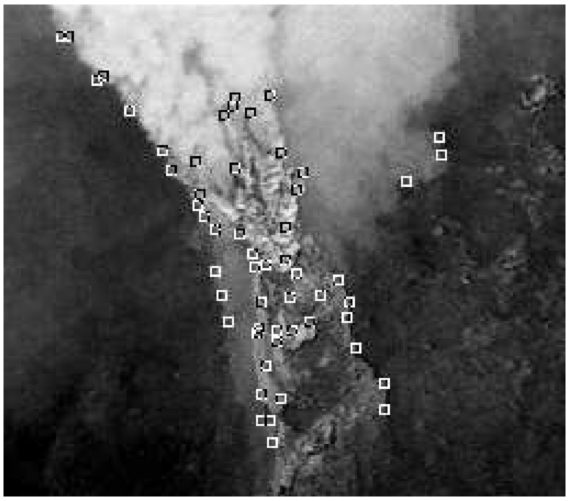
(a) Seuil 40 ; 56 points



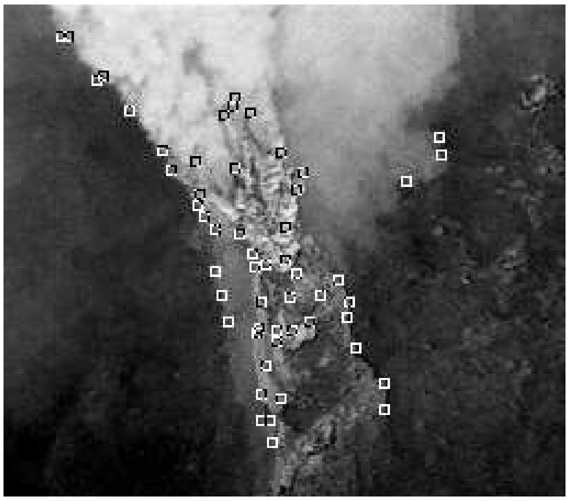
(b) Seuil 42 ; 56 points



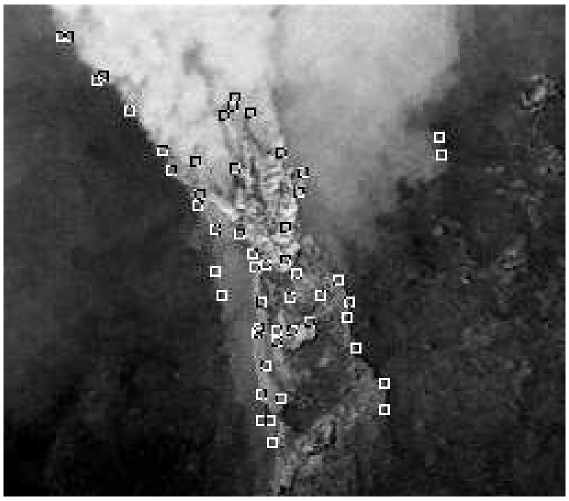
(c) Seuil 44 ; 55 points



(d) Seuil 46 ; 55 points

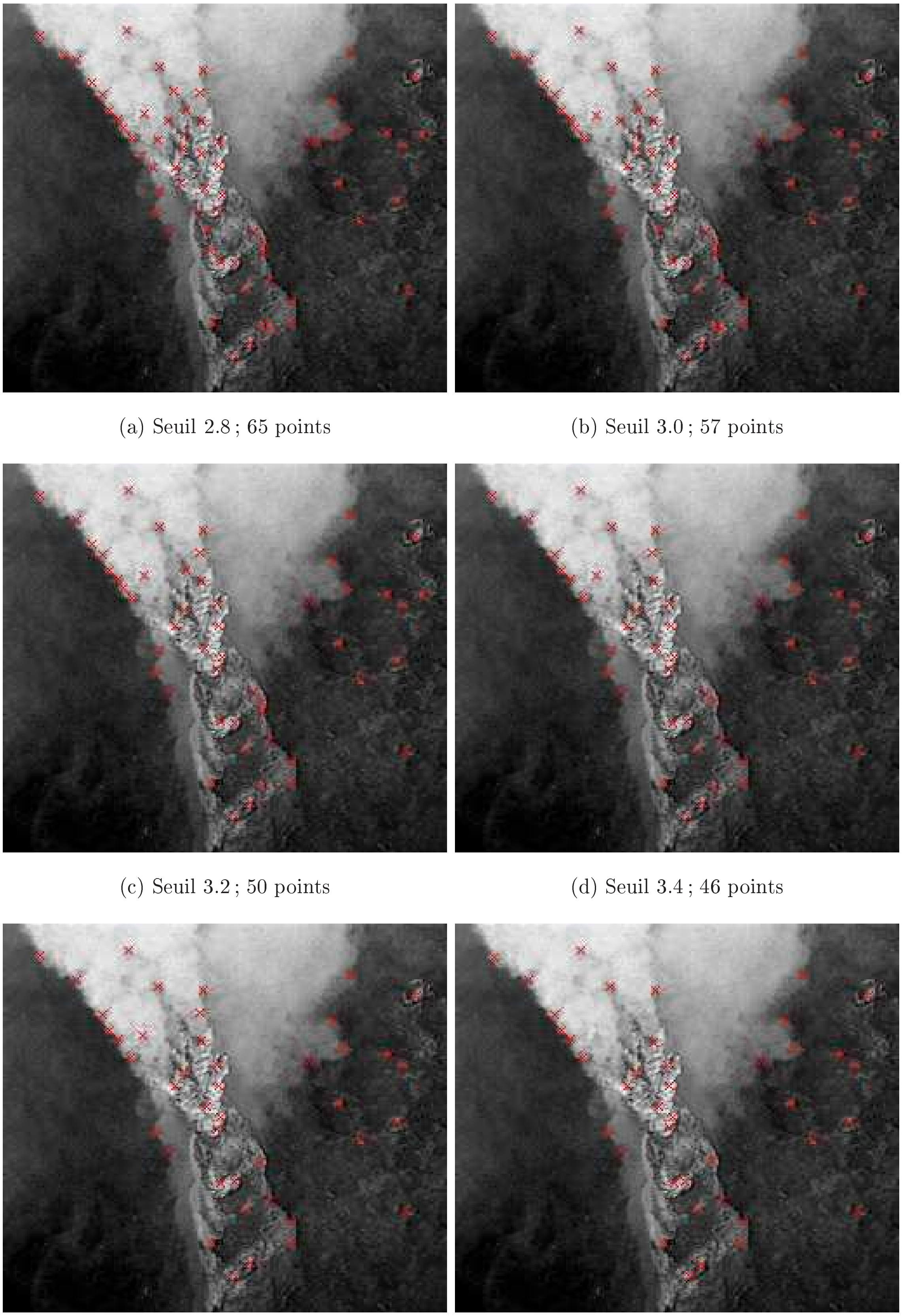


(e) Seuil 48 ; 54 points



(f) Seuil 50 ; 51 points

Fig. 2.10 – Effet du paramètre de seuillage du détecteur Css sur la détection et la localisation des points dans l'image de la fumerolle



(a) Seuil 2.8 ; 65 points

(b) Seuil 3.0 ; 57 points

(c) Seuil 3.2 ; 50 points

(d) Seuil 3.4 ; 46 points

(e) Seuil 3.6 ; 42 points

(f) Seuil 3.8 ; 40 points

FIG. 2.11 – Effet du paramètre de seuillage du détecteur de Harris sur la détection et la localisation des points dans l'image de la fumerolle

le moins bien adapté à nos types d'images et que les réglages de ses seuils et paramètres, était délicat et non générique pour les différentes images, y compris au sein d'une même séquence. En revanche, les deux autres détecteurs sont relativement similaires au niveau de leurs résultats, même si l'on observe une meilleure stabilité dans le détecteur de Harris sur des images comme le liège.

2.4 Conclusion

Après cette étude comparative, nous avons opté pour l'utilisation du détecteur de Harris, et ce pour plusieurs raisons. Tout d'abord celui-ci ne s'appuie pas sur une détection de contours, ce qui nous semble fortement important dans notre cas, car nos images ne sont pas structurées ; ensuite, le détecteur de Harris peut s'implémenter de façon optimale et rapide notamment en se basant les travaux de R. Deriche (Deriche, 1987) et il est tout à fait possible d'envisager une implémentation proche du « temps-réel » grâce aux capacités de calcul des nouveaux ordinateurs. De plus, le détecteur Css ne donne pas de très bons résultats sur des images de petite taille (128x128 pixels ou moins), car il nécessite une étape d'extraction de contours, peu fiable à ces résolutions. Enfin, le réglage des paramètres de Harris, même s'il n'est pas toujours aisé, peut se faire de façon assez générique. Signalons aussi que ce détecteur a été largement utilisé dans la littérature et est donc largement éprouvé et étudié depuis plusieurs années.

Chapitre 3

Appariement d'images naturelles

Lors d'une reconstruction en trois dimensions, qu'elle soit projective ou euclidienne, il est important d'avoir d'une part une grande confiance dans les caractéristiques extraites et d'autre part d'être en mesure de mettre en correspondance ces données. Dans certains cas, cela est aisé, mais dans le notre, cela est plus délicat, entre autres dû à la non-connaissance du mouvement entre images et à la non-connaissance de la scène.

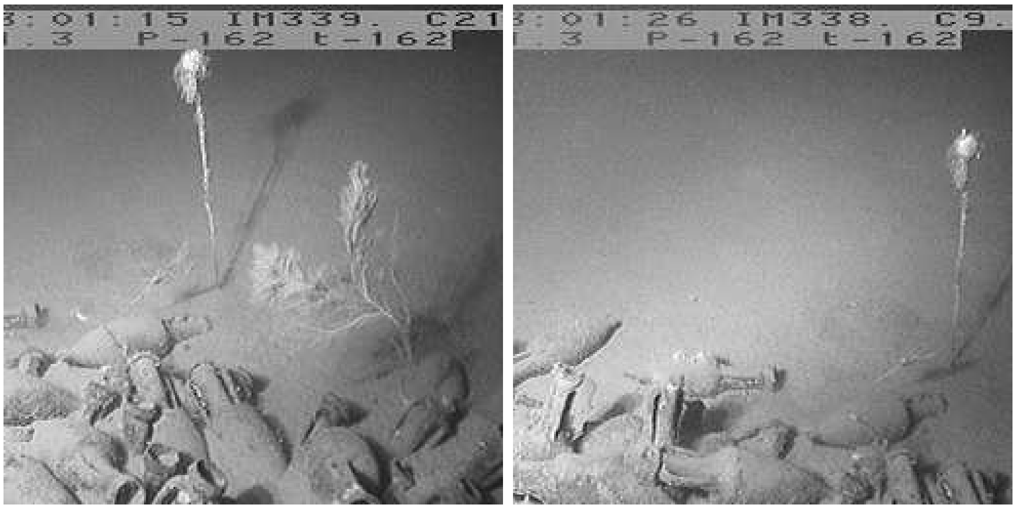
3.1 De la pratique à la théorie, un état de l'art des méthodes existantes

La notion d'appariement est totalement implicite lorsque nous regardons une scène. Nos deux yeux et notre cerveau savent parfaitement interpréter une même scène vue sous plusieurs angles et savent identifier quelles sont les parties communes. Par exemple, si l'on observe une table rectangulaire dans un jardin tout en nous déplaçant autour, retrouver les coins et les bords de la table, ne pose aucun problème, de même que faire la différence entre les chaises, le gazon ou les fleurs. Pourtant, sans vraiment le savoir ou s'en soucier, nous venons d'effectuer un travail difficile de mise en correspondance des coins et des bords de la table, et de plus, nous avons segmenté la scène et extrait les différents objets qui s'y trouvaient : la table, les chaises, le gazon et les fleurs. Tout ce cheminement totalement transparent pour notre esprit montre plusieurs choses. Tout d'abord, nous savons extraire différents types de primitives (coins, contours, textures, formes non structurées) ; ensuite, et c'est le propos de ce chapitre, nous savons les mettre en correspondance tout au long de notre déplacement, et ce, même s'il y a des occultations complètes ou partielles, s'il y a des changements d'éclairage, si l'on s'est rapproché ou éloigné, etc. Enfin, nous savons interpréter la scène, tâche cognitive très complexe. L'analogue, en vision par ordinateur, nécessite de faire appel à des techniques de représentation et traitement des connaissances et rentre dans la catégorie des traitements « haut-niveau », ce qui n'est pas le propos de cette thèse, et donc que nous n'aborderons pas.

Nous allons nous intéresser maintenant à cet aspect de mise en correspondance de caractéristiques, appelée aussi appariement (« matching » en anglais). Bien que cela ait été depuis longtemps abordé et étudié par la communauté scientifique, il n'en reste pas moins que c'est un exercice très difficile et un sujet de recherche toujours ouvert. Par exemple, en observant les images 3.1(a) et 3.1(b), s'il nous paraît évident de mettre en correspondance les amphores ou la grande algue, en pratique, les algorithmes existants sont mis en défaut.

En effet, nous sommes toujours confrontés à des difficultés lorsque l'on souhaite apparier des images où il y a de grands déplacements ou bien lorsque les conditions de prise de vue ont changé (focale différente, éclairage différent, ...) ou encore lorsqu'il y a des occultations. La liste est loin d'être exhaustive et on rencontre plus souvent ce type de situations dans le cas de scènes en environnement naturel que dans le cas d'images synthétiques ou d'intérieurs.

Dans ce chapitre, nous allons donc passer en revue un certain nombre d'approches classiques relativement robustes dans le cas d'images de scènes naturelles inconnues. Elles n'ont pas toutes vocation à être uniquement des méthodes d'appariement, mais permettent néanmoins de trouver entre deux images des correspondants. En ce qui nous concerne, cette étape d'appariement est primordiale car elle conditionne la qualité de la reconstruction 3D que l'on pourra effectuer par la suite.



(a) Amphores $n°92$ (b) Amphores $n°165$

FIG. 3.1 – Exemples d'images en environnement naturel très difficiles à apparier

3.1.1 Appariement par corrélation

Cette méthode est certainement la plus connue : elle est facile à utiliser et donne des résultats qui peuvent être bons ou suffisants sous certaines conditions ou pour certaines applications. L'idée majeure est la suivante : entre deux images à apparier, si le mouvement de la caméra entre les deux prises de vue n'est pas trop important et si les conditions d'éclairage n'ont pas trop varié, alors, il existe une grande similitude entre les valeurs des pixels correspondants aux mêmes parties physiques de la scène observée. En clair, l'hypothèse forte est que, dans un voisinage proche, l'image n'a sensiblement pas ou peu « bougé ». Il suffit donc de faire un calcul de mesure de ressemblance entre les pixels des images. Les méthodes varient alors sur le calcul de la mesure de similarité, qui pourra être plus ou moins bien adaptée à l'application visée. On mesure alors la ressemblance d'un point (u_1, v_1) de l'image 1 de signal I_1, à celle d'un point (u_2, v_2) de l'image 2 de signal I_2, sur un masque carré $(2n+1) \times (2n+1)$. L'équation 3.1 montre par exemple la mesure de similarité appelée SSD (*sum of squared differences*), qui minimise la somme des différences des intensités sur l'ensemble de la fenêtre de corrélation :

$$c_{u,v}^{Ssd}(d) = \sum_{du=-n}^{du=+n} \sum_{dv=-n}^{dv=+n} (I_2(u_2 + du, v_2 + dv) - I_1(u_1 + du, v_1 + dv))^2 \tag{3.1}$$

où I_1 et I_2 sont les fonctions d'intensité dans chacune des images. Notons que ce critère SSD est très sensible aux différences d'illumination entre les deux images.

Une version plus élaborée de SSD est le critère ZNSSD (*zero-mean sum of squared dif-*

ferences) qui minimise la somme des différences des intensités filtrées par la moyenne sur l'ensemble de la fenêtre de corrélation, normalisée par la variance locale des intensités. Ce critère permet de s'affranchir des différences de gains et d'offsets des caméras.

On peut aussi citer le critère de *cross-correlation*, qui est le produit scalaire des deux vecteurs formés par les intensités des images, et qui s'écrit :

$$c_{u,v}^{CC}(d) = \sum_{i,j} I_1(u+i, v+j) I_2(u+d+i, v+d) \tag{3.2}$$

En pratique, ce critère favorise les zones d'intensité élevée, donc pas nécessairement similaires à la fenêtre de corrélation de l'image de référence, ce qui fait qu'il n'est guère utilisable. En pratique, on lui préfèrera sa version normalisée.

Le lecteur trouvera une étude fournie et complète dans les travaux de thèse de J. Blanc (Blanc, 1998) sur les différentes mesures de corrélation et sur l'appariement que l'on peut effectuer grâce à cette approche. F. Devernay (Devernay, 1997) a également mené une étude durant sa thèse sur l'utilisabilité de ces mesures de corrélation pour faire de la stéréoscopie.

L'intérêt évident de ces mesures de corrélation est le fait qu'il n'est pas nécessaire d'extraire des primitives pour calculer la similarité entre deux images. En revanche, il est clair que s'il y a des occultations ou des grands déplacements ou encore des changements d'éclairage, ces mesures ne sont plus adaptées et donnent des résultats faux.

De plus, ces méthodes sont purement locales et ne prennent pas en compte une quelconque rigidité qui pourrait être contenue dans la scène ou encore des distances entre des primitives. Enfin, cette approche n'utilise que des pixels, sans aucune autre sorte d'information : réponse à un filtre, dérivée, etc. Cela rend les algorithmes rapides et faciles à utiliser mais peu robustes.

Il est néanmoins possible d'affiner ces approches en n'utilisant que des points d'intérêt extraits préalablement dans chaque image. Ainsi, on diminue la recherche des appariements uniquement entre deux ensembles de points candidats, abaissant d'une part le temps de calcul et d'autre part, le taux d'erreurs.

3.1.2 Le suivi de caractéristiques

Plutôt que de tenter d'apparier deux ensembles de points entre deux images souvent prises sous des points de vues très différents, il peut être intéressant, lorsque le système d'acquisition le permet, d'essayer de suivre ces points tout au long de la séquence d'images en utilisant la cohérence temporelle entre images. On parle alors de « tracking » ou « suivi » de caractéristiques. Le but de cette méthode est de suivre entre plusieurs images successives d'une même scène, le plus grand nombre de primitives. Cela peut être des coins, des points d'intérêt, des contours, des droites, des formes, ... C. Tomasi et T. Kanade (Tomasi et Kanade, 1991) ont introduit cette approche avec l'idée simple de suivre des points tout au long d'une séquence grâce à une mesure de corrélation (partant du principe qu'entre deux images successives, la variation en luminance est très faible) et de les marquer. J. Shi et C. Tomasi (Shi et Tomasi, 1994) ont également travaillé par la suite sur la qualité intrinsèque des points à suivre : ils doivent répondre d'une certaine robustesse lors de transformations affines par exemple. L'avantage majeur de cette approche réside dans le fait qu'une seule caméra peut suffire, ce qui est notre cas et qu'il n'est pas nécessaire de connaître son mouvement. Là encore, de nombreuses approches ont été développées, mais la littérature étant abondante dans ce domaine, nous laissons le soin au lecteur de s'y reporter.

Le suivi de caractéristiques est toutefois un problème délicat dans un grand nombre de situations d'une part et dépend également des primitives que l'on souhaite suivre d'autre part. Il est clair que suivre des contours déformables lors d'un déplacement rapide est plus difficile à réaliser que suivre des coins dans un environnement structuré à très faible vitesse. En effet, les contours peuvent se scinder ou encore changer de taille, ce qui rend encore plus difficile l'appariement.

L'intérêt du suivi peut être évident lorsque l'on connait le déplacement de la caméra, car il est alors possible de prévoir, à une erreur bornée près, la position probable d'un point dans l'image. Egalement, le suivi peut s'effectuer sur de longues séquences vidéo sans nécessiter l'extraction des points d'intérêt à chaque image. Par exemple, on peut extraire des points dans l'image de départ, puis effectuer une simple corrélation sur ces points sur les images suivantes, puis refaire une extraction de points lorsque le nombre de points appariés est en-dessous d'un pourcentage du nombre de points initiaux, comme l'a fait par exemple S. Christy (Christy, 1998).

Néanmoins, pour que cela fonctionne bien, il faut que les séquences vidéo soient denses et que les déplacements entre images soient assez faibles. Sous ces conditions, on obtient de bon résultats, même avec des images complexes. De plus et c'est un avantage, cette méthode fonctionne dans le cas d'une seule caméra non calibrée.

3.1.3 Caractérisation par des invariants locaux

Les méthodes de suivi ou d'appariement par corrélation exploitent uniquement l'hypothèse de similarité locale de la fonction intensité entre deux images après déplacement de la caméra. Des approches plus récentes visent à caractériser les points extraits en leur associant un vecteur d'attribut caractéristique. Une approche récemment développée par C. Schmid dans sa thèse (Schmid, 1998) (Schmid et Mohr, 1997), permet de caractériser des points d'intérêts extraits dans une image par des invariants locaux. Cette idée est utilisée pour retrouver des images dans des bases de données d'images. La caractérisation utilisée est inspirée des travaux de J. J. Koenderinck (Koenderinck et Doorn, 1987). En fait, à chaque point d'intérêt détecté, on a associé un vecteur à neuf composantes, qui sont des valeurs invariantes à un certain nombre de transformations, comme par exemple, le changement de luminosité ou les rotations. En revanche, il n'est pas invariant aux changements d'échelles et il faudra donc calculer ce vecteur à plusieurs échelles et on soulignera de plus qu'il n'est pas invariant non plus aux occultations. Le vecteur d'invariants différentiels exprimé en notation d'Einstein est alors le suivant :

$$\mathbf{v} = \begin{pmatrix} L \\ L_i L_i \\ L_i L_{ij} L_j \\ L_{ii} \\ L_{ij} L_{ji} \\ \varepsilon_{ij}(L_{jkl} L_i L_k L_l - L_{jkk} L_i L_l L_l) \\ L_{iij} L_j L_k L_k - L_{ijk} L_i L_j L_k \\ -\varepsilon_{ij} L_{jkl} L_i L_k L_l \\ L_{ijk} L_I L_j L_k \end{pmatrix} \tag{3.3}$$

où L est la fonction de luminance convoluée avec une gaussienne. Avec cette notation, un indice i correspondant à la somme par rapport aux variables telle que :

$$L_x = \frac{\partial}{\partial x} L$$

et

$$L_i = \sum_i L_i = L_x + L_y \tag{3.4}$$

$$L_{ij} = \sum_i \sum_j L_{ij} = L_{xx} + L_{xy} + L_{yx} + L_{yy} \tag{3.5}$$

Toujour en utilisant cette notation, les autres composantes représentent la somme des dérivations par rapport à l'ensemble des variables, ce qui donne par exemple :

$$L_i L_{ij} L_j = \frac{\partial L}{\partial x} \frac{\partial^2 L}{\partial x^2} \frac{\partial L}{\partial x} + 2 \frac{\partial L}{\partial x} \frac{\partial^2 L}{\partial x \partial y} + \frac{\partial L}{\partial y} \frac{\partial^2 L}{\partial y^2} \frac{\partial L}{\partial y} \tag{3.6}$$

Enfin, $\varepsilon_{xy} = -\varepsilon_{yx} = 1$ et $\varepsilon_{xx} = \varepsilon_{yy} = 0$.

A partir des points d'intérêt et de leur vecteur, C. Schmid évalue la pertinence de l'image par rapport à une base de donnée d'images. Elle calcule la mesure de ressemblance entre deux points p_1 et p_2 et leurs vecteurs $\mathbf{v_1}$ et $\mathbf{v_2}$ avec la distance de Mahalanobis :

$$d(p_1, p_2) = \sqrt{(\mathbf{v_2} - \mathbf{v_1})^T \mathbf{\Lambda}^{-1} (\mathbf{v_2} - \mathbf{v_1})} \tag{3.7}$$

où $\mathbf{\Lambda}$ est la matrice de covariance de taille 9x9 des composantes du vecteur. Le problème est que cette matrice est calculée expérimentalement sur des images d'une base de données. S'il est relativement facile de modéliser le bruit dans l'image, il est par contre difficile de modéliser « a priori » les erreurs de localisation des points, ce qui pose alors le problème d'une bonne estimation de cette matrice $\mathbf{\Lambda}$.

Cette approche a montré de bons résultats dans le cas de recherche d'image dans une base de données, mais montre ses limites dans le cas où les objets ont des structures 3D importantes : l'approche ne fonctionne bien que dans le cas d'objets relativement plans. De plus, elle nécessite des temps de calculs assez longs.

Se basant sur ces travaux, Y. Dufournaud (Dufournaud, 2001) a appliqué cette approche à l'appariement de points dans des images naturelles, mais, là encore, les scènes sont pratiquement planes. Plutôt que de rechercher une sorte de moyenne entre tous les points d'une image et ceux d'une base de donnée, il va rechercher directement à apparier les points en utilisant ces vecteurs. Il impose également des contraintes de voisinage pour affiner la recherche d'appariement et pour éviter des recherches exhaustives entre tous les points. Les résultats obtenus sont de bonne qualité et l'approche fonctionne avec des images de scènes naturelles inconnues, quoique suffisamment planes. Y. Dufournaud a également travaillé sur l'optimisation du détecteur de Harris pour le robustifier lors de grands changements d'échelles entre deux images (Dufournaud et al., 2000), ce qui a pour effet de limiter le calcul des vecteurs d'invariants à différents échelles.

3.1.4 Autres méthodes plus spécifiques

Enfin, pour conclure cet état de l'art, nous évoquons ici des méthodes spécifiques développées en vue d'un type d'application bien précise ou dans un but autre que le simple appariement.

- L'appariement dense est une méthode qui est utilisée pour estimer le mouvement entre deux images. Le principe est de prendre un nuage dense de points et d'apparier au mieux l'ensemble des points sans en rejeter. Ainsi, on utilise à la fois des notions de localité (points), mais aussi de globalité (nuage de points). M. Lhuiller et L. Quang (Lhuillier et Quan, 2000) ont proposé récémment de combiner des approches géométriques globales et locales avec des appariements denses pour la synthèse de nouvelles vues.
- On trouve aussi des méthodes complexes pour apparier des scènes lors de grands déplacements. M. Lhuiller (Lhuillier, 1999) propose de joindre tous les points détectés

par une triangulation de type Delaunay et d'apparier en fait selon des contraintes de ressemblance au niveau de la triangulation. Ainsi il peut apparier des scènes naturelles, même lors de grands déplacements, mais de type "translation" essentiellement. Notons que ces méthodes sont néanmoins coûteuses en temps de calcul.

– En imagerie médicale, on trouve des images particulières, car même si elles sont très complexes, elle sont assez ressemblantes. Par exemple, les images de coupes transversales du cerveau ont toutes des caractéristiques communes : matière grise, boîte crânienne, etc. Selon les applications, il existe plusieurs types de méthodes d'appariement. En effet, on peut essayer de recaler l'image d'une coupe du cerveau avec une image de référence. Il faut donc alors déformer l'image et pour cela, il est nécessaire de passer par des phases d'apprentissage pour évaluer la meilleure déformation. On peut aussi utiliser des techniques basées sur les croissances de régions pour coller à l'image de référence ou à un modèle. En règle générale, pour ce type d'images, la présence d'un opérateur est nécessaire, ne serait-ce que pour spécifier les parties communes entre images et celles qui doivent se déformer. Les travaux de thèse de J. Montagnat fournissent au lecteur une étude précise et de qualité sur ces problèmes de segmentation dans des images médicales (Montagnat, 1999).
– Dans un autre domaine, en cartographie et en localisation maritime et sous-marine, le problème de la mise en correspondance se pose souvent en termes de recalage d'une carte globale par rapport à une carte locale déjà acquise avec, éventuellement des modalités sensorielles différentes. Par exemple, sous l'eau, un sous-marin calcule une carte locale de l'environnement qui l'entoure, et il faut le localiser par rapport à une carte globale connue. Dans ce cas de figure, certains auteurs ont utilisé des méthodes robustes de minimisation et de résolution d'équations aux dérivées partielles, comme l'ont montré L. Lucido *al.*(Lucido et al., 1996).
– On trouve également des méthodes basées sur le flot optique, concept relativement ancien (Gibson, 1950) (Horn et Schunck, 1981), où l'on cherche à estimer le mouvement à travers un champ de vecteur dense des vitesses apparentes entre des images d'une séquence video. L'hypothèse que l'on fait est de considérer que l'intensité (ou la couleur) est conservée au cours du déplacement. Ensuite, il est possible de « segmenter » les différentes parties de l'image en fonction des mouvements respectifs des différents objets obtenus par le calcul du flot optique. Ces méthodes basées sur le calcul du flot optique sont très nombreuses et nous renvoyons le lecteur aux travaux de thèses de T. Papadopoulo (Papadopoulo, 1995) et de P. Kornprobst (Kornprobst, 1998) par exemple. Dans cette catégorie, on peut citer les approches développées par J.-M. Odobez et P. Bouthémy (Odobez et Bouthemy, 1995) basées sur des approches multi-résolution et bayésiennes dont le but est de compenser les mouvements dans des séquences video.

3.1.5 Utilisation de la géométrie épipolaire pour le rejet de mauvais appariements

Une autre méthode très en vogue utilise les propriétés de la géométrie épipolaire. Elle sert ici à rejeter les mauvais appariements. En effet, dans toutes les approches précédement décrites, nous n'avons pas évoqué ce problème important en pratique. La géométrie épipolaire lie deux images I_1 et I_2 entre elles et de ce fait, les points appariés doivent satisfaire la contrainte fondamentale représentée par l'équation 3.8. En clair, il existe une matrice $\mathbf{F}$ de taille 3x3 et de rang 2 telle que pour deux points appariés q_1 et q_2, qui sont les projections d'un même point 3D Q dans les images I_1 et I_2, on ait la relation suivante :

$$q_2^T \mathbf{F_{12}} q_1 = 0 \tag{3.8}$$

A partir de cette équation et après avoir effectué la mise en correspondance, on peut alors valider ou non les appariements prédits. Si deux points appariés ne vérifient pas la relation, alors ce sont de faux appariements. L'hypothèse de départ est donc de connaître cette matrice $\mathbf{F_{12}}$, or nous le verrons, ce n'est pas toujours le cas et son calcul est loin d'être simple, en particulier dans le cas d'une seule caméra et d'images complexes. Accessoirement, cela suppose de connaître au minimum 8 points bien appariés pour estimer cette matrice avec un algorithme linéaire. L'utilisation de la contrainte fondamentale 3.8 permet donc de rejeter les mauvais appariements. Le chapitre 4 traitant de la géométrie épipolaire, nous y renvoyons le lecteur pour davantage de détails.

3.1.6 Conclusion préliminaire / Synthèse

Que dire alors de l'appariement et comment réaliser cette étape ? A la vue de ce qui vient d'être dit, il est clair que toutes les solutions ne sont pas adaptées à notre problème. Tout d'abord, nous avons pris le parti d'extraire des points d'intérêt et donc il convient de les exploiter autant que faire se peut. Ensuite, dans la mesure où nous sommes dans le cas d'images acquises avec une seule caméra non calibrée, et que nous ne connaissons pas son mouvement, nous ne pouvons pas utiliser cette information pour faire de la prédiction ou de la vérification de mise en correspondance. De plus, nos objets étant non structurés (pas de coins, de droites où de formes simples), nous ne pouvons pas non plus utiliser des contraintes géométriques.

Nous avons donc décidé d'utiliser une structure multi-échelles pour résoudre notre problème. Nous allons le voir, cela va nous permettre de réduire les temps de calcul tout en assurant une bonne qualité d'extraction et d'appariement de points. Nous allons reprendre l'idée de base de C. Schmid et d'Y. Dufournaud en ce qui concerne la caractérisation de points d'intérêt et développer notre approche.

3.2 Pyramide d'images et robustification des appariements

Dans le chapitre précédent, nous avons choisi d'extraire des points d'intérêts, mais dans le cas de nos images, cela s'avère insuffisant pour faire de la reconstruction 3D. En effet, la qualité des images est telle que nous risquons d'extraire des points qui ne sont ni robustes, ni intéressants. Nous souhaitons donc trier les points extraits et les caractériser. Le problème est de trouver un ou plusieurs critères satisfaisants en fonction du but à atteindre.

Dans un premier temps, nous voulons caractériser les points robustes et stables et dans un second temps, nous voulons trouver les points qui seront appariés avec un grand degré de confiance. Les autres points, dans les deux cas, seront classés selon un critère que nous allons définir dans la suite.

3.2.1 Pyramide d'images

Notre choix s'est donc porté vers une approche multi-échelles via la construction et l'utilisation d'une pyramide d'images. Cette approche n'est certes pas nouvelle (Witkin, 1987), mais est bien adaptée à notre problème. On pourra trouver par exemple dans le livre de J.-M. Jolion et d'A. Rosenfeld davantage d'informations concernant le formalisme associé à l'approche multi-échelle (Jolion et Rosenfeld, 1994). De plus, dans notre optique d'implémentation temps-réel, les pyramides d'images peuvent être implémentées très facilement et leur temps de calcul est très raisonnable. Notons aussi que certains constructeurs de cartes d'acquisition proposent la construction de cette pyramide en « hardware » et donc en temps-réel.

Une des caractéristiques fortes de nos images est le bruit qu'elles contiennent. Celui-ci peut-être très grand au point que sur des images comme celles du Titanic, le rapport signal/bruit est très faible. L'idée de la pyramide est alors naturelle, car elle a pour vocation d'éliminer ce bruit en diminuant l'échelle et la résolution. En effet, un signal lissé plusieurs fois ne conserve que ses fortes variations, ce qui revient à dire que plus on lisse un signal, moins il reste de pics d'intensité, mais que ceux restants sont significatifs. Autrement dit, les points détectés à une faible résolution pour un signal ayant subi plusieurs lissages, sont robustes par rapport au bruit. En contre-partie, ce lissage risque de dégrader la localisation des points.

Le principe de l'algorithme de construction de la pyramide est simple :

```
Répeter N fois
   1- lissage de l'image courante par une gaussienne
   2- sous-échantillonnage de l'image (du type ''1 pixel sur 2'')
```

On obtient alors la pyramide suivante :
$I_{k,0}$: Image k, résolution maximale, niveau 0
$\vdots$
$I_{k,l}$: Image k, niveau l (ce qui correspond à l boucles de l'algorithme)

Le lissage par la gaussienne s'obtient par une convolution qui s'écrit de façon formelle comme suit :
- soit $I_{k,l}$ l'image k au niveau l
- soit $m^i_{k,l} = (u^i_{k,l}, v^i_{k,l})^T$ un point de $I_{k,l}$

On obtient donc $I_{k,l+1} = f(I_{k,l})$ où f est la composée d'une fonction de convolution et d'une fonction d'échantillonnage.

La convolution s'écrit dans le cas continu :

$$f(x) \otimes g(x) = (f \otimes g)(x) = \int_{-\infty}^{+\infty} f(x-y).g(y)dy \tag{3.9}$$

Ce qui s'écrit dans le cas discret et pour deux variables par :

$$f(x,y) \otimes g(x,y) = \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} f(k,l).g(x-k, y-l) \tag{3.10}$$

En pratique, il est difficilement concevable d'aller en deçà de la résolution 64x64 car l'image obtenue à cette résolution est quasi-inexploitable. Un autre problème qui peut se révéler gênant et être source d'erreurs, est celui de la discrétisation. En pratique, nous nous contentons de sous-échantillonner en prenant un pixel sur deux en lignes et en colonnes ; mais cette méthode est arbitraire et dans certains cas peut engendrer un comportement insidieux. Par exemple, supposons que nous ayons une fonction de type « step » comme sur la figure 3.2 et que nous la lissions avec une gaussienne, alors le sous-échantillonnage appliqué peut-être différent.

On le constate sur la figure 3.2, la discrétisation obtenue n'est pas la même selon les deux cas. Or, comme les détecteurs de points travaillent sur les données entières (en fait les pixels) et qu'ils se basent sur les valeurs calculées à partir des celles-ci, on comprend bien que le résultat dépend de ce sous-échantillonnage. En pratique, cet effet est d'autant plus sensible que la résolution est faible et surtout sur des données pixels représentant des structures géométriques clairement identifiables. Pour pallier cette contrainte, nous réglons

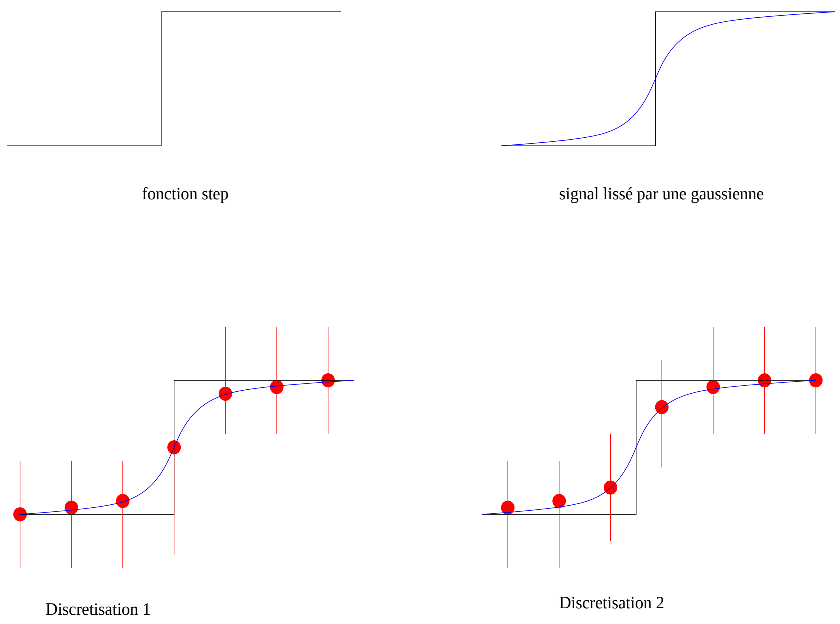


FIG. 3.2 – Problème de discrétisation

soigneusement nos paramètres de lissage et du détecteur de Harris : en effet si un point est détecté à un seuil donné σ_A (détecteur de Harris) pour une discrétisation A, alors il s'avère qu'il est aussi détecté pour un seuil σ_B et une discrétisation B.

3.2.2 Appariement « pyramidal »

3.2.2.1 L'algorithme d'appariement

Après la construction et l'extraction des points d'intérêt pour chaque niveau, nous allons effectuer ce que nous appelons un appariement « pyramidal », car comme son nom l'indique, il se fait au sein de la pyramide. Le but de cette opération est de déterminer quels sont les points robustes et significatifs au niveau 0, donc dans l'image initiale.

Cette méthode est la partie innovante de ce chapitre. En effet, de par la nature même des images et le fait que nous n'ayons aucune connaissance sur celles-ci, il nous paraît évident qu'une sélection « intelligente » de points caractéristiques s'avère indispensable. C'est pourquoi nous partons du principe que seules les fortes discontinuités du signal (c'est-à-dire de l'image) sont représentatives de l'image et donc de l'objet. La pyramide d'images a pour effet d'atténuer les perturbations du signal et de ne conserver que celles qui sont significatives, donc stables au sens de la détection de Harris.

L'appariement « pyramidal », s'effectue de la façon suivante :
- extraction des points de Harris à chaque niveau de la pyramide

– projection des points d'un niveau k dans un niveau inférieur $(k-1)$
– construction d'un arbre au fur et à mesure

La figure 3.3 décrit le procédé d'appariement en 2D. La construction d'un arbre va nous permettre de trouver très rapidement les points du niveau 0 ayant un correspondant au niveau le plus élevé de la pyramide. Ceux-ci seront alors considérés comme stables et robustes et serviront de base à la construction d'un graphe.

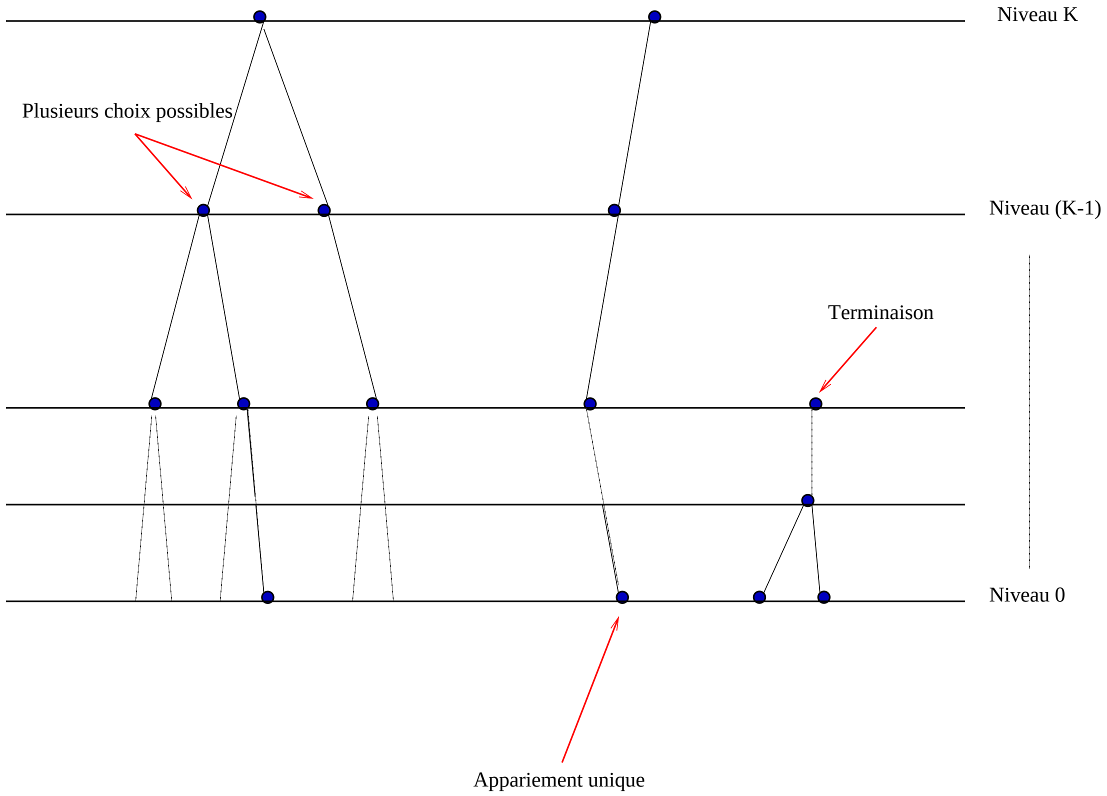


Fig. 3.3 – Principe de l'appariement pyramidal

La projection des points se fait via une simple recherche dans une zone donnée (voir figure 3.4). Nous n'utilisons pas de mesure de corrélation car les niveaux de gris ont été remaniés et cela n'aurait que peu de sens. La fenêtre de recherche est variable et ajustable en fonction des paramètres utilisés pour construire la pyramide et par le détecteur de Harris.

L'algorithme de recherche et de construction de l'arbre est le suivant :
– Pour chaque point $p_k(x, y)$ du niveau $k > 0$
  – créer une fenêtre de recherche de taille λ centrée en $(2x, 2y)$
  – pour tout point $p_{k-1}(x', y')$ contenu dans cette fenêtre, mettre à jour l'arbre d'appariement

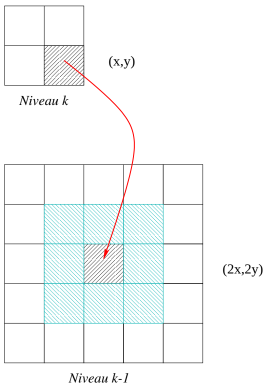


FIG. 3.4 – Projection d'un point d'un niveau k vers le niveau $(k-1)$

Notons que l'arbre obtenu comporte pour chaque noeud, ses fils ou et ses parents s'il en a.

Une fois encore, le problème du réglage de ce paramètre λ se pose. En pratique, nous avons estimé qu'une fenêtre de recherche de taille 4x4 ou 5x5 suffisait amplement. En effet, lors de la construction de la pyramide, nous devons convoluer un point $p(x, y)$ avec une gaussienne $g(x, y, \sigma)$ avant le sous-échantillonnage spatial. Or, nous savons que : $\int_{-2\sigma}^{+2\sigma} g(t)dt = 0.99$. En clair, le support le gaussienne contribuant au résultat de la convolution avec f est de $[4\sigma]^2$ dans le cas 2D. Comme nous discrétisons le signal avant de le sous-échantillonner, nous prenons la partie entière supérieure :

$$\lceil [g \otimes f(x-2\sigma, y-2\sigma), g \otimes f(x+2\sigma, y+2\sigma)]^2 \rceil \tag{3.11}$$

où $\lceil x \rceil$ désigne la partie entière supérieure de x. Ce faisant, nous n'occasionnons pas de perte dans le calcul, car de toute façon les termes au delà de $(x, y) \pm 2\sigma$ ne sont pas pris en compte, étant quasiment nuls.

Puis nous sous-échantillonnons : le support « actif » autour du point $p_{k-1}(x, y)$ au niveau $(k-1)$ devient alors :

$$\frac{1}{2} \lceil [g \otimes f(x-2\sigma, y-2\sigma), g \otimes f(x+2\sigma, y+2\sigma)]^2 \rceil \tag{3.12}$$

Ainsi lors de l'appariement pyramidal et de la projection d'un point d'un niveau k à $k-1$, le support de recherche est donc deux fois celui du niveau inférieur, ce qui donne :

$$\lceil [g \otimes f(x-2\sigma, y-2\sigma), g \otimes f(x+2\sigma, y+2\sigma)]^2 \rceil \tag{3.13}$$

En pratique, σ varie entre 1 et 2.5, ce qui donne des zones de recherche qui vont jusqu'à

6x6. Une amélioration serait de considérer la discrétisation optimale, à savoir choisir la partie entière la plus proche. Dans la réalité, cela n'influence que très peu le résultat.

L'arbre construit, nous obtenons un arbre de hauteur maximale, à savoir, la hauteur de la pyramide, et des sous-arbres de hauteurs diverses. Ces derniers ne sont pas inintéressants pour autant : en effet, ils nous donnent des indications sur la validité de certains points. Néanmoins ceci repose sur une hypothèse forte : qu'il y ait *injectivité entre les points du niveau k et* $(k-1)$. Autrement dit, qu'à un point du niveau $(k-1)$ il existe au moins un point apparié au niveau k (chaque point détecté a au moins un fils au niveau inférieur). Pour que cela soit toujours valide, il faut imposer certaines conditions sur les paramètres. Il est clair que si l'on garde le même paramètre σ pour la gaussienne de lissage lors du calcul des points de Harris, cela n'a pas beaucoup de sens, car le signal est affecté différemment.

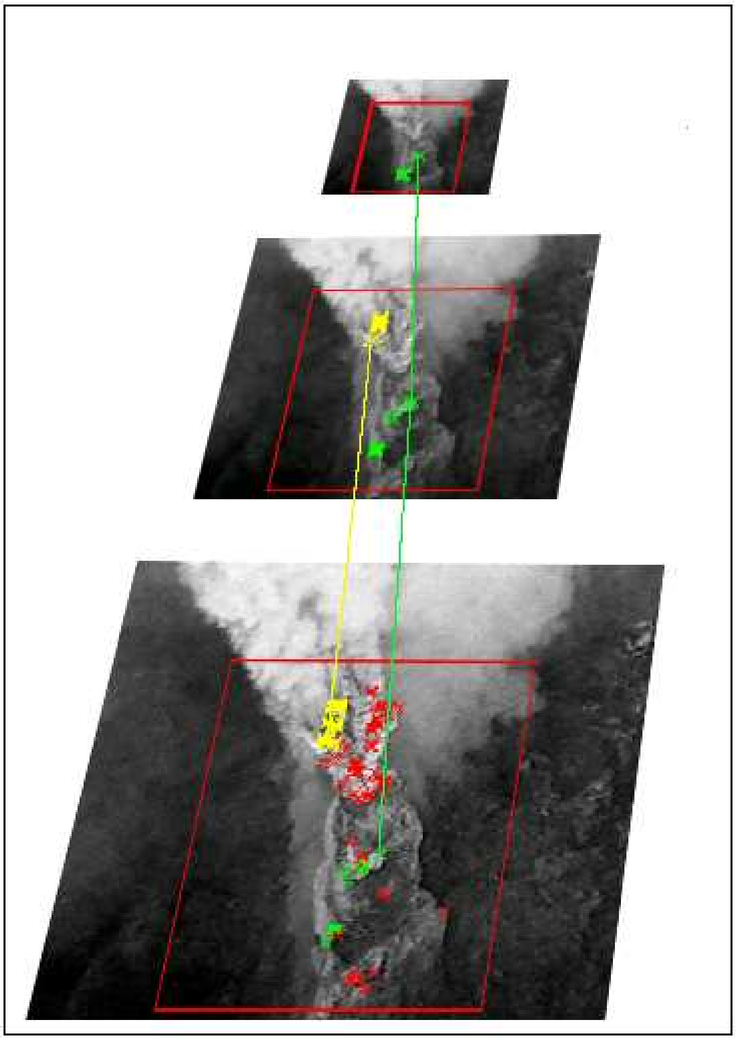


FIG. 3.5 – Exemple d'appariement pyramidal sur une pyramide d'images

La figure 3.5 montre le résultat de cet appariement pyramidal. Les points de couleur verte représentent les points stables et robustes ; ceux de couleur rouge sont assimilés quant à eux à du bruit ; enfin les points de couleur jaune sont des points de seconde importance, car ils ne sont pas robustes au sens où nous l'entendons, mais ne sont pourtant pas du bruit. Cette méthode robuste d'appariement a été validée sur l'ensemble des images dont nous disposons.

Un autre résultat intéressant concerne le nombre de points détectés tout au long d'une séquence. Par exemple, la figure 3.6, correspondant à un morceau de la séquence « Amphores », montre l'évolution du nombre de points de Harris détectés pour chaque image de la séquence, avec les mêmes paramètres pour les trois niveaux de la pyramide. On constate que statistiquement le nombre de points détectés dans les niveaux supérieurs de la pyramide reste constant ce qui s'explique par l'homogénéité des images de la séquence « Amphores » : la décroissance constatée au niveau 0 correspondant à la définition maximale, s'explique quant à elle par la variation d'éclairement tout au long de la séquence qui aura tendance à illuminer les points ayant une faible réponse au détecteur et qui peuvent être considérés comme peu fiables.

3.2.2.2 Choix des points dans un voisinage

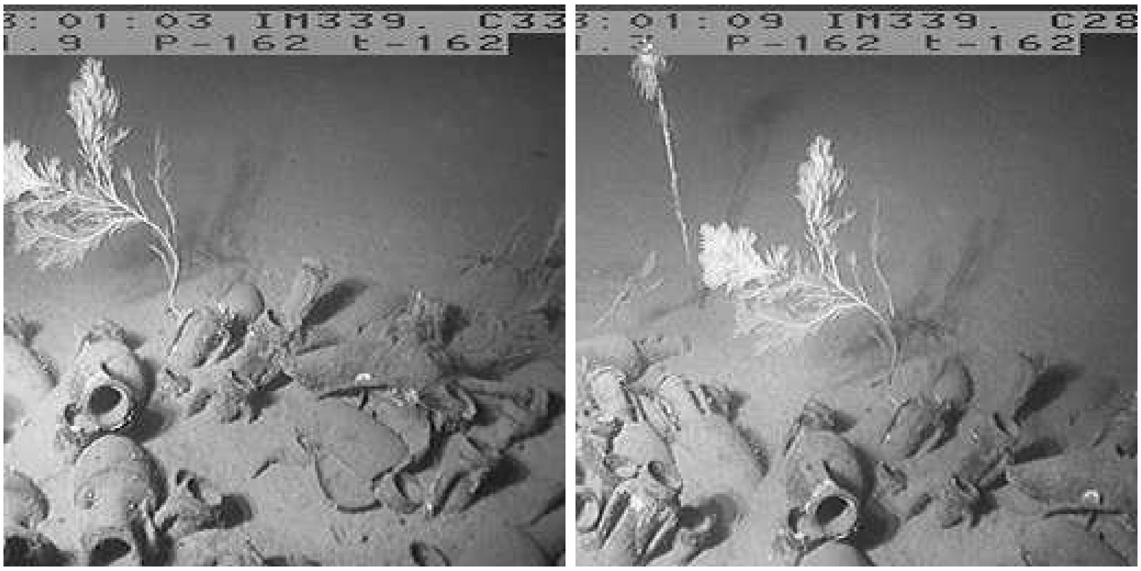
La seconde étape est le choix des points lorsqu'il y a plusieurs possibilités. En effet, il est fort probable qu'un point possède plusieurs correspondants dans un niveau supérieur. Nous devons donc choisir le meilleur candidat. Il y a plusieurs façons d'affecter aux points des critères de qualité, certains étant complexes et coûteux en temps de calcul, commes les vecteurs d'invariants introduits précédemment et d'autre plus simples, comme la valeur de Harris (rappel : $DET - \lambda Tr^2$). Nous avons opté pour la seconde solution pour deux raisons principales :

- la valeur de Harris a déjà été calculée lors de l'étape de détection des points d'intérêts,
- il est à la fois facile et rapide d'effectuer un classement d'un ensemble de points à partir d'un critère.

Ce choix comme critère de sélection de points lors de l'appariement pyramidal a montré de bons résultats et est suffisamment discrimant pour choisir le meilleur correspondant. De plus, comme dans un voisinage les points sont classés par leur valeur de Harris, nous avons par ailleurs une information sur la qualité relative des autres points de ce voisinage.

3.2.3 Contraintes de localités

Une fois que nous avons construit nos arbres d'appariement pyramidal, avec le classement par ordre décroissant d'intérêt des points de Harris dans ces différents arbres, nous devons effectuer l'appariement entre deux ou plusieurs images.



(a) Première image de la séquence (b) Dernière image de la séquence

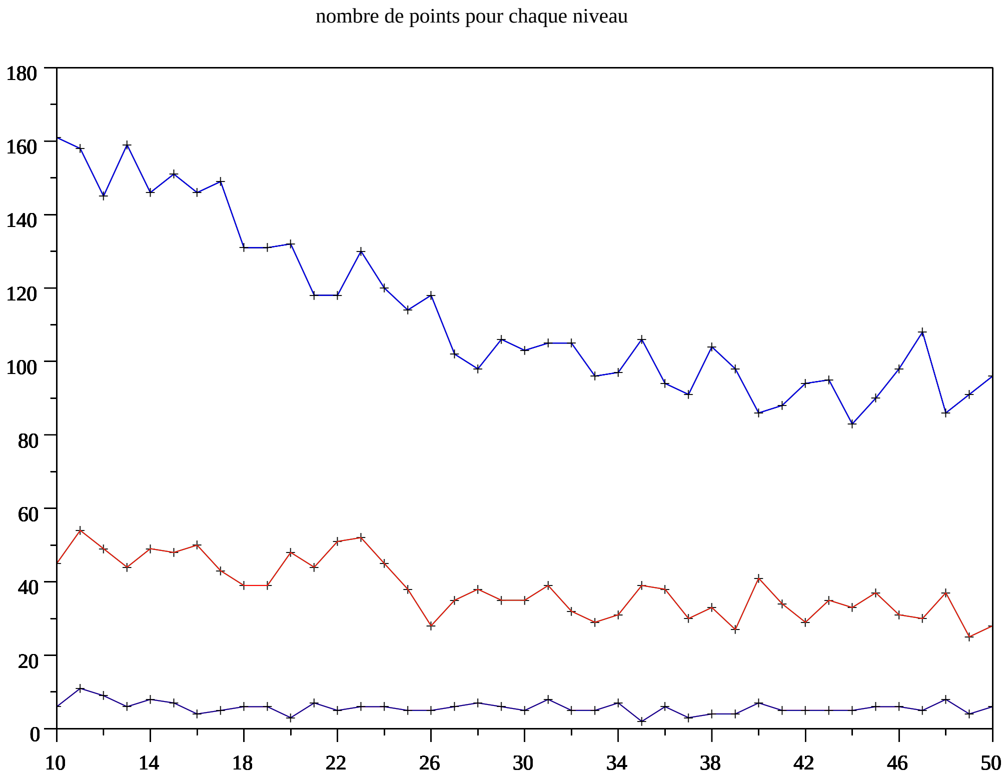


FIG. 3.6 – Nombre de points de Harris détectés sur une séquence

L'idée est là encore d'exploiter au maximum la pyramide d'images et les points robustes. D'une part, nous avons intérêt à apparier d'abord les points robustes, puis les points détectés au niveau inférieur jusqu'au niveau final, si l'on désire un maximum de points, avec le risque d'apparier des points liés au bruit. D'autre part, nous souhaitons aussi utiliser la pyramide afin de limiter les zones de recherche des points à apparier.

Pour satisfaire ce problème d'appariement, nous utilisons une contrainte de localité et une contrainte liée à la robustesse des points. La figure 3.7 illustre le principe de propagation des appariements à travers la pyramide et la prise en compte de contraintes de localité.

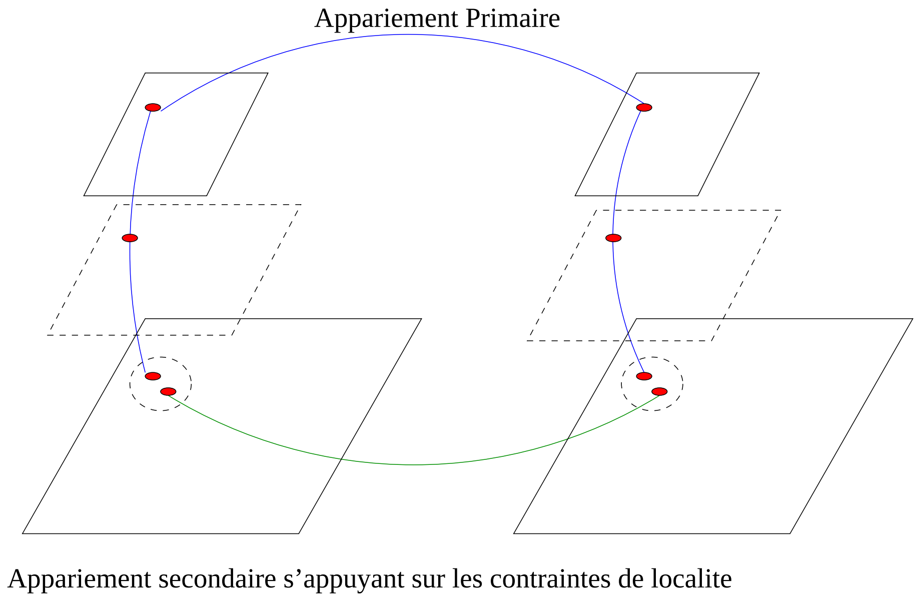


FIG. 3.7 – Contrainte de propagation et de localité pour l'appariement entre deux ou plusieurs images

Comme nous venons de le dire, l'idée est de s'appuyer sur les points robustes pour effectuer un premier appariement. Pour cela, nous apparions les points détectés au plus haut niveau de la pyramide directement dans ce niveau. D'une part, nous avons beaucoup moins de points à manipuler (ce qui peut permettre dans certains cas une recherche exhaustive des meilleurs appariements) et d'autre part, nous pouvons apparier des points très éloignés dans l'image initiale de résolution maximale. Par exemple, si deux points p_1 et p_2 sont distants de 40 pixels dans les images de niveau 0 et de taille 512x512 pixels, alors au niveau 3 de la pyramide (taille 64x64 pixels), ils ne seront éloignés que de 5 pixels.

La seconde étape est d'utiliser l'appariement pyramidal, étape déjà effectuée, pour re-projeter les points robustes et leurs correspondants dans les autres images, vers les niveaux 0 des pyramides. Grâce aux arbres d'appariements, on a alors les points robustes qui sont correctement appariés dans les images initiales. Il reste alors à apparier les autres points.

Pour cela, il suffit de se focaliser sur les points contenus dans un voisinage proche des points robustes. Ainsi on réduit l'espace de recherche dans l'image, le temps de calcul et les erreurs potentielles.

3.3 Résultats

Nous présentons dans les figures 3.8 et 3.9 les résultats d'un suivi de points effectués selon deux méthodes :

1. extraction de points d'intérêt avec le détecteur de Harris, puis corrélation entre les images
2. extraction de points robustes via la pyramide d'images, puis appariement via le critère de localité

Dans le premier cas, on remarque que la méthode classique donne de bons résultats jusqu'à ce qu'un changement de luminosité mette en échec l'algorithme de corrélation. Notons que nous avons testé différentes mesures de corrélation et que les résultats sont de même nature. En revanche, le nombre de points suivis est important.

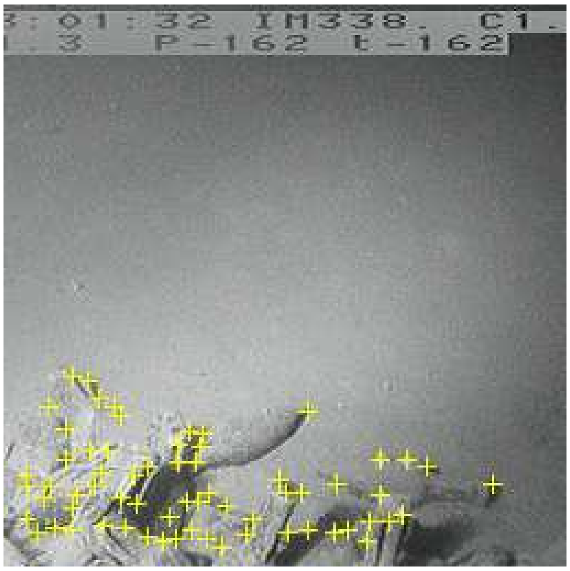
La seconde méthode, développée dans cette thèse, ne suit que les points robustes, donc moins de points que dans l'approche classique. En contre-partie, nous les avons suivis tout au long de la séquence, malgré le changement de luminosité important.

3.4 Vers une intégration des contraintes de rigidité

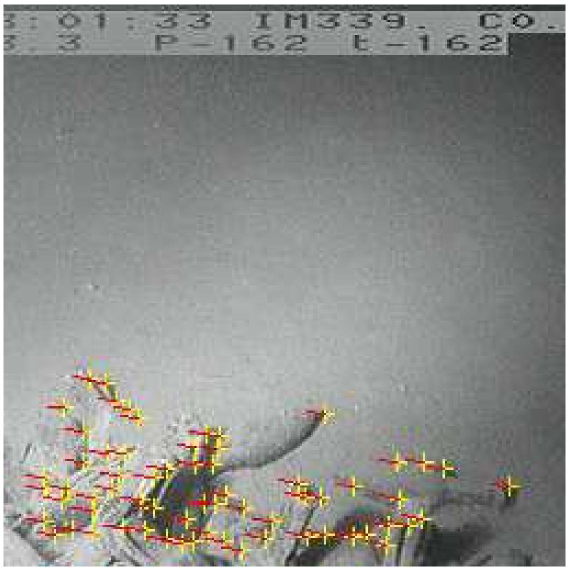
Nous l'avons déjà dit auparavant, l'une des seules informations dont nous disposions sur les scènes que nous observons est le fait que les objets à reconstruire sont rigides. Nous souhaitons donc tout naturellement utiliser cette information pour renforcer la qualité de l'appariement par exemple. En effet, les points d'intérêt, même s'ils sont robustes ne représentent qu'une information très locale du signal image. Nous souhaiterions donc intégrer des contraintes qui prennent en compte la topologie globale ou partielle de l'objet observé.

Pour cela, nous nous sommes donc intéressés aux représentations à base de graphe. Dans la mesure où nous avons extrait des points d'intérêt, il suffit de les relier entre eux pour former un graphe. Plusieurs problèmes surviennent alors. Le plus important est celui de la pondération des arcs : quel(s) type(s) d'information(s) doit-on attacher à un arc ? Ensuite vient le problème du type de graphe : complet, Delaunay, etc ?

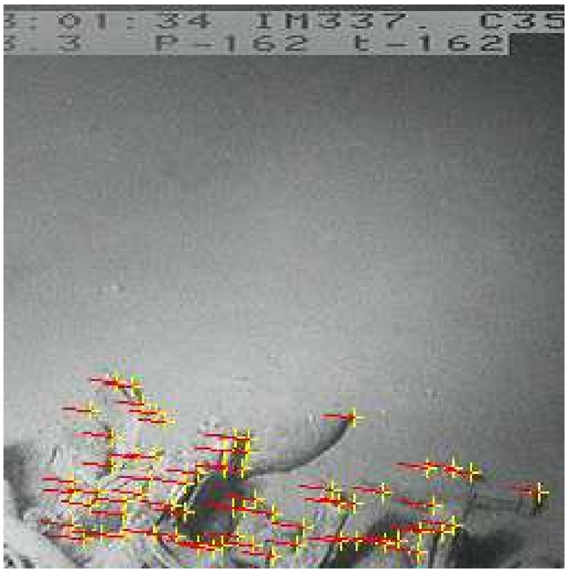
Nous avons alors testé une représentation en graphe de Delaunay. Ce choix n'est pas arbitraire car cette représentation est en fait une triangulation optimale des points dans l'image, au sens de Delaunay : chaque triangle ne peut pas contenir de points. Les graphes



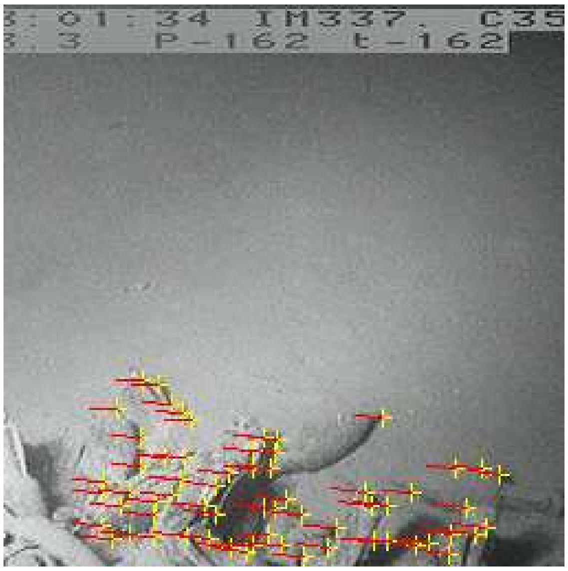
(a) Image 1



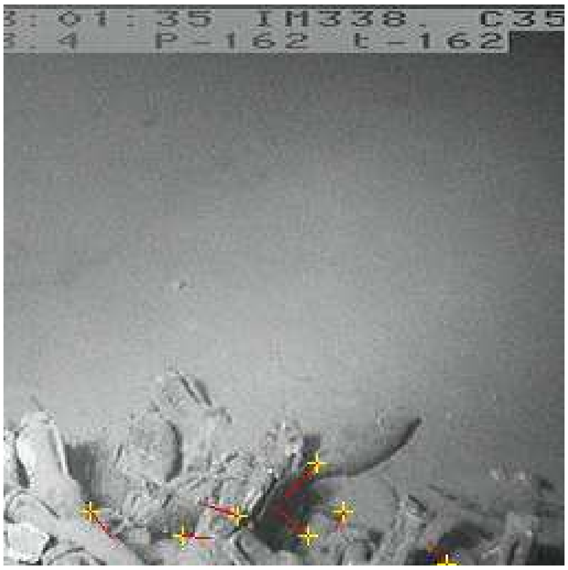
(b) Image 2



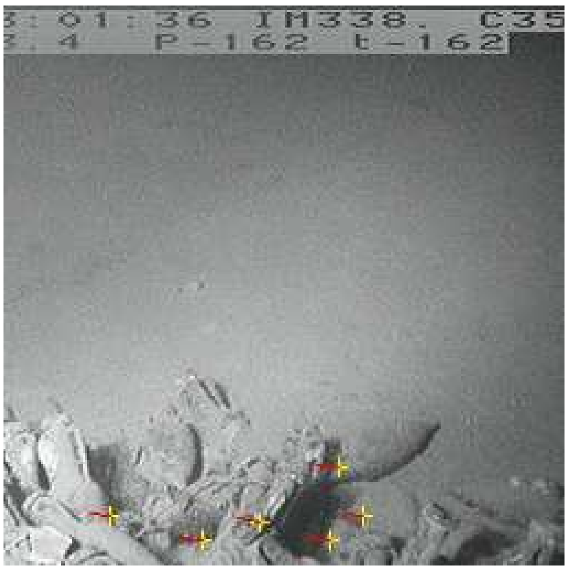
(c) Image 3



(d) Image 4

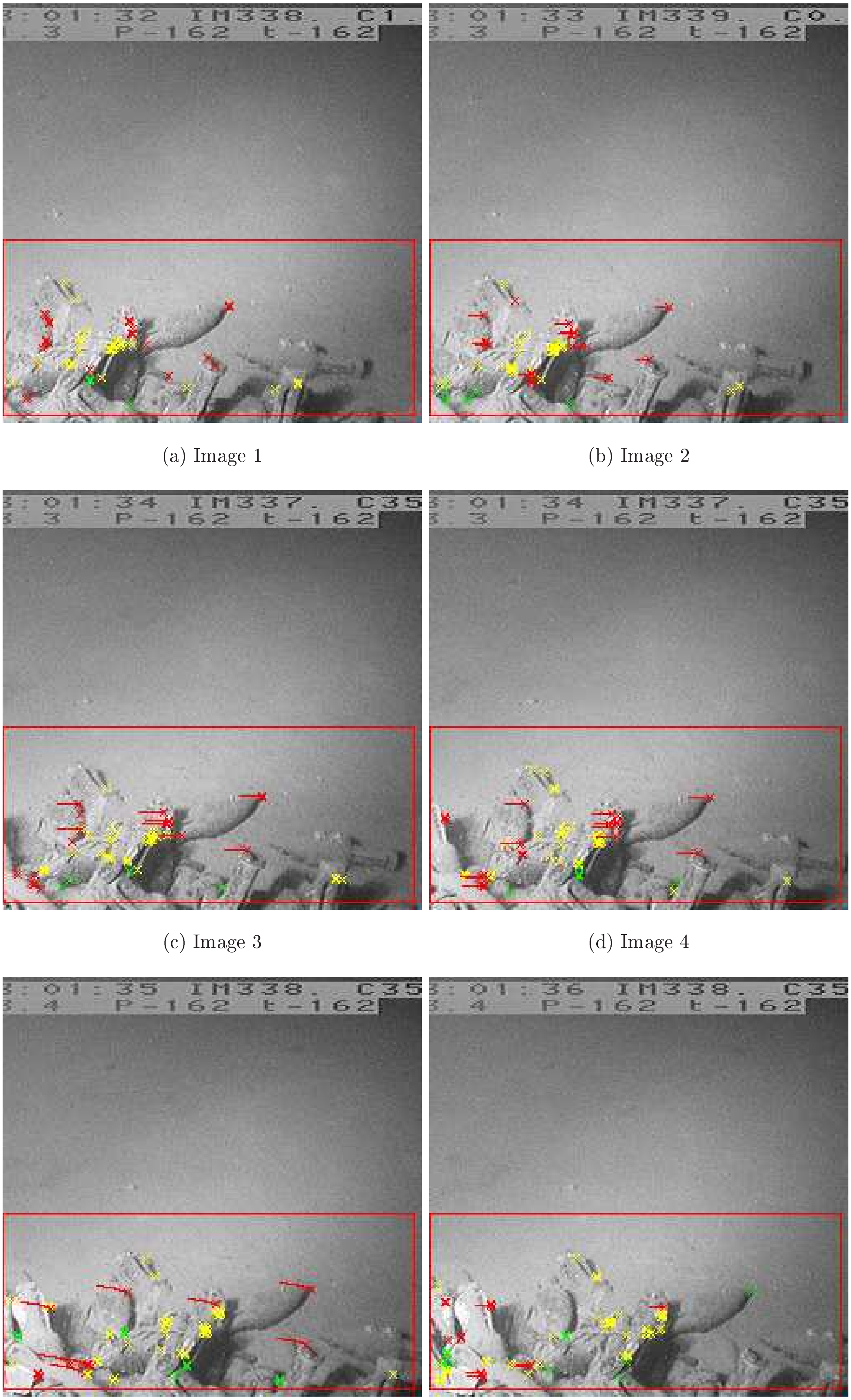


(e) Image 5



(f) Image 6

FIG. 3.8 – Suivi de points standard



(a) Image 1

(b) Image 2

(c) Image 3

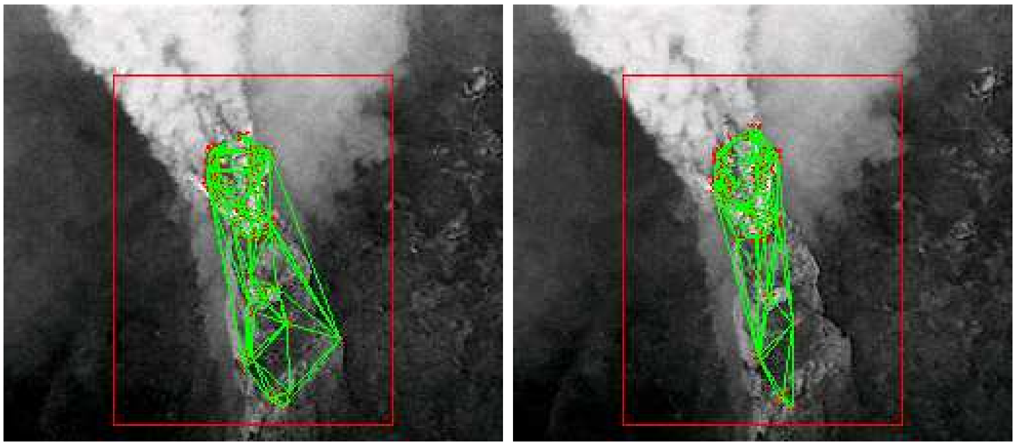
(d) Image 4

(e) Image 5

(f) Image 6

Fig. 3.9 – Suivi de points en utilisant la pyramide d'images

de Delaunay ont de bonnes propriétés, comme par exemple le fait que si l'on ajoute ou que l'on enlève un point, la modification du graphe reste locale.



(a) Image 1 (b) Image 5

FIG. 3.10 – Exemples de triangulations de Delaunay sur des images de la séquence « Fumerolle »

La figure 3.10 montre le résultat d'une triangulation de Delaunay sur deux images de la Fumerolle. Il est clair que certaines parties de la triangulation conservent la topologie et donc la rigidité du rocher. Malheureusement, cette représentation, bien que très attractive pour des travaux futurs semble très difficile à maîtriser et à exploiter. En effet, on le constate, il suffit de perdre un point important (par exemple, un point de l'enveloppe convexe du maillage), pour que celui-ci change suffisamment d'aspect. L'idée d'apparier les arcs des deux maillages pour chaque image n'est pas nouvelle (par exemple, il suffit de lire les travaux de thèse de N. Ayache (Ayache, 1983)), mais reste difficile dans une optique temps-réel. De plus, certains points étant très proches les uns des autres, il n'est pas évident que le maillage obtenu soit stable au cours du temps.

Nous pensons qu'il est alors plus sage d'utiliser des contraintes semi-globales. En effet, utiliser la totalité du graphe n'est pas le plus adapté. En revanche, il semble judicieux, à partir d'un maillage, de s'appuyer sur les plus proches voisins pour l'appariement. Plus précisemment, supposons que l'on ait apparié un point robuste p_1 de l'image I_1 avec un autre point robuste p_2 de l'image I_2 par le processus décrit auparavant. Pour apparier les autres points, on peut alors utiliser les plus proches voisins de p_1 et les apparier avec ceux de p_2. Il y a plusieurs avantages à cela. Tout d'abord cela ne nous restreint pas à un cercle autour de p_1 et il n'est plus nécessaire de fixer un diamètre, ensuite on peut effectuer des calculs entre les arcs comme par exemple, vérifier que des angles sont bien conservés dans l'image ou utiliser des birapports invariants, en s'appuyant par exemple sur les travaux de thèse de

G. Csurka (Csurka, 1996) . Notons toutefois que ces contraintes sont liées au mouvement de la caméra.

En définitive, nous pensons que l'intégration de contraintes semi-globales par une représentation sous forme de graphe, permettrait de « capturer » la topologie et la rigidité de l'objet observé. Il s'agit là d'une voie de recherches prometteuse qui sera exploitée lors de travaux ultérieurs. Toutefois, nous sommes bien conscients que les temps de calcul augmenteront et que la manipulation de graphe n'est pas toujours très simple.

3.5 Conclusion

Dans ce chapitre, nous avons présenté un ensemble de méthodes d'appariement dans le cas d'images naturelles. Si cette thématique de recherche a été largement étudiée par la communauté scientifique, elle n'est pas encore totalement résolue, comme l'ont montré les exemples sur les images naturelles.

Nous avons donc opté pour une utilisation de la pyramide d'images, qui a déjà permis de robustifier et de classifier les points d'intérêts lors de l'étape de détection de caractéristiques robutes. À nouveau, nous pouvons exploiter cette pyramide pour effectuer l'appariement sous forme de « strates ». Les résultats obtenus sont de bonne qualité et fonctionnent bien là où d'autres méthodes classiques échouent. En revanche, nous n'utilisons que des informations locales, ce qui peut s'avérer suffisant dans quelques cas, mais nous ne prenons pas en compte l'information de rigidité dont nous diposons. Nous avons étudié les triangulations de Delaunay, mais nous pensons que celles-ci sont mal adaptées au problème de l'appariemment. Nous préconisons donc une représentation de contraintes semi-globales qui prendrait en compte la rigidité entre des points voisins et non plus l'objet dans sa totalité.

Chapitre 4

Reconstruction 3D Projective

A la suite de l'étape d'appariement vient l'étape délicate de la reconstruction euclidienne ou projective dans notre cas. On le sait, c'est un problème algorithmique difficile et mal conditionné. De nombreuses méthodes existent mais toutes restent dépendantes d'une part de la qualité de l'appariement et d'autre part de la répartition des points dans la scène. Nous allons montrer ce que l'on peut espérer faire avec nos images dégradées.

4.1 Introduction

La reconstruction d'une scène tridimensionnelle, qu'elle soit euclidienne ou projective, reste délicate à effectuer en pratique. Si la théorie nous permet de poser clairement les équations de reconstruction, la stabilité des calculs reste très faible, et il n'est pas rare d'obtenir des résultats completement erronés. Avant de nous intéresser aux problèmes d'implémentations, nous présentons ici le modèle de caméra que nous utilisons et nous rappelons quelques équations classiques de la géométrie épipolaire pour la reconstruction projective. Nous invitons le lecteur à se référer à trois ouvrages de références pour la théorie générale (Faugeras, 1993) (Faugeras et al., 2001) et (Hartley et Zisserman, 2000).

4.2 Rappels

4.2.1 Modèle de caméra et type de projection

De manière assez classique, nous supposerons que notre caméra peut être approximée par un modèle de caméra sténopé. Celui-ci est bien connu et montre des résultats tout à fait satisfaisants dans un grand nombre de cas. La figure 4.1 rappelle le principe de ce modèle. De plus, comme nous ne connaissons pas la scène observée, il nous faut prendre un modèle de caméra qui soit le plus générique possible. Le lecteur trouvera une caractérisation de différents modèles de caméras dans la thèse de D. Lingrand (Lingrand, 1999).

Soit les repères suivants :

- repère absolu ou repère monde ou encore repère global ; tous les points 3D sont reconstruits dans celui-ci
- R_c : repère caméra ; repère 3D local attaché à la caméra
- R_i : repère image ; repère 2D du plan image
- R_p : repère pixels ; second repère 2D du plan image attaché à la grille des pixels

Par convention, le repère caméra aura le centre de projection pour origine, l'axe optique étant l'axe des Z et le plan image se trouvant à $Z = f$, où f est la distance focale de la caméra. On définit le repère image normalisé comme ayant son origine en $Z = 1$ et ses axes parallèles aux axes X et Y du repère caméra. Pour le repère pixels, l'origine est placée en haut à gauche de l'image. D'une façon générale, on considèrera que le repère pixel est lié au repère image normalisé par une transformation affine A prenant en compte les caractéristiques et les défauts de construction de la caméra, à savoir, non orthogonalité du plan image par rapport à l'axe optique, dimension réelle des pixels de la matrics CCD, centrage de la matrice CCD sur l'axe optique.

De fait, on peut écrire la projection perspective P (i.e. $m = P.M$) comme une combinaison

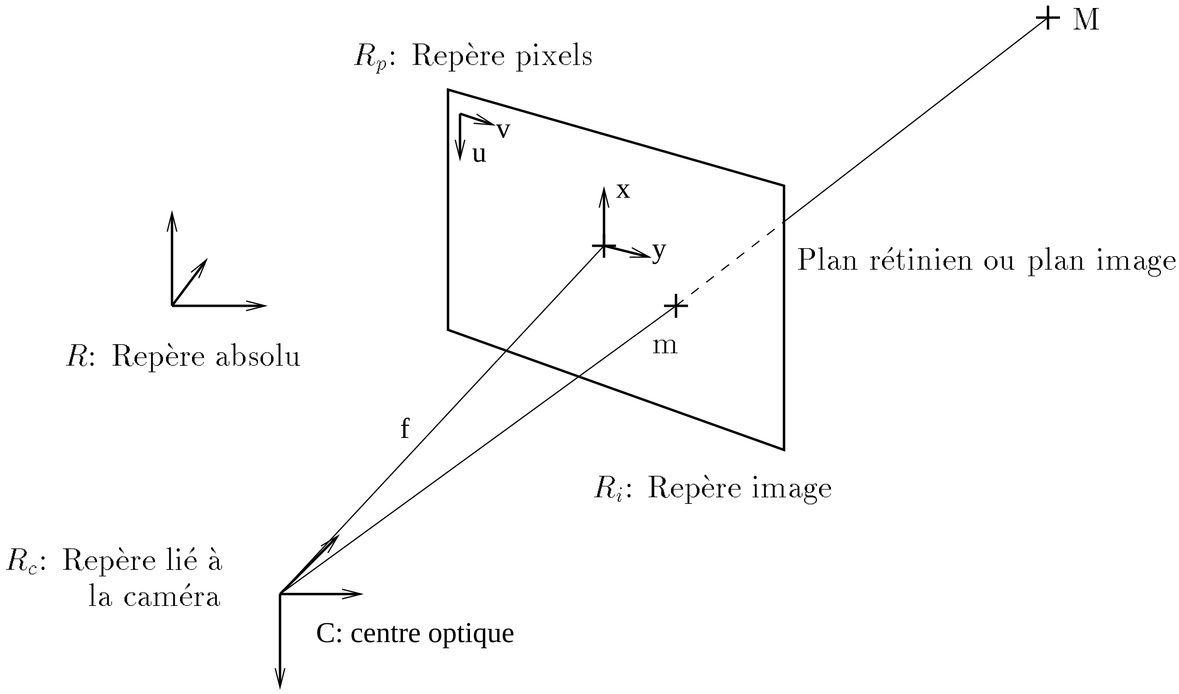


FIG. 4.1 – Modèle de caméra sténopé

de transformations entre les différents repères :

$$P \backsim AP^pT$$

$\star$ T est la transformation rigide entre le repère absolu et le repère caméra et s'écrit :

$$T_{4X4} = \begin{pmatrix} R_{3X3} & -Rt_3 \\ 0_3^T & 1 \end{pmatrix} \tag{4.1}$$

où R est une matrice (orthogonale) de rotation indiquant l'orientation de la caméra par rapport au repère absolu et t sa position

$\star$ P^p est la projection perspective. un point $M \backsim (X, Y, Z, 1)^T$ de R_c se projette en un point $m \backsim (x, y, 1)^T$ où : ($x = f\frac{X}{Z}$, $y = f\frac{Y}{Z}$). La matrice de projection peut donc s'écrire avec les coordonnées homogènes :

$$P^P_{3X4} \backsim \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{4.2}$$

$\star$ A est la transformation affine qui relie les repères image R_i et pixels R_p. Son expression générale est la suivante :

$$A \backsim \begin{pmatrix} k_u & -k_u cot\theta & u_0 \\ 0 & \frac{k_v}{sin\theta} & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{4.3}$$

où (k_u, k_v) est la dimension des pixels (longueur inverse des cotés), θ l'angle entre les axes des pixels et (u_0, v_0) la projection du centre optique dans le repère pixels.

Remarque : Les valeurs R et t sont appelées *paramètres extrinsèques* de la caméra et tandis que les valeurs $(fk_u, fk_v, \theta, u_0, v_0)$ sont les *paramètres intrinsèques* de la caméra.

$\star$ Enfin on a la matrice suivante pour le produit de A et de P^p :

$$AP^p \backsim \begin{pmatrix} k_u f & -k_u f cot\theta & u_0 & 0 \\ 0 & \frac{k_v f}{sin\theta} & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} K_{3X3} & 0_3 \end{pmatrix} \tag{4.4}$$

Si on pose $\alpha_u = k_u f$ et $\alpha_v = k_v f$, on a une nouvelle matrice K que l'on appelle *matrice de paramètres intrinsèques* :

$$K = \begin{pmatrix} \alpha_u & -\alpha_u cot\theta & u_0 \\ 0 & \frac{\alpha_v}{sin\theta} & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{4.5}$$

Si $\theta = 90°$, les pixels sont rectangulaires et si l'on impose $\frac{k_u}{k_v} = 1$ alors on a la matrice K qui s'écrit simplement comme suit :

$$K = \begin{pmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{4.6}$$

Enfin, à partir d'une matrice de projection perspective P, on peut facilement extraire les paramètres intrinsèques et extrinsèques avec cette décomposition :

$$P = KR(I_3| - t)$$

Comme nous avons choisi le type de projection perspective, cela sous-entend qu'il en existe d'autres : para-perspective et ortho-perspective, affine (ou faible). Le lecteur pourra se référer à (Lingrand, 1999) et (Sturm, 1997) pour trouver de plus amples informations, notamment sur les différentes équations et formulations.

4.2.2 Vers la reconstruction 3D

Comme nous l'avons déjà dit dans les chapitres précédents, l'un des buts de la vision par ordinateur est de faire de la reconstruction géométrique tridimensionnelle de scènes 3D statiques à partir d'un ensemble d'images ou de vues de la dite scène. S'il vient tout de suite à l'esprit une reconstruction euclidienne, avec des angles, des longueurs, en fait bien souvent, il faudra se contenter d'une reconstruction projective ou affine.

En effet, nous venons de le voir, la scène 3D réelle se projète dans une image plane en deux dimensions, qui est créée d'une part, par les conditions d'acquisitions de la scène telles que les angles de vue ou l'éclairage et d'autre part par le capteur (dans notre cas, le capteur CCD de la caméra) et qui peut donc comporter certaines propriétés, telles que la forme des pixels ou la résolution optique, par exemple.

La première conséquence de cette projection est la perte d'une dimension d'espace (la distance à la caméra) lors de l'acquisition d'une image. On le sait depuis quelques années, la géométrie la plus à même de décrire les relations géométriques directement à partir des images brutes, sans autres informations supplémentaires, est la géométrie projective. Plus précisemment, on sait reconstruire la scène modulo une transformation projective tridimensionnelle quelconque. La démonstration de ce résultat fondamental se trouve dans (Faugeras, 1992).

L'étape de reconstruction projective ou euclidienne, nécessite d'avoir au moins deux images et des primitives issues de celles-ci mises en correspondance. Les aspects d'extraction et d'appariement de primitives ayant été développés dans les chapitres 2 et 3 de ce manuscrit, nous y renvoyons le lecteur.

Comme nous allons le détailler dans la section suivante, il existe des relations géométriques particulières entre des points appariés pour un couple d'images. En fait, un point n'est apparié avec un autre que si et seulement si, il se trouve sur une droite déterminée par son correspondant, et réciproquement. On appelle cette droite, *droite épipolaire*. En fait, cette relation entre un point et sa droite épipolaire est une relation projective. Cette relation est exprimée par une matrice 3x3 de rang 2, appelée *matrice fondamentale* dont on trouvera une étude sur ses propriétés remarquables, par exemple dans la thèse dans Q.-T, Luong (Luong, 1992). En pratique, l'estimation de cette matrice est délicate et elle dépend fortement des points et du mouvement de la caméra entre les deux images. Notons qu'il existe une extension de ces travaux dans le cas d'un triplet d'images, avec le *tenseur trifocal*, sur lequel nous reviendrons rapidement par la suite.

Cette géométrie épipolaire, représentée entre autres par la matrice fondamentale et les droites épipolaires, directement estimées à partir des images, permet d'avoir une reconstruction dite projective. De plus, si l'on connaît le plan à l'infini, on pourra alors remonter à une reconstruction affine. Enfin, si les paramètres intrinsèques de la ou des caméras sont connues, nous pourrons estimer une reconstruction euclidienne. Ce cheminement entre le monde projectif et euclidien est appelé stratification de la reconstruction.

4.2.3 Rappels sur la géométrie projective

4.2.3.1 Formulations et équations

En vision par ordinateur, la géométrie projective a pris une place importante depuis le début des années 80 (Longuet-Higgins, 1981) et a amené un nouveau formalisme à cette discipline (Faugeras, 1993). Nous rappelons brièvement les équations principales et les caractéristiques de l'espace projectif.

Pour simplifier, on peut rappeler que l'espace projectif, par rapport à l'espace métrique, ne conserve pas les angles et les distances ; néanmoins, la coplanarité et la topoplogie des objets le sont. Une des caractéristiques principales de la géométrie projective est la prise en compte des points situés à l'infini. On note en coordonnées homogènes un point M ainsi lorsqu'il n'est pas à l'infini $M = (X, Y, Z, 1)^T$ et de cette façon s'il l'est : $M = (X, Y, Z, 0)^T$. Pour une introduction générale à la géométrie projective et ses équations le lecteur pourra, entre autres, se reporter au chapitre 2 de (Faugeras, 1993).

Dans la mesure où nous ne connaissons pas les paramètres intrinsèques de la caméra, nous cherchons à avoir une reconstruction 3D projective de la scène observée, laissant de côté pour le moment les méthodes d'autocalibration permettant d'obtenir un modèle 3D euclidien. Pour obtenir ce premier modèle 3D (i.e. en projectif) il faut plusieurs vues de la scène. O. Faugeras a montré qu'un modèle 3D projectif pouvait être construit à partir de deux images en s'appuyant sur la géométrie épipolaire (Faugeras, 1992).

Si on considère deux images I_1 et I_2 ainsi que les points q_1 et q_2, projections du point 3D Q dans I_1 et I_2 respectivement, alors il existe une matrice F_{21} de rang 2 qui satisfait la relation :

$$q_2^T.F_{12}.q_1 = 0 \tag{4.7}$$

où F est appelée la matrice fondamentale. On a de même la relation $q_1^T.F_{21}.q_2 = q_1^T.F_{12}^T.q_2 = 0$. L'interprétation géométrique de cette équation est liée à la géométrie épipolaire dont on rappelle le principe sur la figure 4.2.

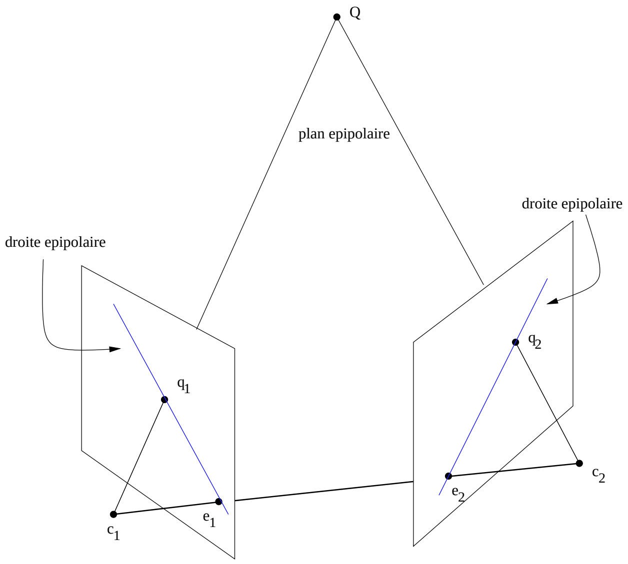


FIG. 4.2 – Principe de la géométrie épipolaire

Remarque : Si l'on dispose des paramètres intrinsèques des caméras, et si K_1 et K_2 sont les matrices des paramètres intrinsèques associées aux caméras 1 et 2 on a alors :

$$E = K_2^T.F.K_1 \tag{4.8}$$

où E est appelée la matrice essentielle et représente alors la géométrie épipolaire calibrée.

Pour rappel, nous donnons ici quelques formulations générales de F selon le modèle de caméras et leurs mouvements respectifs. Ces résultats proviennent de la thèse de P. Sturm (Sturm, 1997).

Par exemple, dans le cas d'une translation pure entre deux image, la forme générique **F** est :

$$\mathbf{F} = \begin{pmatrix} 0 & \text{-c} & \text{b} \\ \text{c} & 0 & \text{-a} \\ \text{-b} & \text{a} & 0 \end{pmatrix}$$

Dans le cas où l'on utilise des caméras affines, on a :

$$\mathbf{F} = \begin{pmatrix} 0 & 0 & \text{a} \\ 0 & 0 & \text{b} \\ \text{c} & \text{d} & \text{e} \end{pmatrix}$$

Notons qu'en pratique, l'estimation de F est difficile et la résolution numérique est très instable. De nombreux auteurs ont proposé des méthodes d'estimation de F. L'une des plus connue est la méthode des huit points de H.C. Longuet-Higgins (Longuet-Higgins, 1981) et améliorée par R. Hartley (Hartley, 1995), qui a proposé une normalisation des points avant l'estimation de F . L'avantage principal de cette méthode réside dans le fait que le calcul de F se fait de façon linéaire et huit points suffisent. Il a également montré l'importance de la normalisation des données, avant la résolution du système linéaire.

Z. Zhang *et al* (Zhang et al., 1994) (Zhang et al., 1995) a étudié et proposé une solution générale robuste au problème d'estimation de la matrice fondamentale à partir d'un ensemble de points appariés. De ces travaux, il est possible de tirer un certain nombre de conclusions. Tout d'abord, en pratique, même si la méthode de Z. Zhang est robuste, il n'est cependant pas toujours possible de l'appliquer. En effet, plusieurs facteurs pour le calcul de $\mathbf{F}$ rentrent en ligne de compte, comme le temps de calcul, le nombre de points, l'erreur souhaitée, etc. De plus, si on ne connaît que les appariements dans les images, sans aucune autre information, on peut a priori utiliser toutes les méthodes existantes, linéaires ou non, même s'il est vrai que les méthodes non linéaires sont plus robustes que celles linéaires. Notons tout de même que les temps de calcul diffèrent sensiblement. Il est donc difficile de trouver la méthode universelle qui fonctionnera dans tous les cas de figure.

Dans notre implémentation, nous avons opté pour la méthode des huit points couplée avec une approche de type RANSAC. En effet, la phase d'appariement de points d'intérêt n'est pas parfaite et il est très rare de n'avoir que des bons appariements. Il faut donc intégrer dans nos algorithmes le fait que certains des appariements que l'on a obtenus à l'étape précédente, sont faux. Traditionnellement, on utilise des méthodes robustes comme les M-estimateurs, les Moindres Carrés Médians (Meer et al., 1991) (Rousseeuw et Leroy, 1987) ou RANSAC (Fischler et Bolles, 1981). Ces méthodes permettent de classifier les données appariées en bons ou mauvais appariements,(*inliers* et *outliers*). L'intérêt majeur de RANSAC réside dans le fait qu'il est possible de fixer à l'avance le taux potentiel d'*outliers*. Si celui-ci s'avère être en-dessous de celui fixé, cela ne perturbe pas l'algorithme pour autant. En contre-partie, il faudra déterminer le seuil de rejet.

4.2.3.2 Reconstruction euclidienne

Pour passer d'un modèle projectif à un modèle euclidien, il existe plusieurs techniques. F. Devernay (Devernay et Faugeras, 1995) utilise une paire de caméras pour ce faire. Ce passage du projectif à l'euclidien nécessite une calibration des ou de la caméra. Dans le cas de deux caméras rigidement liées, le problème est relativement bien posé et de nombreuses techniques existent pour évaluer les paramètres intrinsèques de la caméra. On trouvera dans le chapitre 1 de la thèse d'E. Malis (Malis, 1998) une étude intéressante sur le passage de la reconstruction projective à la reconstruction euclidienne.

De nombreuses techniques existent alors pour faire de l'auto-calibration avec une paire stéréo ou bien une seule caméra. On peut envisager par exemple d'utiliser des plans pour calculer les paramètres de la caméra. Dans sa thèse, D. Lingrand étudie l'auto-calibration en fonction des mouvements de la caméra (Lingrand, 1999). Egalement, P. Sturm a étudié les mouvements critiques pour l'auto-calibration (Sturm, 1997). Ce domaine de recherches est très étudié encore aujourd'hui et nous renvoyons le lecteur aux références pour de plus amples détails.

4.3 Autres approches

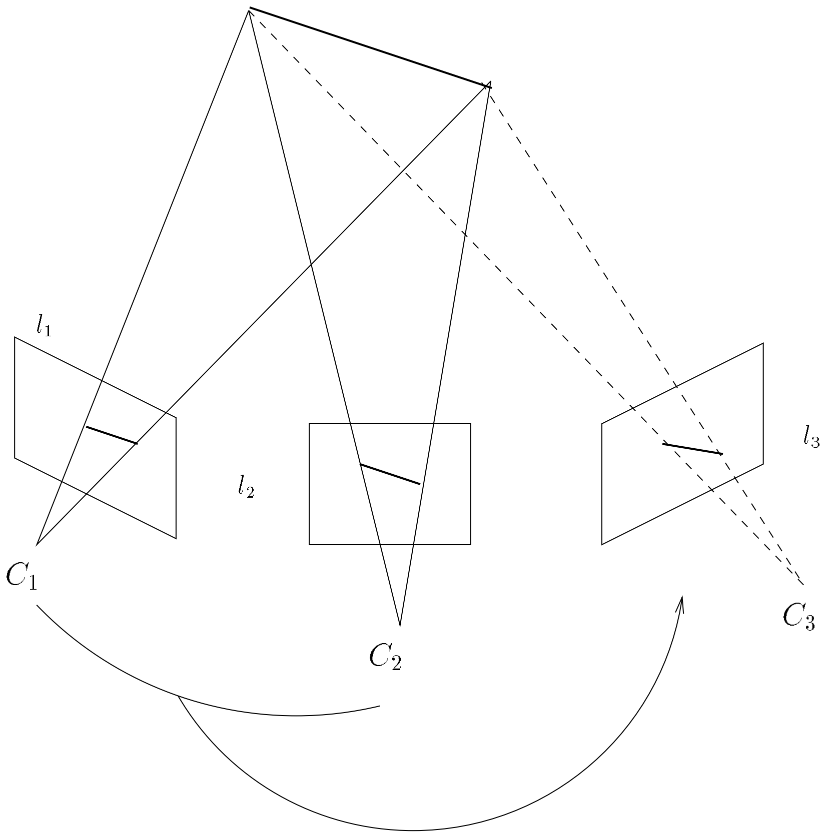
4.3.1 Tenseur trifocal

Nous l'avons évoqué un peu en amont, le tenseur trifocal peut être vu comme l'extension de la géométrie épipolaire à trois images par rapport à la matrice fondamentale. Cela sous-entend à nouveau le problème de l'appariement de caractéristiques entre trois images. Néanmoins, il a été montré que les points appariables satisfont des contraintes algébriques de degré trois (Hartley, 1997). Ces contraintes sont issues d'une application bilinéaire, qui donne à partir de deux droites dans deux images la droite dans la troisième image, qui est la projection de la même droite tridimensionnelle que celle qui a donné lieu aux deux premières. On appelle cette application le *tenseur trifocal*. La figure 4.3 en évoque le principe.

Ce n'est que récemment que l'on a établi clairement les équations des tenseurs trilinéaires (Faugeras et Papadopoulo, 1998a). Comme dans le cas de la matrice fondamentale, estimer le tenseur trifocal à partir des correspondances entre trois images est délicat à cause des fortes contraintes algébriques entre ses coefficients et son calcul robuste n'a été résolu dans sa généralité qu'il y a peu (Faugeras et Papadopoulo, 1998b). Nous invitons le lecteur à se reporter au chapitre 3 du livre de R. Hartley et A. Zisserman (Hartley et Zisserman, 2000) pour le formalisme du tenseur trifocal.

4.3.2 Ajustement de faisceaux

Plus récemment, après avoir étudié le formalisme lié à deux puis trois images, la communauté de la vision par ordinateur s'est intéressée au cas de N images. Le problème que l'on considère, toujours dans le cadre de la reconstruction est alors posé de la façon suivante. Considérons N images et m points appariés tout au long de la séquence. Alors, on cherche à estimer directement les matrices de projections et les coordonnées des points 3D. On cherche



$\tau(l_1, l_2) = l_3$

FIG. 4.3 – Principe du tenseur trifocal

à déterminer n points 3D Q_p et m matrices de projections P_i tel que :

$$\forall i = 1, \ldots, m \; \forall p = 1, \ldots, n \; P_i.Q_p \sim q_{ip} \tag{4.9}$$

Géométriquement, cela revient dans le même temps à orienter toutes les caméras et à déterminer les points 3D tels que les rayons des points images correspondants se coupent aux points 3D correspondants. On parle alors *d'ajustement de faisceaux* ou de *bundle adjustment* en anglais. On trouvera dans la thèse de P. Sturm une bonne introduction à l'ajustement de faisceaux (Sturm, 1997). Si le problème a l'air simple, ainsi posé, on imagine aisément que sa résolution pratique est délicate : pour s'en convaincre, il suffit d'imaginer le problème avec par exemple cinq images et deux cents points. On voit de suite que l'estimation des matrices de projections P_i et les coordonnées des points 3D Q_p nécessite l'utilisation de méthodes d'optimisation très robustes.

Nous renvoyons le lecteur à l'article très complet de B. Triggs *al* (Triggs et al., 1999) concernant le *bundle adjustment*. Il y est décrit de façon précise cette approche, et une étude détaillée sur les différentes implémentations y est présentée.

4.3.3 Cas dégénérés

En pratique, nous avons vu que le calcul de la matrice fondamentale était délicat dans des conditions normales. Mais à cela, il faut ajouter un cas de figure un peu particulier, celui où l'estimation de F devient instable, il faut alors estimer la matrice d'homographie $\mathbf{H}$.

En fait, les causes de l'échec de l'estimation de F sont variées et on citera par exemple les deux suivantes, en supposant que la matrice fondamentale existe :

- une mauvaise répartition des appariements dans la scène 3D, c'est-à-dire que les points sont très proches les uns des autres et sont mal distribués sur la structure 3D
- trop peu de points appariés et mal répartis sur la scène 3D, c'est-à-dire qu'il y a suffisamment peu de points pour estimer F avec des algorithmes adaptés, mais que la confiance dans ces points en terme de qualité est faible ou moyenne et cela engendrera un calcul approximatif de F

Mais il existe des cas dégénérés où la matrice fondamentale ne peut pas être estimée, comme par exemple le cas où les points appariés sont sur un plan ou encore dans le cas d'une rotation pure. Théoriquement, F n'existe pas et pourtant, du fait du bruit sur les capteurs, dans les images et de l'incertitude numérique, il va être possible d'estimer cette matrice fondamentale. Bien entendu, celle-ci sera totalement fausse car elle n'est pas censée exister ! Dans le cas du plan, il faut donc estimer la matrice d'homographie. Un autre exemple, est celui d'un très faible déplacement entre deux images successives. On le sait, l'estimation de F va être, là encore très mauvaise.

C'est donc un problème clé de la vision par ordinateur que celui du choix de l'estimation de la matrice fondamentale ou d'homographie. Une méthode couramment employée développée par P. Torr (Torr et al., 1998) estime parallèlement des modèles de H et de F. L'idée est d'utiliser les points appariés pour valider les contraintes des modèles ; si un modèle est mieux validé qu'un autre, alors, il estime la matrice correspondante. Cette méthode est assez coûteuse en temps de calcul.

4.3.4 Parallaxe virtuelle

Une autre méthode intéressante, est celle que l'on appelle la méthode de la parallaxe virtuelle. L'idée est d'estimer la matrice d'homographie en prenant 4 points sur un plan. Le problème vient du fait que rien ne nous assure que ces 4 points soient bien sur un même plan. E. Malis *et al.* améliorent cette approche en considérant seulement 3 points qui définissent un plan virtuel (Malis et al., 2000). Ainsi, on est certain que ces points sont bien sur un même plan. Cette méthode permet alors d'estimer directement la matrice d'homographie, sans avoir à estimer la matrice fondamentale. Le principe repose sur l'utilisation d'un plan et de la factorisation parallaxe. On choisit un plan virtuel π définit par 3 points appariés p_{1i} et p_{2i} ($\{i = 1, 2, 3, ..., m\}$) dans deux images I_1 et I_2. Alors les points sont liés par la relation :

$$\gamma_i p_{1i} = \mathbf{H}_{12} p_{2i} + \mu_i e_{12} \quad \{i = 1, 2, 3, ..., m\} \tag{4.10}$$

où $\mathbf{H}_{12}$ est la matrice d'homographie liant les points au plan π, e_{12} est l'épipole dans l'image I_2, γ_i et μ_i sont des scalaires. Notons que $\mu_i = 0$ si le points 3D correspondant appartient au plan π. La droite épipolaire l_{2i} correspondant au point p_{1i} dans l'image est donnée par :

$$\mathbf{l}_{2i} \propto p_{1i} \wedge \mathbf{H}_{12}\, p_{2i} \propto p_{1i} \wedge e_{12} \quad \{i = 1, 2, 3, ..., m\} \tag{4.11}$$

Si l'on considère alors la matrice 3xm $\mathbf{L}_2$ contenant toutes les droites épipolaires, on a :

$$\begin{aligned} \mathbf{L}_2(\mathbf{H}_{12}) &= \begin{bmatrix} \mathbf{l}_{21} & \mathbf{l}_{22} & ... & \mathbf{l}_{2m} \end{bmatrix} \\ &= \begin{bmatrix} p_{11} \wedge \mathbf{H}_{12} p_{21} & ... & p_{1m} \wedge \mathbf{H}_{12} p_{2m} \end{bmatrix} \end{aligned} \tag{4.12}$$

Cette matrice est telle que : $rang(\mathbf{L_2}) < 2$, c'est-à-dire que $\det(\mathbf{L}_2\mathbf{L}_2^T) = 0$ dans la mesure où toutes les droites épipolaires se coupent à l'épipole. De plus, si la scène observée est plane, on a $rang(\mathbf{L_2}) = 0$. Notons $\mathbf{x}$ un vecteur contenant les entrées de $\mathbf{H}_{12}$. On peut donc estimer la matrice d'homographie en résolvant un problème de minimisation non-linéaire comme suit :

$$\min_{\mathbf{x}} f(\mathbf{x}) = \|\mathbf{L}_2\|$$

$$\text{sous la contrainte} \tag{4.13}$$

$$\min_{\mathbf{x}} g(\mathbf{x}) = \det(\mathbf{L}_2 \mathbf{L}_2^T)$$

La fonction g est minimisée lorsque $\partial g(\mathbf{x})/\partial \mathbf{x} = 0$. Le problème (4.13) est alors équivalent au problème suivant :

$$\min_{\mathbf{x}} f(\mathbf{x}) + \lambda^T \frac{\partial g(\mathbf{x})}{\partial \mathbf{x}} \tag{4.14}$$

où λ est un vecteur contenant les multiplicateurs de Lagrange. E. Malis a utilisé cette approche pour contrôler un robot en extrayant les informations contenues dans $\mathbf{H}_{12}$ comme il l'explique dans (Malis et al., 1999).

4.4 Expérimentations et résultats

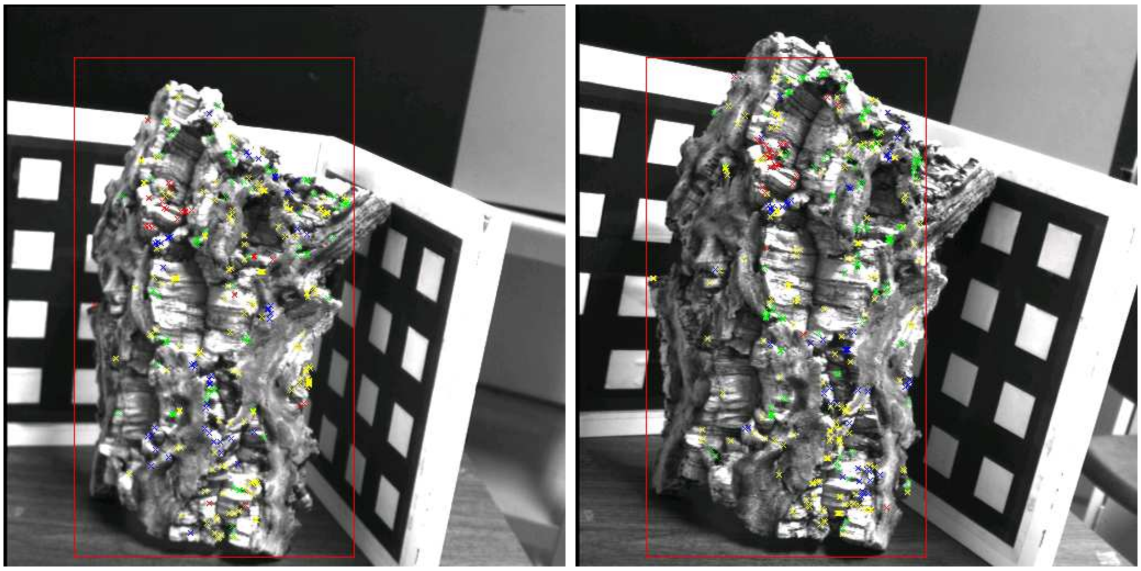
Dans la mesure où ne nous connaissons pas la vérité terrain des images fournies par l'Ifremer, il nous était impossible d'avoir des résultats quantitatifs sur les reconstructions réalisées à partir de ces images. Donc, pour estimer la qualité de nos algorithmes et évaluer notre approche, nous avons utilisé une tête stéréo calibrée. Ainsi nous avons pu estimer la matrice fondamentale en utilisant une mire de calibration entre les paires d'images.

4.4.1 Validation avec une tête stéréo calibrée

La figure 4.4 montre le résultat de l'extraction des points selon notre approche pyramidale pour un couple d'images acquises avec la tête stéréo calibrée. Ces images sont de taille 512x512 pixels et les pyramides d'images sont de 4 niveaux.

Après appariement de tous les points par une méthode de corrélation d'une part et d'autre part uniquement des points robustes, nous avons pu comparer notre estimation de la matrice fondamentale à partir de ces points avec la vraie matrice fondamentale. Les résultats sont présentés dans le tableau 4.1.

Comme prévu, le nombre de points robustes est plus faible avec l'approche multi-échelles. Si l'on observe le nombre de points appariés, on constate qu'il y en a moins au total avec l'approche multi-échelles (64 contre 85). En revanche, on peut remarquer que le taux de rejet est plus faible avec cette approche. Ce résultat est satisfaisant, car cela signifie qu'un faible nombre de points robustes seraient mieux appariés que l'ensemble total de points.



(a) Image gauche

(b) Image droite

FIG. 4.4 – Exemple d'appariement avec un objet non structuré

	Nombre de points appariés	Nombre de points extraits gauche	droite	Taux d'« outliers » gauche	droite
Sans pyramide	85	304	385	72.04%	77.92%
Avec pyramide	64	192	186	66.66%	65.59%

TAB. 4.1 – Résultats sur le liège avec une tête stéréo calibrée

Pour valider totalement ceci, nous vérifions l'erreur des projections des points appariés aux droites épipolaires réelles calculées à partir de la véritable matrice fondamentale. Le tableau 4.2 présente ces résultats.

Dans les deux cas, l'estimation de la matrice fondamentale a été effectuée soit avec un algorithme linéaire (algorithme des 8 points normalisés + RANSAC) soit avec un algorithme non linéaire (de type Moindre Carrés). Comme prévu, ce dernier donne des résultats meilleurs que dans le cas linéaire. Mais ce qui nous intéresse ici c'est le fait qu'avec la structure pyramidale, on obtienne des résultats de meilleure qualité.

L'apport de la pyramide est indéniable. L'erreur est plus faible si l'on n'utilise que les points robustes appariés pour estimer la matrice fondamentale. Egalement, les temps de calcul sont plus faibles de l'ordre de 30%.

En définitive, l'approche multi-échelles permet d'une part de sélectionner des points robustes et d'autre part de les utiliser pour estimer la matrice fondamentale avec une qualité

	Algorithme linéaire	Algorithme non-linéaire
Sans pyramide	3.13	1.72
Avec pyramide	2.61	0.97
Amélioration	16.61%	43.60 %

TAB. 4.2 – Erreurs de reprojections des points appariés par rapport aux droites épipolaires réelles

supérieure que celle obtenue en utilisant tous les points détectés.

4.4.2 Estimation du mouvement dans des images naturelles

Nous présentons ici un autre exemple concernant cette fois-ci l'estimation du mouvement dans l'image dans le cas d'images naturelles et pour un faible déplacement. Pour valider nos résultats, nous avons comparé notre méthode à une méthode robuste de suivi de points. Le but de cette expérience est de prendre deux images, d'en extraire les points robustes et de calculer la matrice d'homographie entre ces deux images. Nous n'estimons pas la matrice fondamentale, car nous savons que la structure de la scène est relativement plane, donc que le calcul de **F** n'a pas de sens. Les images proviennent de la séquence « Amphores » et sont de taille 256x256 pixels ; on a donc construit des pyramides à 3 niveaux.

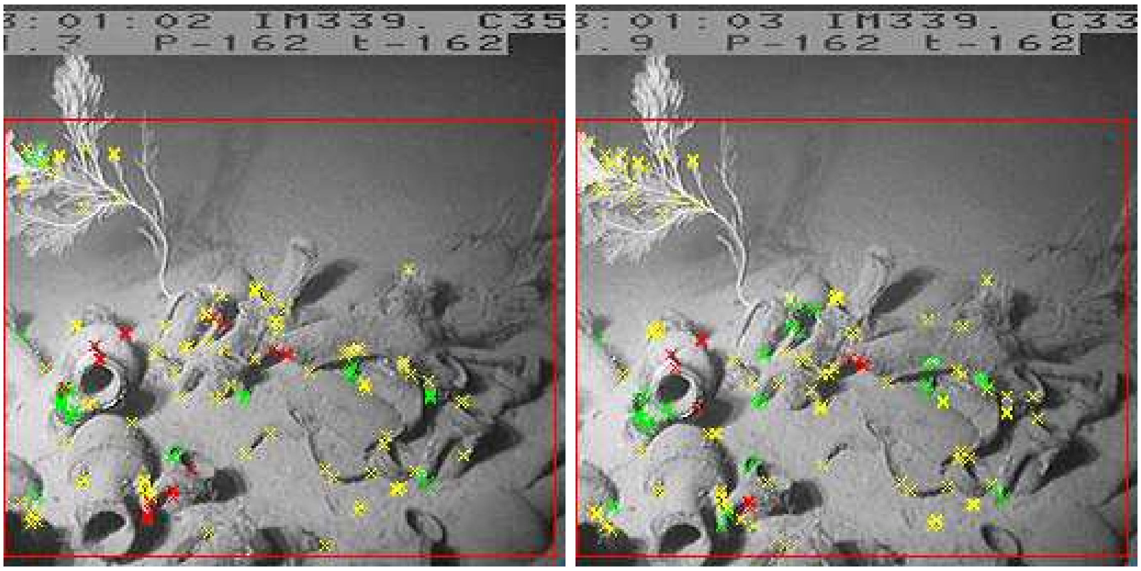
Les deux images à apparier sont celles de la figure 4.5 d'où l'on a déjà extrait et classé les points robustes, de couleur rouge. Les points de couleur verte correspondent aux points détectés au niveau intermédiaire de la pyramide, les points jaunes n'étant détectés qu'au niveau 0 de la pyramide.

La méthode robuste de suivi de points se base sur une extraction de points dans toute l'image, sans caractérisation et sans classification, puis sur une mesure de corrélation entre chaque image et pour chaque point détecté. A l'issue du traitement, nous avons donc une liste de points suivis sur les 6 images, qui serviront à estimer le déplacement dans l'image.

Nous estimons alors la matrice d'homographie entre l'image 1 et l'image 6, avec d'une part les points issus de la méthode robuste de suivi et d'autre part avec les points robustes appariés par une simple mesure de corrélation. La matrice d'homographie est calculée avec la méthode d'E. Malis précédemment décrite :

$$H = \begin{pmatrix} 1.0000 & -0.0000 & 1.0002 \\ -0.0000 & 1.0000 & 2.0010 \\ -0.0000 & 0.0000 & 1.0000 \end{pmatrix} \tag{4.15}$$

Nous avons correctement estimé le mouvement qui est une translation pure, dans cette séquence. Cette translation est également celle trouvée par le suivi de points.



(a) Image 1 de la séquence Amphores (b) Image 6 de la séquence Amphores

FIG. 4.5 – Images à apparier pour estimer le mouvement

Il y a donc un double intérêt ici à utiliser la pyramide d'images et les points robustes. D'une part, on n'utilise que deux images au lieu de six necéssaires au « tracking » pour estimer H et d'autre part, on utilise moins de points, car on sait qu'ils sont robustes.

4.5 Conclusion

Dans ce chapitre, nous avons présenté succintement des rappels de géométrie projective et le modèle de caméra que nous utilisons. L'étape de reconstruction projective ou euclidienne est toujours délicate et même si la théorie est bien posée, le passage à la pratique reste difficile. Il n'est pas rare, surtout avec le type d'images que nous manipulons, d'avoir des estimations fausses de la matrice fondamentale, par exemple dans le cas des amphores. Néanmoins, nous avons validé notre approche multi-échelles dans le cas stéréo où nous disposions de la véritable matrice fondamentale. Nous avons effectivemment montré qu'il est possible d'estimer une matrice fondamentale correcte à partir d'un faible nombre de points robustes sélectionnés via la pyramide d'images, sur des objets totalement inconnus et non structurés.

Egalement nous avons appliqué une extension de la méthode de la parallaxe virtuelle à notre problématique. Nous estimons systématiquement la matrice d'homographie et nous pouvons ainsi remonter au déplacement dans l'image. Cette méthode a été testée et a donné de bons résultats sur des images sous-marines.

Chapitre 5

Cas des images très dégradées

La vision par ordinateur : un outil de modélisation universel pour les applications sous-marines ? Hélas, quiconque a déjà mis un masque de plongée a pu constater que la visibilité sous l'eau était très réduite : d'une part, la lumière très diffuse, ne porte pas loin sous l'eau, et d'autre part, les algues ont tendance à tout recouvrir. Par conséquent, les images prises dans ce contexte posent de nombreux problèmes qui se révèleront dans certains cas insolubles. Pourtant parfois, la mise en oeuvre de traitements simples permettra d'améliorer de façon spectaculaire le résultat de nos algorithmes. Dans ce chapitre, nous présentons quelques idées et des résultats préliminaires allant dans ce sens.

5.1 Limitations de notre approche

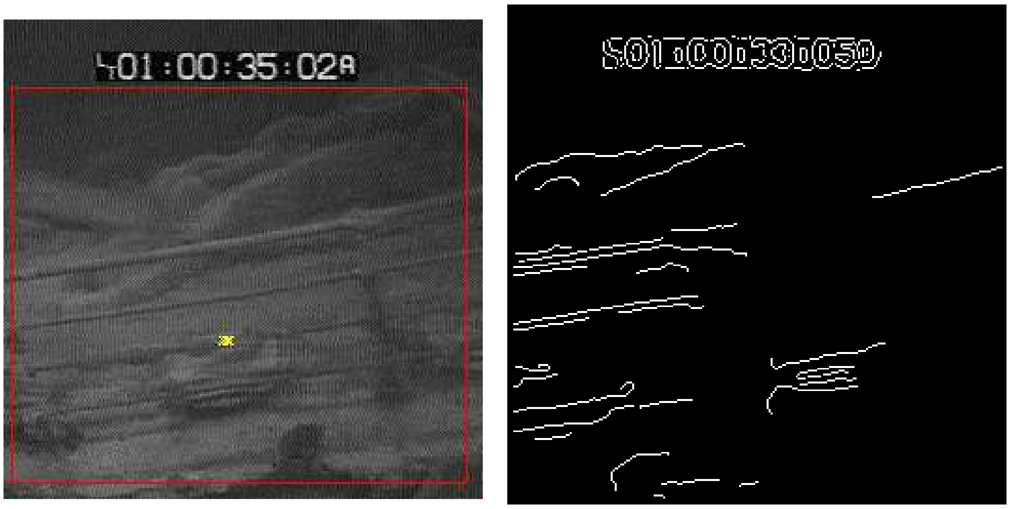
Dans le cas de nos images sous-marines, il n'est pas rare, pour des raisons évidentes liées à l'environnement marin que les images à traiter soient de piètre qualité comme l'atteste l'image de la figure 5.1 issue de la séquence du Titanic. Sur cette image, les gradients sont très faibles et il est difficile, même pour un oeil exercé, de discerner correctement les différents objets contenus dans la scène.

FIG. 5.1 – Titanic

Avec ce type d'images, les algorithmes d'extraction de contours ou de points d'intérêts sont clairement voués à l'échec comme le montrent les figures 5.2(a) et 5.2(b), même en utilisant les structures de pyramide d'images.

Le but de ce chapitre est donc double. D'abord il est de montrer, au travers d'exemples, que la théorie ne se transpose pas toujours dans la pratique, et en particulier dans les environnements sous-marins. Ensuite, nous voulons montrer qu'il est tout de même envisageable de traiter certains types d'images dégradées. En fait, sur ce type d'images, on se rend bien compte d'une part des limites des approches quelles qu'elles soient, et d'autre part du mauvais conditionnement du problème de la vision en général. Ainsi, même si la géométrie a permis de résoudre nombre de problèmes, il n'en reste pas moins que dans un grand nombre de cas, en l'occurence dès que l'on se situe dans un environnement extérieur, donc dégradé au sens de l'image, la robustesse des algorithmes est rapidement mise en défaut. Il nous faut alors utiliser certaines astuces algorithmiques.

De plus, et on le retrouve tout au long de ce travail de thèse, il est difficile de définir ce que l'on souhaite obtenir à partir de telles images. Bien évidemment, l'idéal serait d'estimer le mouvement de la caméra ou de reconstruire un modèle 3D de la scène, mais est-ce réellement possible ? Nous allons montrer dans la suite de ce chapitre, d'une part les limites des méthodes



(a) Extraction de points de Harris via notre approche multi-échelles

(b) Extraction de contours (Canny-Deriche)

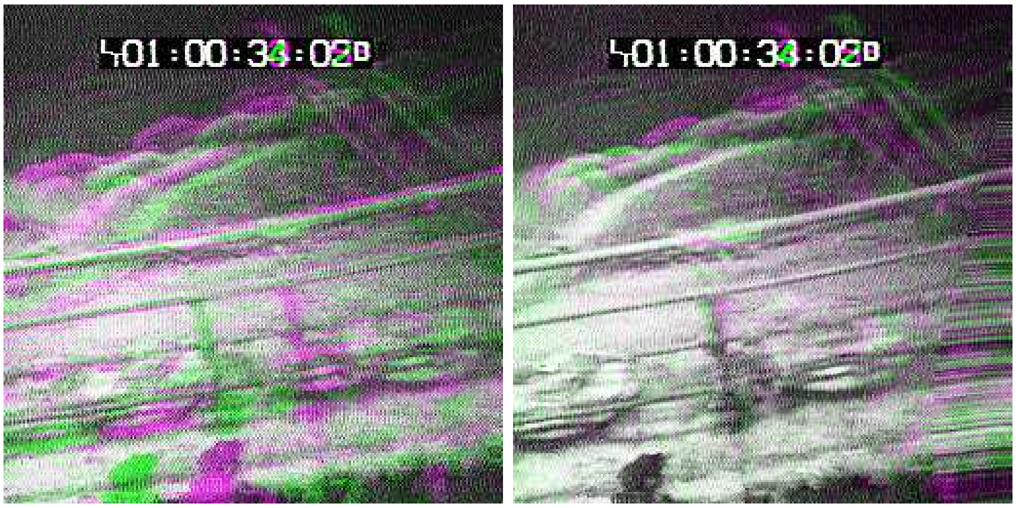
Fig. 5.2 – Exemple de résultats avec des algorithmes classiques sur une image du Titanic

classiques robustes et d'autre part, nous allons essayer de donner un élément de réponse pour pouvoir traiter tout de même ces images.

5.2 Approche Variationnelle

Afin de comparer notre approche avec une méthode robuste de référence, nous avons testé tout d'abord une méthode variationelle qui utilise une approche multi-échelle. Elle a été développée par G. Hermosillo et O. Faugeras récemment (Faugeras et Hermosillo, 2001) (Hermosillo et Faugeras, 2001). Le principe de cette approche est d'effectuer un appariement dense entre deux images en s'appuyant sur une approche variationnelle. Pour cela, les auteurs estiment des déformations semi-locales. Les résultats validés essentiellement sur des images de coupes de cerveaux sont de bonne qualité, mais les images traitées sont nettement moins dégradées que celles du Titanic. Néanmoins, cette méthode est en mesure d'estimer des déformations très grandes entre deux images. De plus, le fait d'estimer des déformations semi-locales avec des méthodes d'appariement denses permet d'estimer des déformations non uniformes, que l'on retrouve lorsque la scène a beaucoup de profondeur par exemple (le mouvement apparent des objets éloignés est moins important que celui des objets proches de la caméra).

Sur la figure 5.3(a), on a superposé les deux images à apparier. Ainsi on se rend mieux compte du déplacement dans l'image. Les couleurs vertes et rouges correspondent à chacune des images d'origine et montrent la disparité entre elles, correspondant essentiellement à un mouvement de translation.



(a) Superposition des images à apparier

(b) Résultat

Fig. 5.3 – Superposition

La figure 5.3(b) montre le résultat de cette approche variationnelle sur les images du Titanic. A partir des deux images, on estime les champs de vecteurs semi-locaux, via une pyramide d'images, et on calcule alors les disparités. Les parties grisées correspondent à un appariement correct et les parties colorées en vert au déplacement estimé, la couleur rouge correspondant au but à atteindre.

On le constate aisément, l'appariement est loin d'être parfait et finalement, peu de structures sont recouvrées. On peut remarquer que la rigidité de la scène n'est pas préservée dans l'estimation du déplacement (les parties éloignées ne sont pas mises en correspondance). Cela est dû au caractère local des déformations introduites dans le modèle. Notons aussi qu'une quinzaine de secondes sur un PC récent est nécessaire pour obtenir ce résultat. En ce qui concerne le réglage des paramètres, celui-ci se fait par un apprentissage supervisé par l'opérateur. En effet, il fournit à l'algorithme une estimation du paramètre de déformation du champs de vecteur. Néanmoins, le résultat obtenu semble prometteur dans la mesure où l'approche testée n'a pas été exploitée à son maximum car le travail théorique est loin d'être achevé pour ces méthodes variationnelles.

Le but de cette expérience sur les images du Titanic est donc de montrer que même des méthodes pourtant réputées robustes sont mises en défaut. La séquence du Titanic est certes particulière, car vraiment de mauvaise qualitée , mais est également un bon indicateur pour évaluer les limites des différentes approches.

5.3 Améliorations proposées

A la vue de ces précédents résultats, qui se soldent par des échecs, nous souhaitons tout de même montrer qu'il est possible de manipuler de telles images et d'en extraire des informations.

En fait, ce qui est tout de suite visible, c'est le faible gradient et l'uniformité des niveaux de gris dans l'image initiale. L'histogramme permet de s'en convaincre facilement (c.f. figure 5.4). En effet, celui-ci est extrêmement étroit et centré autour d'une valeur. Cette constatation nous amène tout naturellement à utiliser cette information et à générer une nouvelle image en appliquant une égalisation d'histogramme à l'image initiale. Cette technique, classique, fonctionne bien pour rehausser les images peu contrastées, mais a pour effet de renforcer le bruit dans l'image, la rendant moins exploitable. Notons bien que l'égalisation d'histogramme n'améliore pas le rapport signal/bruit.

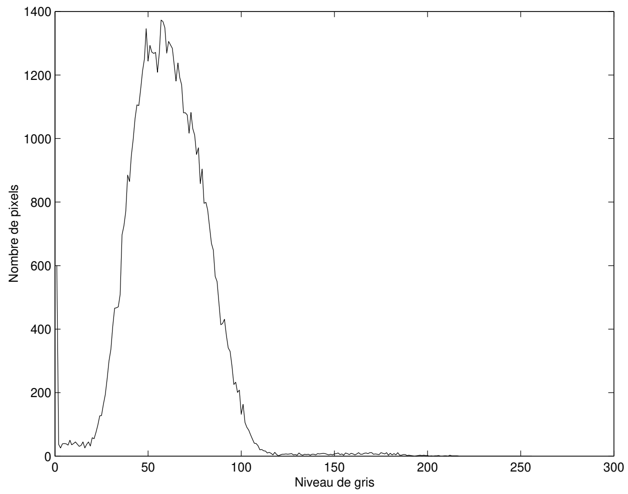


FIG. 5.4 – Histogramme de l'image titanic n^o1

5.3.1 Égalisation d'histogramme

Nous présentons ici le formalisme associé à ce processus d'égalisation, dans le cadre d'un signal discret. Soit une image I de taille NxM et contenant n niveaux de gris. Habituellement, on a $N = M$ et $n = 256$. Soit E l'ensemble discret des points de l'image. On peut donc voir l'image comme une fonction I de E vers $\{0, \ldots, n-1\}$, associant à un point p de E le niveau de gris $I(p)$.

On définit alors l'histogramme de l'image I comme la fonction H_I définie sur l'intervalle de niveaux de gris $\{0, \ldots, n-1\}$ et à valeurs entières non-négatives, associant à tout niveau de gris g le nombre de points dans E ayant le niveau de gris g dans l'image I.

$$H_I(g) = card\ \{\ p \mid I(p) = g\ \}\ =\ N \tag{5.1}$$

Par conséquent, étant donnés deux niveaux de gris a et b (avec $a < b$) de l'intervalle $\{0, \ldots, n-1\}$, le nombre de points de p de E dont le niveau de gris est compris entre a et b est donné par la somme $H_I(a) + ... + H_I(b)$.
En particulier, on a $H_I(0) + ... + H_I(n-1) = card(E) = N$.

Les défauts de contraste dans une image apparaissent donc clairement en observant l'histogramme. Pour les corriger, il faut appliquer une fonction de rehaussement des niveaux de gris, que l'on choisit en fonction de l'étalement de l'histogramme initial. Dans notre cas, cette fonction est appelée « égalisation d'histogramme ». On calcule à partir de l'histogramme H_I une fonction de rehaussement des niveaux de gris f telle que l'image rehaussée I_r, définie par $I_r(p) = f(I(p))$, ait son histogramme H_{I_r} se rapprochant le plus possible d'une fonction « plate ». En pratique, du fait que nous manipulons des ensembles discrets, il est impossible d'obtenir un histogramme réellement « plat ».

Nous définissons alors l'histogramme cumulatif de l'image I comme la fonction C_I définie sur l'intervalle de niveaux de gris $\{0, \ldots, n-1\}$ et à valeurs entières non-négatives, associant à tout niveau de gris g le nombre de points dans E ayant un niveau de gris inférieur ou égal à g dans l'image I. On a donc :

$$C_I(g) = H_I(0) + ... + H_I(g) \tag{5.2}$$

On obtient donc C_I par le schéma itératif suivant :

$$\begin{array}{ll} C_I(0) = H_I(0) & et \\ C_I(g) = C_I(g-1) + H_I(g) & pour\ g = 1, \ldots, (n-1) \end{array} \tag{5.3}$$

On remarque donc que $C_I(n-1) = card(E) = N$. On applique ensuite à l'image I la fonction de réhaussement des niveaux de gris f définie par

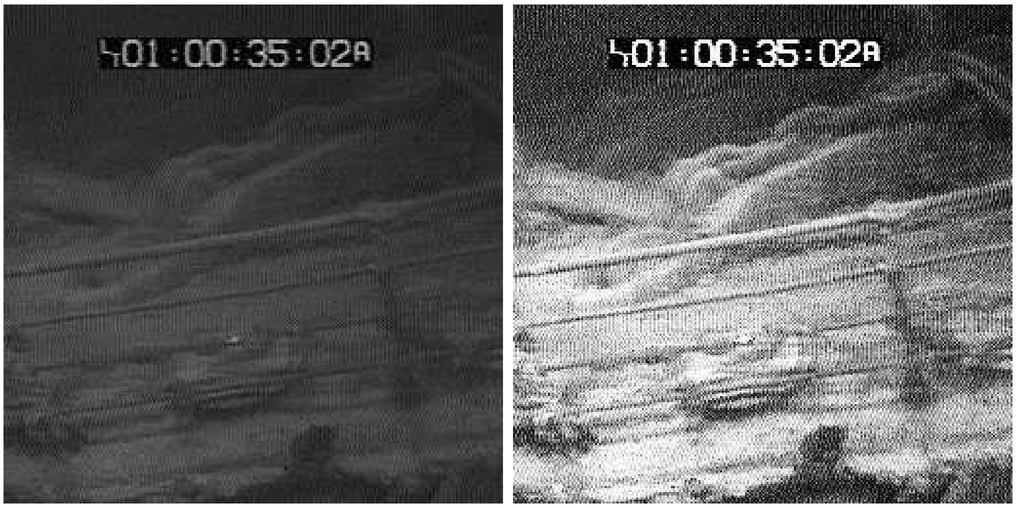
$$f(g) = (n-1)\frac{C_I(g)}{N} \tag{5.4}$$

que l'on arrondit à l'entier le plus proche.

On a ainsi :

$$\left\{ \begin{array}{ccc} f(0) & = & (n-1)\frac{H_I(0)}{N} \\ f(n-1) & = & n-1 \end{array} \right.$$

Dès lors, cet algorithme classique, facile à implémenter et très rapide (quasiment temps - réel), va nous permettre de traiter des images de faible dynamique telles que celles issues de la séquence du Titanic. Un exemple du résultat de l'égalisation d'histogramme est présenté sur les figures 5.5(a) et 5.5(b).



(a) image originale (b) image égalisée

FIG. 5.5 – Exemple d'égalisation sur une image issue de la séquence du « Titanic »

En conclusion et on l'observe très nettement sur les figures précédentes, cette égalisation fait ressortir les structures des objets contenus dans la scène, mais génère en contre-partie beaucoup de bruit. Plus précisémment, la distribution de bruit de l'image initiale sera dilatée de la même façon que le signal utile.

5.3.2 Résultats préliminaires et interprétations

Nous venons de le voir, cette égalisation crée du bruit artificiellement dans l'image (en fait augmente tout autant le bruit que le signal) tout en faisant ressortir les structures qui y sont présentes. Cela a donc deux conséquences antinomiques :

1. les algorithmes d'extraction de caractéristiques peuvent s'appliquer et fournir des informations de type contours ou points,
2. mais le bruit engendré va, soit perturber ces détections en terme de localisation, soit perturber la robustesse des détecteurs vis-à-vis de leurs paramètres.

• La première conséquence est illustrée sur la figure 5.6. Bien qu'il y ait des structures géométriques marquées, comme la rambarde sur le pont du navire, la détection de contours reste difficile à exploiter. Là encore, nous sommes confrontés aux réglages des paramètres, mais on note cependant une amélioration dans le sens où le changement de paramètres du détecteur de contours n'influe que légèrement sur le résultat et semble converger vers une solution stable, mais insatisfaisante tout de même. Par conséquent, à travers cet exemple, on voit bien l'apport que peut amener l'égalisation d'histogramme mais sans fournir encore de résultats exploitables.

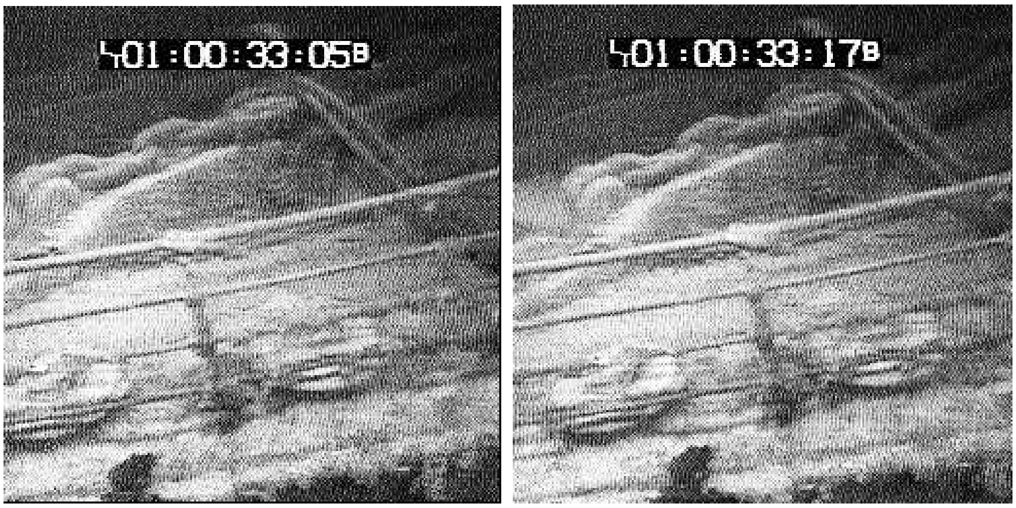
(a) Image 1

(b) Contours extraits (seuil(min, max) : (20, 40)

(c) Contours extraits (seuil (min, max) : (20, 45)

FIG. 5.6 – Titanic : extraction de contours (détecteur de CSS/Canny) avec des paramètres différents sur l'image 1 égalisée

• La seconde conséquence est représentée sur la figure 5.7. Sur deux images proches issues de la séquence, on applique un détecteur de contours avec les mêmes paramètres. On le constate à l'oeil nu, apparier les contours est délicat : leur longueur et leur forme sont différentes. Par conséquent, malgré la présence de structure géométrique forte, il nous est difficile d'envisager l'utilisation des contours.



(a) Image 1

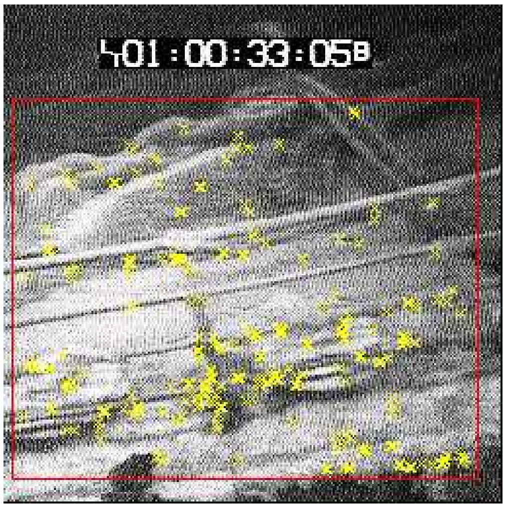
(b) Image 5

(c) Contours extraits (seuil(min, max) : (50, 85)

(d) Contours extraits (seuil (min, max) : (50, 85)

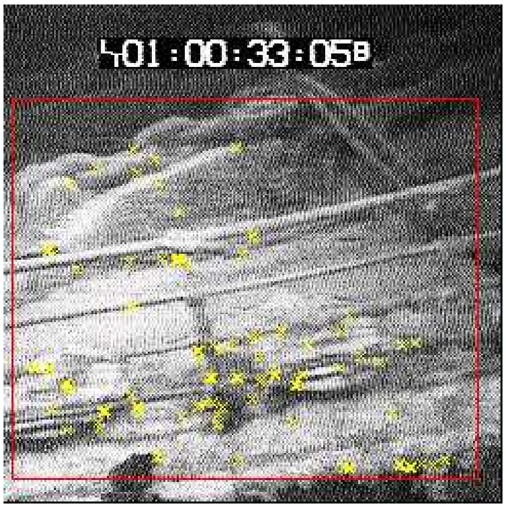
FIG. 5.7 – Titanic : extraction de contours (détecteur de CSS/Canny) avec les mêmes paramètres sur deux images égalisées de la séquence

Nous avons également testé le détecteur de Harris directement sur les images égalisées sans utiliser la structure pyramidale et nous avons fait varier ses paramètres. Les résultats sont présentés sur la figure 5.8. Conformément à ce que l'on attendait, le nombre de points détectés est maintenant satisfaisant mais la localisation de ceux-ci se révèle très sensible aux choix des paramètres. En effet, il est très difficile, pour ne pas dire impossible, de trouver un réglage optimal et générique dans ce type d'images. On constate néanmoins que certains points sont davantage « accrochés » aux structures que d'autres, ce qui n'était pas le cas avant l'égalisation. Nous allons montrer que l'utilisation de la pyramide d'images va nous permettre de sélectionner de façon robuste les points pertinents.

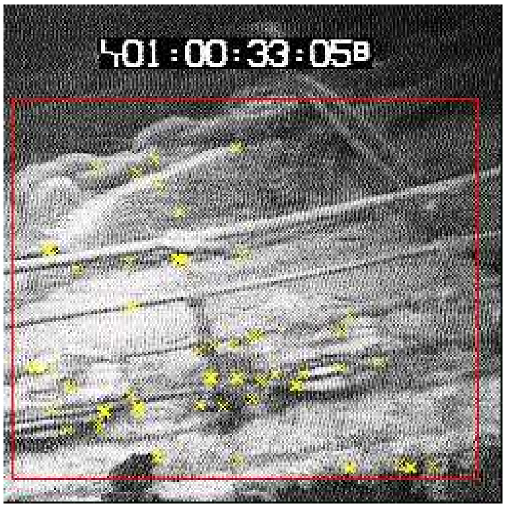


(a) seuil 1000

(b) seuil 1500



(c) seuil 2000



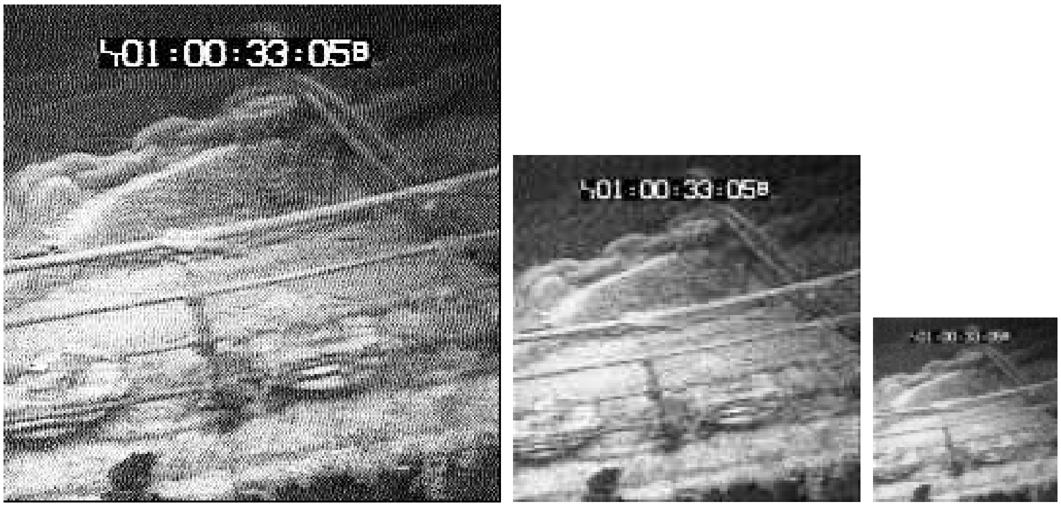
(d) seuil 2500

FIG. 5.8 – Titanic : extraction de points d'intérêts avec des paramètres différents sur l'image 1 égalisée

5.3.3 Optimisation et robustification de cette approche

Nous proposons donc d'utiliser à nouveau la pyramide d'images déjà construite, mais sous une autre forme. En effet, puisque nous avons appliqué l'égalisation sur l'image initiale, la construction de la pyramide se fera à partir de l'image égalisée. Le but de la pyramide étant d'éliminer le bruit dans le signal-image, celle-ci remplit pleinement son rôle ici. De plus, nous n'alourdissons pas les temps de calcul car l'égalisation ne se fait que sur l'image initiale.

La figure[1] 5.9 montre le résultat de la pyramide d'images construite pour une image de la séquence Titanic. Comme précédemment, on retrouve le bruit ajouté dans l'image de résolution maximale, mais celui-ci s'estompe très nettement lorsque l'on remonte dans la pyramide.



(a) image 1, niveau 0 (b) image 1, niveau 1 (c) image 1, niveau 2

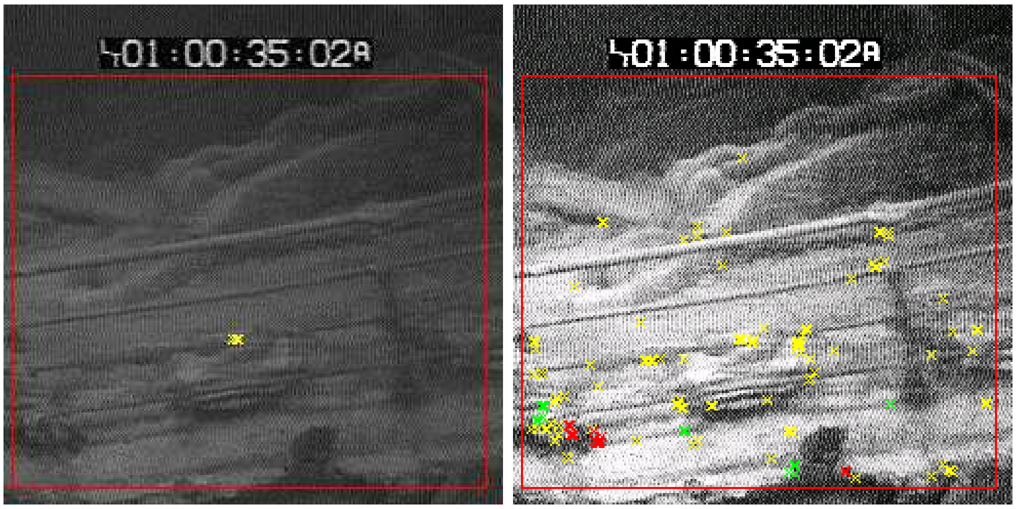
FIG. 5.9 – Titanic : pyramide d'images associée à une image égalisée

Sur cette pyramide d'images, nous appliquons alors notre processus d'extraction et d'appariement pyramidal décrit dans les chapitres précédents. Les figures 5.10(a) et 5.10(b) montrent l'apport important et indéniable de la pyramide d'images couplée avec l'approche multi-échelles.

Dans le cas où nous n'appliquons pas de pré-traitement, nous avons un résultat de mauvaise qualité : d'une part, nous n'avons que très peu de points et d'autre part, leur localisation ne permet d'envisager un quelconque traitement par la suite. En revanche, avec l'égalisation d'histogramme effectuée au préalable, on trouve davantage de points et il est même possible d'effectuer l'appariement pyramidal pour trouver les plus robustes (en rouge sur la figure 5.10(b)).

Le tableau 5.1 donne le nombre de points détectés par niveau de la pyramide. Il est évident que les 6 points robustes ne seront pas suffisants pour estimer le déplacement de la

[1] les échelles ne sont pas respectées pour obtenir un affichage correct



(a) Sans égalisation d'histogramme (b) Avec égalisation d'histogramme

Fig. 5.10 – Résultats de l'extraction des points de Harris avec et sans l'égalisation d'histogramme sur une image

caméra, mais il suffit alors de rajouter les points détectés au niveau 1 pour effectuer le calcul de façon plus robuste.

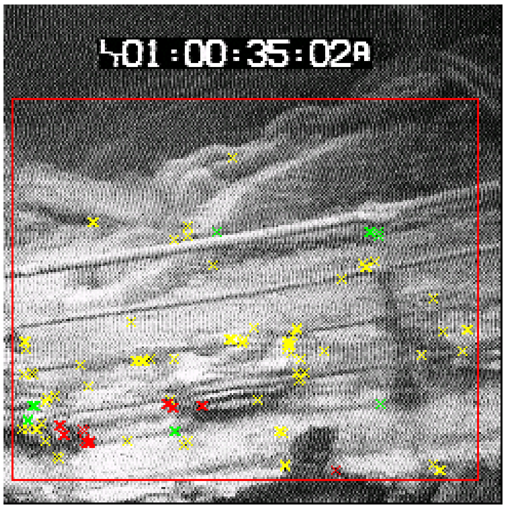
Nombre de points de Harris	sans égalisation d'histograme	avec égalisation d'histograme
Niveau 0	3	125
Niveau 1	0	19
Niveau 2	0	6

Tab. 5.1 – Comparaison de résultats pour l'image issue de la séquence « Titanic » correspondant à la figure 5.3.3

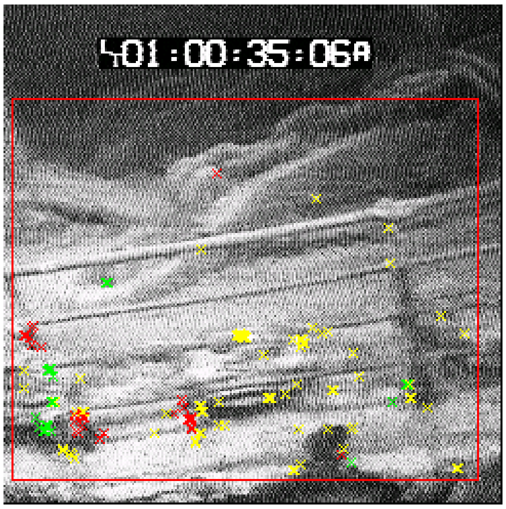
Enfin, nous présentons sur la figure 5.11, le résultat et le suivi des points robustes sur différentes images de la séquence « Titanic ». On constate que malgré un fort bruit dans les images, les points de couleur rouge sont bien extraits successivement tout au long de la séquence. Cette qualité de résultat est obtenue grâce à l'égalisation d'histogramme, appuyée par la structure et l'appariement pyramidal. Nous avons volontairement mis des images plus éloignées ne se suivant pas pour montrer que cela fonctionne lors de déplacements importants, comme c'est le cas dans l'exemple présenté.

5.3.4 Vers le temps-réel

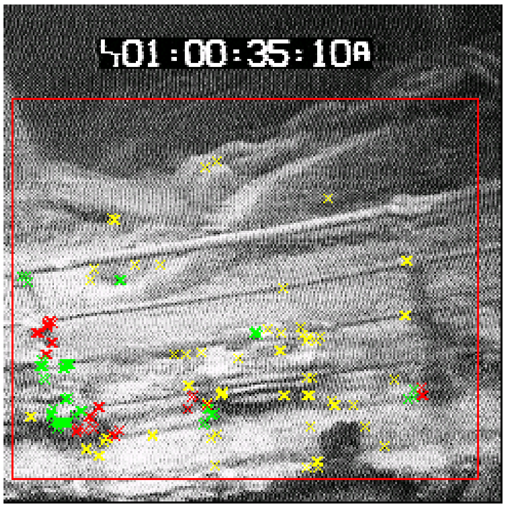
Nous l'avons répété tout au long du manuscrit, nous souhaitons avoir des algorithmes performants et aussi proches que possible de l'éxécution temps-réel. Nous présentons ici



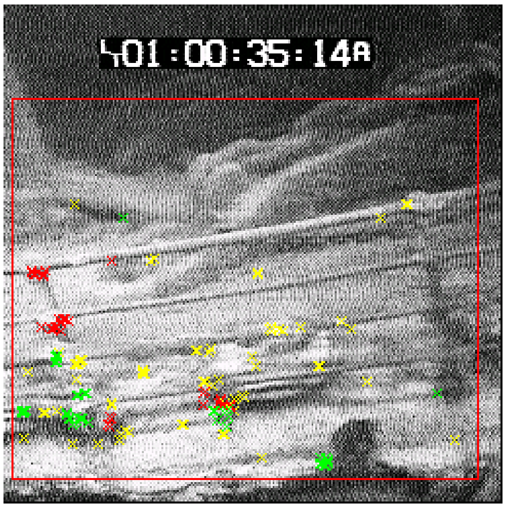
(a) image 14



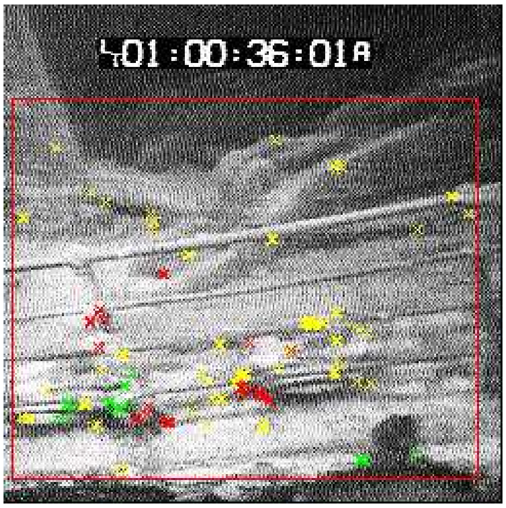
(b) image 15



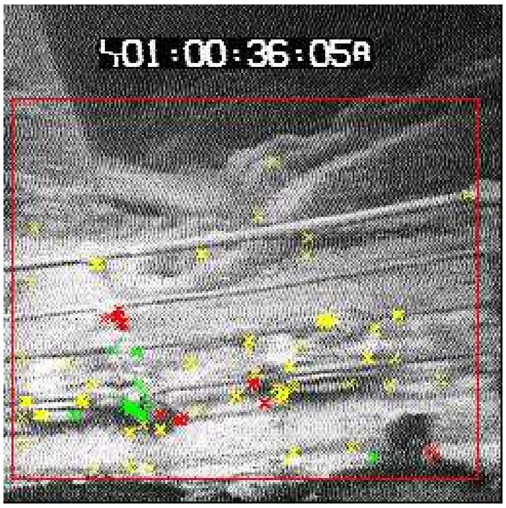
(c) image 16



(d) image 17



(e) image 20



(f) image 21

FIG. 5.11 – Extraction et suivi de points robustes dans la séquence du « Titanic »

quelques résultats de temps de calcul. Précisons que le langage de programmation utilisé n'est pas temps-réel et que l'optimisation de celui-ci n'est certainement pas optimal.

La configuration de la machine utilisée pour les tests est la suivante :

- PC pentium 4 1Ghz, 512 Mo de RAM
- système d'exploitation : Linux 2.2.18
- taille des images intiales : 256x256 pixels
- taille de la zone d'image exploitée : 180x180 pixels

Les temps de calcul sont obtenus en faisant des moyennes sur plusieurs jeux de données (Amphores, Fumerolles, ...). Il faut prendre en compte que l'on manipule aussi des objets graphiques et que l'on gère quelques affichages à l'écran, ce qui ralentit légèrement les algorithmes.

Egalisation d'histogramme	0.03 sec
Construction de la pyramide (3 niveaux)	0.14 sec
Détecteur de Harris (sur les 3 niveaux)	0.19 sec
Appariement pyramidal (pour une image)	0.02 sec
Total pour une image	0.38 sec
Appariement entre images	0.08 sec
Calcul robuste de F	0.5sec

Tab. 5.2 – Temps de calcul

Comme on peut le constater sur le tableau 5.2, les temps de calculs sont faibles. Il n'est pas utopique de croire qu'une implémentation temps réel peut être réalisée. De plus, il est possible d'utiliser certaines cartes vidéo qui ont des circuits intégrés pour des tâches de traitement d'image, comme la construction de pyramides d'images ou le lissage d'une image par une gaussienne.

5.4 Conclusion

Nous avons présenté dans ce chapitre une étude et surtout une solution pour des images comme celles du Titanic où les histogrammes montrent très clairement une très mauvaise distribution des niveaux de gris. L'égalisation d'histogramme permet de faire ressortir les structures de la scène, mais crée du bruit et rend donc à nouveau très difficile l'utilisation des détecteurs de points ou de contours.

Nous avons donc proposé d'appliquer à nouveau la pyramide d'images dont la vocation est d'éliminer les bruits et il est clair qu'ici, elle joue pleinement son rôle. De plus, comme nous la calculons après l'égalisation d'histogramme, nous ne pénalisons pas les temps de calcul.

Les résultats obtenus sur les images de la séquence « Titanic » sont de bonne qualité par rapport d'une part à la qualité intrinsèque des images et d'autre part à d'autres approches robustes. Notons aussi que la taille intiale des images est faible, en règle générale 256x256 pixels, ce qui laisse supposer qu'avec une meilleure résolution, la qualité des résultats serait accrue.

Chapitre 6

Développement logiciel

La mise en oeuvre pratique des différentes approches présentées précédemment n'est pas un problème facile. En effet, il est utile de pouvoir contrôler les multiples étapes de la méthodologie. Comme nous avons étudié un processus global de traitement, nous avons également développé un outil logiciel complet et modulable, que nous présentons dans ce chapitre.

6.1 Introduction

Un travail important de cette thèse a été consacré au développement logiciel des outils nécessaires pour utiliser et traiter des images, quelles qu'elles soient. Nous avons pris le parti de développer un nouvel outil logiciel, et ce, pour plusieurs raisons que nous allons évoquer ici. Le principal avantage de celui-ci est de pouvoir effectuer automatiquement un ensemble de traitements sur des images ou des séquences d'images. Il sera donc très utile lors du prototypage de nouvelles applications.

6.1.1 Pourquoi un nouveau logiciel ?

Si l'on fait le tour des différents logiciels qui existent sur le marché ou dans les laboratoires de recherches, on peut considérer qu'il existe deux tendances : les logiciels libres et commerciaux. Outre les différences de philosophie et de mode de développement, nous souhaitions avoir un logiciel qui réponde à un certain nombre de contraintes :
- portabilité sur diverses plateformes, essentiellement Linux et Windows
- documentation précise et complète pour la programmation
- pérennité du logiciel
- peu coûteux
- algorithmes de traitements d'images récents
- autant que possible, code source accessible
- capacité d'évolution du logiciel, en particulier pour l'ajout de nouveaux algorithmes
- manipulation simple de grands ensembles d'images (multi-images spatial ou séquence temporelle)

Nous avons donc établi une liste concernant les choix potentiels, en essayant de répondre au mieux à ces différents critères :
- **Photoshop** : développé par la société Adobe[1]
- **The Gimp** : développé selon le mode de la communauté "Open-Source", donc par un ensemble de personnes dans le monde, dirigées par des chefs de projets[2]
- **Image Processing Toolbox** pour Matlab : développé par la société Mathworks[3]
- **TargetJr** : intégration des résultats de recherche des principales équipes européennes travaillant dans le domaine de la vision par ordinateur

Ces logiciels ont été conçus avec des buts différents et avec des contraintes qui diffèrent également. Les sociétés qui commercialisent leurs produits ont des avantages majeurs comme

[1]URL : http ://www.adobe.com
[2]URL : http ://www.gimp.org
[3]URL : http ://www.mathworks.com

par exemple, le suivi du logiciel ou une documentation complète, précise et à jour qu'elles fournissent. En revanche, leur principal inconvénient vient du fait que le code source n'est pas disponible, ce qui ne permet pas de savoir ce qui se passe réellement lors de l'utilisation ou qui rend difficile toute exploitation précise de certaines fonctions. De plus, les algorithmes implémentés ne reflètent pas toujours les dernières avancées du domaine de la recherche.

• **Photoshop** et **The Gimp** sont deux logiciels équivalents, l'un étant commercial et destiné à Windows ou à MacOS et l'autre étant sous licence GPL et destiné à Linux. Si les conceptions « philosophiques » diffèrent, il n'en reste pas moins que ce sont tous les deux des outils pour la manipulation d'images 2D. Ils sont destinés avant tout à des infographistes pour faire essentiellement de la retouche d'images. Même si certaines fonctions de traitement bas-niveau existent, cela reste sommaire et difficilement exploitable si l'on envisage de la reconstruction 3D ou si l'on souhaite intégrer des notions temporelles avec des séquences d'images. Les avantages de ces deux logiciels concernent l'ajout de fonctionnalités et le fait que toutes les fonctions d'accès à l'image soient disponibles (cliquer sur un pixel, zoomer, récupérer une zone de l'image, etc...). Etant donnés que ces logiciels sont assez anciens et que leur développement est actif, leur pérennité est assurée et les documentations sont excellentes. En revanche le coût financier est très différent : Photoshop est payant alors que The Gimp est gratuit. Notons aussi que The Gimp offre l'avantage d'un code source disponible, ce qui n'est pas le cas de Photoshop.

• **Image Processing Toolbox** pour Matlab présente de nombreux avantages. Le premier est lié au fait que ce module de traitement est intégré dans Matlab, logiciel de calcul numérique. Cela signifie que nous disposons d'un large ensemble de fonctions mathématiques déjà implémentées, que la manipulation et le tracage des courbes sont possibles et qu'il existe un outil pour créer sa propre interface graphique. De plus, les documentations sont nombreuses et complètes. Cette « Toolbox » reprend un certain nombre de traitements bas-niveau, comme la détection de contours, mais est incomplète par rapport à nos besoins et surtout n'utilise pas des algorithmes récents. Comme précédemment la gestion des séquences vidéo n'est pas intégrée. Accessoirement, le coût financier n'est pas négligeable car il faut acquérir une licence d'utilisation pour Matlab et une autre licence pour la « Toolbox ».

• **TargetJr** propose quant à lui, la solution la plus proche ce que nous recherchons. En effet, dans la mesure où ce sont des chercheurs qui développent ce logiciel, les algorithmes et méthodes implémentées seront ceux parmi les plus récents. Autre intérêt majeur, le code source est disponible. Malheureusement, plusieurs problèmes importants subsistent. Le plus délicat est celui de l'implémentation. Le choix s'est porté sur le langage C++, mais les structures de données et l'agencement des classes sont tels qu'il est difficile d'extraire un simple algorithme de TargetJr. Ensuite, le mode de gestion de projet n'est pas optimal et il n'est pas rare de trouver différentes versions entre les laboratoires, ce qui pose le problème de la pérennité. Toujours du point de vue pratique, TargetJr comporte encore de nombreux bogues, son installation et son utilisation restent délicates.

Pour synthétiser ce qui vient d'être dit, nous avons représenté dans le tableau 6.1 nos critères de choix et la façon dont les logiciels retenus y répondent.

	Photoshop	**The Gimp**	**Imaging Toolbox**	**TargetJr**
Licence	Commerciale	GPL	Commerciale	Copyright TargetJr Consortium
Documentation	Très bonne	Très bonne	Bonne	Bonne
Coût financier	Payant	Gratuit	Payant	Gratuit
Code source	Non	Oui	Non	Oui
Portabilité	Windows	Linux / Windows	Linux / Windows	Linux
Pérénnité	Très bonne	Très bonne	Correcte	Moyenne
Algorithmes récents	Non	Non	Non	Oui
Langage de programmation	C	C / Perl	Matlab	C++
Facilité de programmation	Correcte	Correcte	Bonne	Faible
Facilité d'utilisation	Très bonne	Bonne	Bonne	Faible

TAB. 6.1 – Comparaison des logiciels existants pour le traitement d'images

Comme on peut le constater, chaque logiciel apporte ses qualités propres. Mais aucun ne correspond réellement à ce dont nous avons besoin. Par exemple, si TargetJr répond à nos besoins en terme d'algorithmes robustes et récents, il n'en reste pas moins que son utilisation est loin d'être simple et nos tests l'ont effectivement montrés. De même, si Image Processing Toolbox semble intéressant au prime abord, il n'est pourtant pas orienté vers nos besoins, et de plus, nécessite l'acquisition d'une licence Matlab. Enfin, les logiciels comme Photoshop ou The Gimp sont destinés à la manipulation d'images et non à la vision par ordinateur donc peu adaptés au type de développement que nous souhaitons mettre en oeuvre.

Nous avons donc pris le parti de développer notre propre outil logiciel, selon un cahier des charges bien défini. Ce logiciel se veut complet, modulable et évolutif, tout aussi facile à utiliser pour l'utilisateur final, spécialiste ou non du domaine, que pour le développeur qui souhaite ajouter des algorithmes. Par ailleurs, nous voulons tester et prototyper de nouvelles méthodes à travers ce logiciel qui agira alors comme un outil de validation.

6.1.2 Quel logiciel pour quelles applications ?

Le titre pose bien le problème : un logiciel, oui, mais pour qui et pour quoi faire ? Comme dit précédemment, nous nous sommes soumis à un cahier des charges exigeant. Plusieurs aspects sont d'ailleurs difficiles à spécifier. Le choix a été fait de dissocier au maximum la partie utilisateur final et la gestion graphique du logiciel, de la partie algorithmique et manipulation des structures de données. Il a donc fallu dans un premier temps choisir le langage de programmation pour la partie graphique. Nous avons opté pour les librairies QT de la société TrollTech[4], car elles remplissent les conditions suivantes :

- le code source étant disponible, il est plus facile de bien comprendre ce qui ce passe lors des appels aux différentes fonctions.
- il existe un ensemble de fonctions permettant de manipuler des images et de travailler au niveau du pixel
- les librairies existent aussi bien sous Linux que sous Windows
- la documentation est complète
- QT est stable
- le développement de cette librairie étant actif et de qualité, la pérénnité est assurée
- il existe plusieurs générateurs d'interface pour QT
- les librairies sont gratuites sous Linux
- QT est déjà largement utilisé dans des grosses applications (par exemple, le gestionnaire de bureau KDE pour Linux) et est donc éprouvé
- enfin, QT est codé en C++ et se compile très bien avec un compilateur classique comme g++

Nous avons confronté cette librairie graphique avec Gtk[5], programmée en C Ansi, mais au moment du choix, nous avons estimé que Gtk n'était pas suffisamment avancée et stable, et que les fonctions élémentaires pour la manipulation de pixels n'étaient pas encore pleinement fonctionnelles. Enfin, la documentation n'était pas non plus suffisamment claire entraînant un risque important au niveau du temps de développement.

Nous avons donc retenu la librairie QT et le générateur d'interface QT Architect[6]. L'implémentation du logiciel est donc totalement effectuée en C++ .

6.2 Cahier des charges

Comme tout logiciel, nous le voulons complet, facile à implémenter et à utiliser et ce, avec un cahier des charges bien fourni. Ce cahier des charges doit satisfaire la liste des

[4]http ://www.trolltech.com
[5]http ://www.gtk.org
[6]http ://qtarch.sourceforge.net

fonctionnalités suivantes :
- possibilité de gérer une, deux ou trois caméras calibrées ou non,
- possibilité d'avoir des séquences d'images, donc d'intégrer l'aspect temporel entre les images,
- possibilité d'avoir l'intervention d'un opérateur à tout instant,
- possibilité de travailler sur plusieurs zones dans l'image en même temps,
- possibilité d'affecter différents traitements selon ces zones,
- avoir un système de sauvegarde pour rejouer et/ou modifier une configuration,
- exploiter au mieux les capacités de C++ pour optimiser les algorithmes,
- intégrer des algorithmes récents,
- possibilité d'interfacer ce logiciel avec d'autres logiciels ou algorithmes,
- rester le plus indépendant possible de la bibliothèque graphique pour la partie algorithmique,
- gérer des structures de données complètes mais pas trop lourdes,
- minimiser la taille du code,
- possibilité de manipuler des objets 3D OpenGL.

A partir de ce cahier des charges, nous présentons alors les spécificités des trois grandes parties du logiciel VPI *: Visual Processing Imaging.*

6.3 Structures de données

Le but de VPI est de pouvoir d'une part, manipuler des images et d'autre part, de leur appliquer certains traitements, en particulier, ceux développés durant cette thèse. Plutôt que de prendre le problème au cas par cas, nous le prenons dans son ensemble. Nous sommes donc confrontés à un certain nombre de questions telles que :
- Combien d'images maximum doit-on manipuler à la fois ?
- Peut-on intégrer l'aspect temporel des séquences vidéo dans le traitement ?
- Comment optimiser les coûts algorithmiques ?
- Comment stocker et représenter les données telles que des données images, des points d'intérêt ou des contours ?
- Comment gérer plusieurs caméras ?
- etc ...

L'intérêt du C++ comme langage de programmation et la notion d'héritage de classe deviennent alors évidents. Nous définissons trois classes d'objets que nous manipulons dans VPI :

1. les données de type « images » : pour gérer les pyramides, les zones d'intérêt, ...

2. les données de type « caractéristiques » : les points, les contours, ...

3. les différentes matrices : repères caméras, les matrices intrinsèques, les matrices d'homographies et fondamentales, ...

6.3.1 Les images

6.3.1.1 Vues

Nous avons choisi de manipuler des séquences d'images pouvant provenir de n caméras, $n = 1, 2, 3$, ($n = 1$ dans le cas de la vision monoculaire). Nous considérons que la configuration du système d'acquisition (mono, stéréo ou tri-caméras) est fixe et connue à priori pour une application donnée. Nous définissons alors une *Vue* qui contient soit une, deux ou trois images, selon la configuration de l'acquisition. Pour la première image de la séquence, on a alors : $V_1(I_g, I_m, I_d)$.

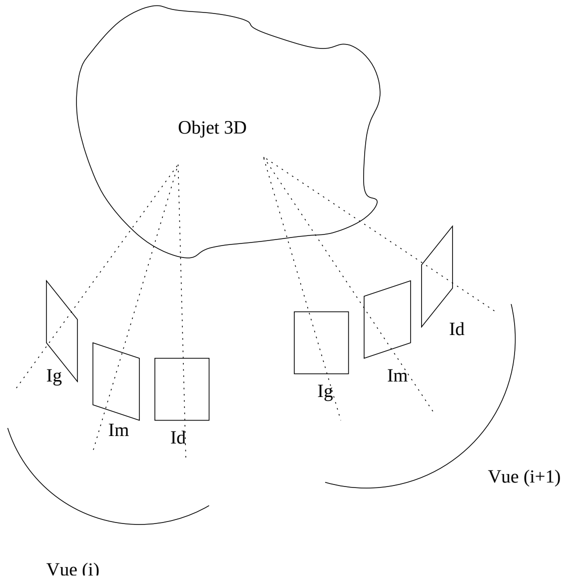


FIG. 6.1 – Notion d'une *Vue* pour VPI

Dans le cas où nous utilisons moins de caméras, il suffit de considérer celle(s) absente(s) comme nulle(s). Si l'on a une séquence d'images en entrée, nous lui associons donc une liste chaînée de *Vues*, que l'on parcourt grâce à un itérateur. Cette liste est indexée pour prendre en compte l'aspect temporel de la séquence.

6.3.1.2 Pyramide d'images et égalisation d'histogramme

Les deux algorithmes liés à ces traitements sont considérés comme à part dans Vpi. En effet, ce sont des étapes préliminaires à tout traitement postérieur. Nous avons donc choisi de les effectuer au chargement de la séquence, sachant qu'il reste tout de même possible de les désactiver.

Avant la construction de la pyramide d'images pour chaque image et comme nous l'avons vu dans le chapitre précédent, nous devons choisir si l'on applique un pré-traitement global, comme par exemple, l'égalisation d'histogramme. Si tel est le cas, c'est au tout début que nous le faisons, de façon à construire la pyramide sur ces nouvelles images. A chaque *Vue*, on applique alors l'algorithme de construction de la pyramide. Ainsi une *Vue* comporte l'ensemble des images et leurs pyramides associées.

6.3.1.3 Méta-Régions

Pour chaque vue, nous avons donc au plus trois images avec les pyramides associées. Par souci de simplicité, dans la suite, on ne considèrera qu'une seule caméra. Nous avons vu que nous souhaitions pouvoir associer à une image plusieurs zones d'intérêt avec des traitements associés différents. L'avantage en termes de temps de calcul et de robustesse de ne faire les traitements que sur des zones ou régions d'intérêt n'est plus à démontrer dans le cadre d'applications de vision dynamique ou active. Par exemple dans la figure 6.2, la fenêtre rouge contient le résultat d'une détection de contours et la fenêtre bleue contient le résultat d'un lissage par une filtre gaussien. Cette possibilité a été rendue possible par l'implémentation du concept de *Méta-Région*.

Le principe des *Méta-Régions* est le suivant : considérant une pyramide d'images, nous définissons une *Méta-Région* comme étant l'ensemble d'une zone créée au niveau le plus bas de la pyramide (image initiale) et projetée dans toute les images de la pyramide.

A l'initialisation, on ne crée une *Méta-Région* que dans la première *Vue*. Elle sera propagée lors des traitements à la fois dans la pyramide et à la fois dans les autres *Vues*, mais restera, par défaut, toujours positionnée au même endroit. Nous avons également prévu la possibilité de déplacer les *Méta-Régions* au cours de la séquence en fonction du traitement réalisé, ce qui peut être utile par exemple, pour prendre en compte le déplacement de la caméra au cours du temps.

6.3.1.4 Régions

Enfin, pour appliquer les algorithmes de traitement dans les zones d'intérêt et ce tout au long de la pyramide, nous avons défini la notion de *Régions*. Ce sont en fait les zones contenues dans les *Méta-Régions*. Par exemple, pour la *Méta-Région* bleue de la figure 6.3, il

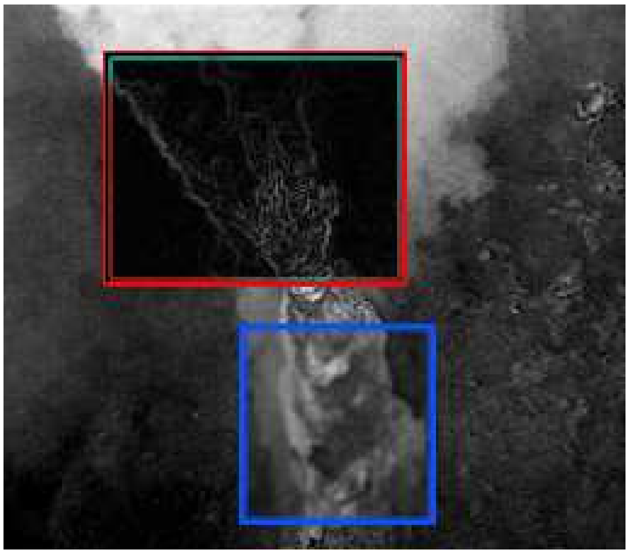
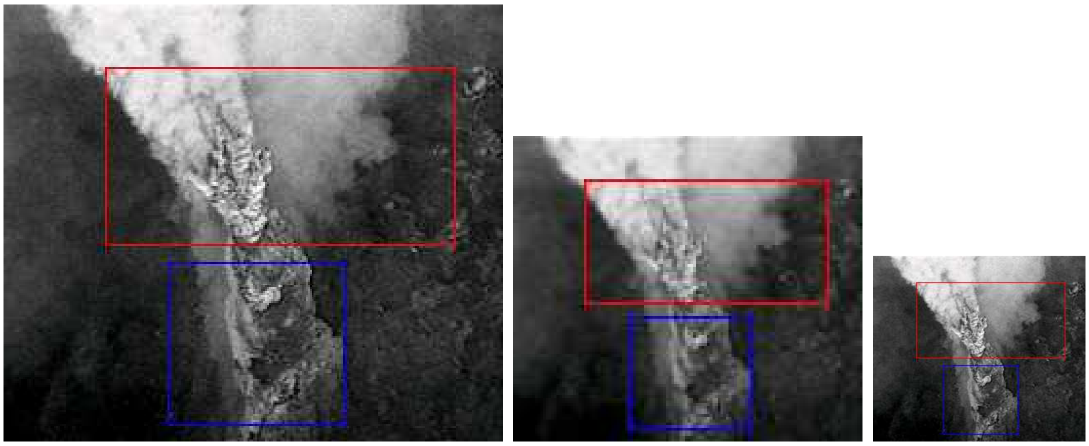


FIG. 6.2 – Zones d'intérêt avec différents algorithmes de vision



(a) Niveau 0

(b) Niveau 1

(c) Niveau 2

FIG. 6.3 – Exemples de deux Méta-Régions

y a trois régions, liées chacune à un niveau de la pyramide. Cette structure permet d'exploiter au mieux l'aspect programmation en langage objet du C++ ainsi que les héritages de classes. C'est au niveau des *Régions* que l'on applique le détecteur de Harris ou un filtre quelconque dans la zone d'intérêt. Une *Région* est donc « rattachée » à un niveau précis de la pyramide, ce qui permet d'avoir la possibilité d'affecter un traitement différent à chaque niveau de la pyramide. Cette possibilité n'a pas été utilisée dans notre application.

6.3.2 Les points d'intérêt

Nous décrivons maintenant la structure adoptée pour les points d'intérêt, détectés avec un des trois détecteurs suivants : SUSAN, CSS ou Harris. Ces points doivent contenir beaucoup d'informations en plus de leur position en pixels dans l'image. On leur définit alors un ensemble d'attributs très complet :

- la valeur en niveau de gris du pixel,
- les coordonnées du pixel dans le repère image courante de la pyramide,
- un attribut de couleur pour l'affichage,
- des pointeurs sur ses parents et ses enfants dans l'arbre d'appariement pyramidal,
- le niveau de la pyramide auquel il appartient,
- le nombre de niveaux auquels on peut l'apparier.

Dans la mesure où nous programmons en C++, il est possible d'avoir plusieurs définitions d'un point. Cela peut-être utile si l'on ne souhaite pas surcharger inutilement les structures des données à manipuler ou bien si l'application recherchée n'a besoin que du strict miminum. De même, si l'on souhaite ajouter de nouveaux attributs, comme des vecteurs d'invariants locaux, il suffit alors de créer une nouvelle structure. Par exemple, nous avons implémenté, entre autres, deux structures de points, ce qui donne le code présenté sur la figure 6.4. Dans ce cas-ci, on peut choisir de spécifier la couleur d'affichage des points dans les images.

On le constate dans cet exemple, les **vpiPoint** dérivent des classes QT et plus précisement **QPoint**. L'intérêt apporté par QT est le suivant : de nombreuses fonctionnalitées sont déjà implémentées pour les **QPoint**, comme par exemple des opérateurs de comparaison, d'égalité, etc.

6.3.3 Structure du logiciel

Dans un soucis de clarté et de pérénnité du logiciel, nous avons, autant que possible, séparé les parties graphiques des parties algorithmiques. Egalement, nous avons choisi d'utiliser des fichiers de configuration permettant de rejouer des algorithmes sur des données.

Plus précisemment, VPI est un outil graphique s'appuyant sur QT. Nous avons donc tout un ensemble de classes destinées à « l'enrobage » du paramétrage des filtres de lissage par

```
vpiPoint::vpiPoint(int x, int y)
  :QPoint(x,y)
{
  color = QColor("blue");
  defaultAttributes();
}

vpiPoint::vpiPoint(int x, int y, QColor col)
  :QPoint(x,y)
{
  color = col;
  defaultAttributes();
}

void vpiPoint::defaultAttributes()
{
  gray = 0;
  uid = POINT_ILLEGAL;
  image = POINT_ILLEGAL;
  level = POINT_ILLEGAL;
  match_level = -1;
  _parent.clear();
  _children.clear();
  weight = -1.0;
  dx=x();
  dy=y();
}
```

FIG. 6.4 – Exemple de code C++ pour la structure des vpiPoints

exemple. Ces parties de code sont donc isolées de la partie algorithmique de VPI. Dans cette partie algorithmique, nous avons là aussi séparé les notions de filtres des classes permettant la manipulation des matrices. Ainsi, ces dernières peuvent même être utilisées en dehors de VPI. De même les structures de données sont isolées des classes gérant le graphisme.

6.3.3.1 Dictionnaires pour les filtres

Dans la terminologie que nous avons choisie, lorsque nous appliquons un traitement à une *Méta-Région*, nous parlons de filtre. Celui-ci peut être un détecteur de points ou encore un filtre de lissage. Afin de permettre à l'utilisateur de gagner du temps et d'optimiser son travail, nous avons prévu la possibilité de stocker sous forme de dictionnaires différents filtres avec ses propres paramètres. La figure 6.5 montre par exemple, la boîte de dialogue du détecteur Css.

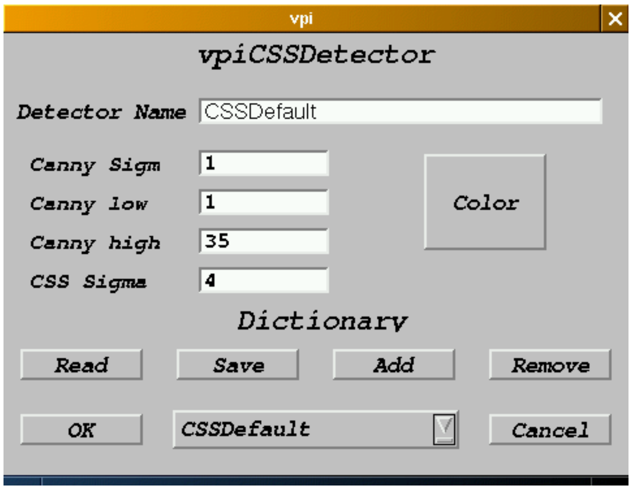


Fig. 6.5 – Détecteur Css dans Vpi

6.3.3.2 Fichiers de configuration

De même que l'on peut stocker et/ou créer de nouveaux filtres, il est possible de stocker des configurations de simulations. Il est clair que si l'on a 200 images de taille 512x512 pixels, que l'on doit calculer la pyramide, choisir plusieurs *Méta-Régions*, les filtres à appliquer et tester différents jeux de paramètres, la manipulation devient vite lassante et longue. Il est donc possible de sauvegarder à tout moment la configuration courante de la simulation. Cela autorise une grande lattitude dans le chargement de configurations : du simple chargement de la séquence d'images jusqu'à l'estimation de la matrice fondamentale.

6.3.3.3 Autres particularités

Parmi les autres spécificités de Vpi , signalons le fait que chaque image est réellement stockée en fonction de son emplacement dans la séquence et donc cela permet de prendre en compte l'aspect temporel de la séquence. Egalement, nous avons fait en sorte que la programmation soit assez simple et que la modification ou l'ajout d'algorithme ne se fasse que dans très peu de fonctions et de classes. Nous avons pour cela utilisé une routine appellée *runit* qui centralise à chaque étape du processus toutes les fonctions mises en jeu. Il est ainsi très simple de tracer l'exécution du programme. Rappelons enfin que la gestion de trois vues simultanées est possible.

6.3.4 Partie graphique

Cet aspect du logiciel est rarement développé et évoqué. Nombres de personnes le négligent, mais dans la mesure où nous manipulons des images, il nous paraît important de pouvoir travailler directement dans les ensembles de pixels. Plus précisemment, un opérateur pouvant intervenir par exemple pour définir des régions d'intérêt, nous avons implémenté un système de gestion à la souris des fenêtres au sein de l'image.

Nous avons pris en compte le fait que l'utilisateur final n'était pas nécessairement un spécialiste de la vision par ordinateur. Ainsi, il peut tout gérer à la souris. Les figures 6.6 et 6.7 montrent deux parties de l'interface de Vpi. La première figure représente le centre de commandes générales (chargement des configurations, construction des pyramides, ...). La seconde figure montre quant à elle, la fenêtre de traitement pour une image de la séquence. On le constate sur le côté gauche, plusieurs boutons permettent de choisir (via des boîtes de dialogue) les traitements à appliquer. Pour l'utilisateur il suffit de choisir les traitements à appliquer sur l'image de début de la séquence et du fait de l'aspect temporel, ceux-ci s'appliqueront sur l'ensemble de la séquence automatiquement.

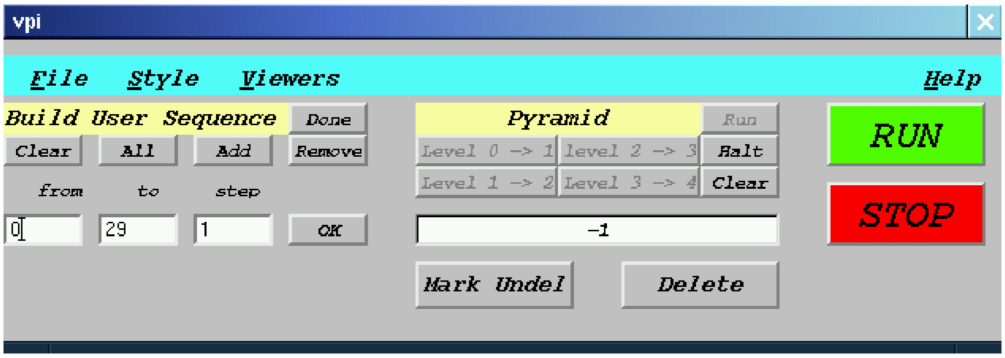


Fig. 6.6 – Vpi : centre de commandes générales

6.3.5 Algorithmes de traitement d'images

Cette partie est sans conteste la plus intéressante de Vpi. Les algorithmes de vision présentés tout au long de ce manuscrit sont intégrés dans le logiciel. Voici la liste des plus importants :

- Implémentation récursive des gaussiennes de R. Deriche
- Extraction de contours : Méthode optimisée de Canny-Deriche, Susan et Css
- Extractions de coins et de points d'intérêt : Détecteurs de Susan, Css et Harris optimisé
- Appariement de points : Mesure de corrélations et appariement pyramidal

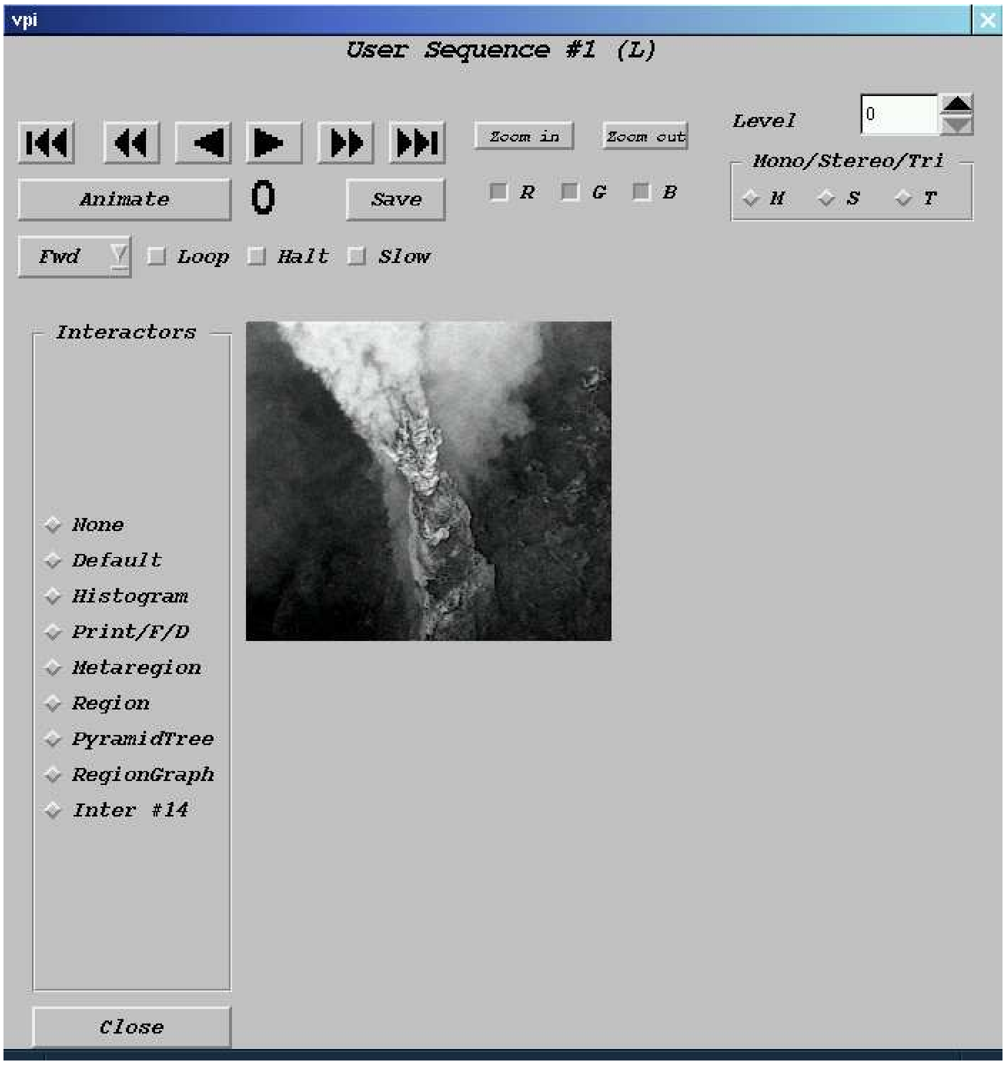


FIG. 6.7 – VPI : algorithme pour une image

– Estimation de la matrice fondamentale : Algorithme des 8 points normalisés et Ransac

A cela, il faut ajouter les librairies de manipulation de matrices (définition, addition, multiplication, décomposition SVD, etc) et de graphes (GTL[7]). Egalement, nous avons aussi ajouté des exécutables comme FMatrix de Z. Zhang pour comparer nos implémentations et ajouter des méthodes de calcul de la matrice fondamentale non linéaire.

6.3.6 Fiche technique

Dans cette partie un peu austère, nous présentons brièvement quelques aspects techniques de Vpi. Ce logiciel a été conçu avec plusieurs objectifs tant sur le plan des algorithmes du traitement d'image que sur le plan technique. Nous avons voulu concevoir un outil logiciel, qui soit robuste, facile à utiliser, portable sur différentes plateformes et évolutif. Voici donc la liste des spécificités de Vpi :

– Bibliothèque graphique : QT. Sa documentation est complète et fournie et les librairies QT existent aussi bien sous Linux que sous Windows.
– Langage de développement : C++ standard. Celui-ci se compile avec un compilateur tel que g++[8]. Le code source est disponible.
– Temps de compilation avec un bi-pentium 4 à 1Ghz sous Linux (noyau 2.2.18), 512 Mo de RAM, QT 2.2, g++ 2.95-3 : 1 minute et 50 secondes
– Taille du code source : 36 Mo
– Possibilité d'extension de Vpi : utilisation des classes C++
– Possibilité d'interfacer avec d'autres logiciels pré-compilés

6.4 Conclusion

En définitive, le logiciel que nous avons développé remplit correctement le cahier des charges que nous nous étions fixé. Sans avoir la prétention de concurrencer ou de remplacer les logiciels existants, il nous permet d'appréhender comme nous le voulons la reconstruction de scènes naturelles ou le prototypage d'applications. L'intérêt majeur vient du fait que nous diposons du code source et que nous pouvons le modifier et le faire évoluer à notre guise.

Egalement, son utilisation est tout à fait possible pour quelqu'un de non-expert dans le domaine de la vision par ordinateur. La programmation de Vpi étant en C++, il est relativement facile d'ajouter des fonctionnalités. Cet outil est donc très complet et permet de traiter dans sa globalité le problème de la reconstruction de scènes naturelles.

[7]http ://www.infosun.fmi.uni-passau.de/GTL/
[8]http ://gcc.gnu.org/

Chapitre 7

Conclusions et perspectives

La vision par ordinateur occupe un espace de recherche de plus en plus important dans les Sciences de l'Ingénieur car elle devient un carrefour incontournable tant ses domaines d'utilisation sont variés. Les recherches se portent de plus en plus sur des problématiques concrètes et dans des cadres naturels. Ce travail de thèse s'inscrit dans cette perspective, néanmoins, il est clair qu'un long chemin reste à parcourir pour mettre à la disposition d'utilisateurs finaux des outils performants et robustes capables de résoudre des problèmes complexes sans nécessiter des compétences de spécialistes du traitement d'images.

7.1 Contributions

Cette thèse a apporté des contributions sur un problème encore ouvert aujourd'hui : l'utilisation de techniques de traitement d'images dans des applications mettant en jeu des scènes naturelles. En effet, si on peut affirmer aujourd'hui que la théorie de la vision géométrique est maintenant bien posée, entre autres, via la géométrie épipolaire, les équations qui en découlent sont, hélas, souvent très difficiles à résoudre en pratique, l'exemple le plus connu étant celui de l'estimation de la matrice fondamentale. De nombreux auteurs ont proposé différents algorithmes pour l'évaluer.

Pourtant, la vision par ordinateur est amenée à s'appliquer dans des environnements naturels complexes. Rappelons que nous sommes dans le cas très général et quasiment le pire qui soit, car nous n'avons qu'une seule caméra non calibrée, que nous ne connaissons ni son mouvement, ni scènes observées, que nous ne disposons d'aucune information sur les situations à l'exception du fait que les objets qui nous intéressent sont rigides. Cela signifie, que ceux-ci n'ont pas de propriétés géométriques remarquables (axe de symétrie, coins, contours bien définis), mais aussi que leur texture est totalement inconnue. Enfin, le but de cette thèse est de fournir d'une part, une étude précise sur ce que l'on peut attendre des algorithmes de la vision par ordinateur sur ce type d'images, si possible pour remonter à des modèles 3D complets et d'autre part de fournir des nouvelles approches là ou les autres fonctionnent mal. Notons que nous n'avons pas pris en compte les problèmes d'occultations.

La principale contribution de cette thèse est donc liée à l'utilisation de la vision dans le monde « naturel » et plus particulièrement sous-marin. Nous avons donc vu au cours des différents chapitres que les approches théoriques classiques ne s'appliquent que rarement ou mal à ces milieux-là. En effet, les structures géométriques fortes ne sont guère présentes, les bruits externes sont nettement plus nombreux qu'en laboratoire et rarement modélisables, il est difficile d'avoir des modèles des scènes, etc.

Nous avons donc apporté un soin particulier à rester pragmatique et à coller à la réalité, en tenant compte de ses spécificités. Par exemple, essayer de travailler au dixième de pixels n'a qu'un intérêt très limité sous l'eau dans la mesure où il est déjà quasi-impossible d'obtenir des informations stables et robustes ! Nous avons donc développé une méthodologie globale adaptée aux images naturelles et une chaîne de traitement complète, pour effectuer de la reconstruction 3D.

Pour cela, nous nous sommes appuyés sur une représentation multi-échelles et pyramidale des images. Ce choix n'est pas totalement arbitraire, car il est vrai que nous aurions pu choisir d'autres types d'échelles (par exemple avec des approches par ondelettes), mais en pratique, il existe des cartes d'acquisition de flux vidéo qui construisent en temps réel ces pyramides. Nous avons donc pris le parti de s'appuyer sur ce type d'opportunité.

Une étude sur les détecteurs classiques de la littérature et certains très récents nous a montré qu'en pratique, leur comportement n'est pas des plus fiables. L'étude comparative entre les détecteurs de Harris, Css et Susan, dans le cas d'images naturelles est, à notre connaissance, unique pour le moment. Nous avons donc choisi le détecteur de Harris que nous avons implémenté de façon robuste et très rapide. Nous avons combiné le détecteur avec la pyramide d'images en poursuivant deux objectifs. Le premier est que les points détectés soient robustes aux bruits et le second est que ces points soient bien localisés. Également, nous nous sommes intéressés à la classification ou plus précisemment à l'étiquetage de points d'intérêt ce qui permet d'avoir des classes de points en fonction de leur qualité intrinsèque.

L'appariement est une étape préalable à la reconstruction d'un modèle 3D. Nous avons comparé les performances de deux approches maintenant classiques. Les points robustes sont dès lors beaucoup plus facile à mettre en correspondance. Cela nous assure deux choses : tout d'abord, on diminue le risque d'avoir des faux appariements et ensuite, les points que nous avons appariés sont de bons points robustes. Nous avons à nouveau utilisé la pyramide d'images pour apparier les points robustes aux échelles les plus élevées, puis nous avons imposé des contraintes de localité, ce qui nous permet de limiter les recherches lors de l'algorithme d'appariement. Nous nous sommes intéressés au problème du suivi de points. Notre approche permet de suivre de bons points même lorsqu'il y a des bruits ou des changements de mouvement brusques et les résultats obtenus sont encourageants.

Enfin, la dernière étape est la reconstruction 3D proprement dite. Soit nous utilisons des méthodes basées sur l'ajustement de faisceaux ne nécessitant pas l'estimation directe de la matrice fondamentale F, soit nous la calculons explicitement. Nous avons testé les deux approches en nous concentrant sur l'estimation pratique de la matrice fondamentale. A nouveau, notre approche multi-échelles et la classification des points nous permet d'estimer F correctement avec peu, mais de bons points. En s'appuyant sur les travaux d'autres chercheurs de l'équipe, une étude a également été menée concernant les conditions d'existence de la matrice fondamentale.

Cette approche a été testée avec succès sur des images acquises en laboratoire et sur des images sous-marines. En comparant notre approche avec des algorithmes classiques de la littérature, nous avons pu mettre en évidence leurs défauts et lacunes. Notre approche s'est révélée efficace et plus rapide que ceux-ci dans les cas d'images naturelles.

Ces algorithmes et la méthode globale de reconstruction à partir de séquence d'images ont été intégrés dans un outil logiciel que nous avons développé. Celui-ci est très complet et sert à présent de plateforme de développement et de prototypage d'applications au sein du laboratoire. Comme décrit dans le chapitre 6, ce logiciel dénommé Vpi, est développé de telle sorte que l'ajout de nouvelles fonctionnalités soit aisé pour le développeur. Egalement, l'utilisateur final n'a pas besoin d'être spécialiste du domaine de la vision par ordinateur pour effectuer ses différents traitements.

7.2 Poursuite des travaux

A l'issue de ce travail de thèse, plusieurs problèmes demeurent en suspens et d'autre méthodes restent encore à développer. Nous présentons ici ce qui nous semble être le cadre d'investigation et où des avancées importantes sont tout à fait envisageables.

7.2.1 Contrainte de rigidité

Tout d'abord, et nous l'avons évoqué dans ce manuscrit, la seule information que l'on a a priori de la scène, est que les objets à reconstruire sont rigides. Nous n'avons pas pris parti de l'exploiter car les temps de calculs peuvent être longs, mais cette piste semble prometteuse. En effet, on imagine très bien que si l'image se dégrade au cours de l'acquisition, l'objet étant rigide, certains points d'intérêts qui disparaissent peuvent facilement être « virtuellement » suivis. On pourrait ainsi gérer une partie des problèmes liés aux occultations. Ensuite, la rigidité de l'objet permettrait de définir une sorte de graphe de l'objet considéré. Ce graphe a deux fonctionnalités : tout d'abord, on peut envisager d'apparier grossièrement les graphes entre deux images pour ensuite apparier finement les points d'intérêt localement. Une telle approche permettrait de manipuler au niveau local des informations de niveau « signal » en superposant des contraintes géométriques globales à l'objet. La représentation de la rigidité peut se faire par exemple par des triangulations ou des graphes. Nous avons commencé à étudier la triangulation de Delaunay, mais nous nous heurtons à de nombreux problèmes (Espiau, 2000). Par exemple, du fait que la localisation des points n'est pas parfaite et change au cours du temps, les triangulations obtenues ne sont pas globalement stables et il est alors très difficile de mettre en correspondance certains arcs du graphe. De plus, les algorithmes de manipulation de graphes peuvent se réveler assez coûteux en temps de calcul. Néanmoins, nous avons observé que certaines structures des graphes sont conservées et nous pensons que l'exploitation de cette propriété accroitrait la robustesse de la reconstruction.

7.2.2 Modèles de lumière et de bruits

En imagerie sous-marine, nous l'avons vu, les conditions d'acquisitions sont assez mauvaises et nombre de paramètres viennent perturber le signal-image : le sable, les poissons, ... L'illumination des objets par une source spot rigidement liée à la caméra pose également deux problèmes majeurs : d'une part, la diffusion de la lumière est importante sous l'eau et la portée reste faible et d'autre part, les textures varient au cours du temps à cause du phénomène d'ombres portées. Il serait également intéressant d'étudier plus en profondeur les modèles de source de lumière car nous observons un phénomène de halo où la lumière est forte au centre et très faible sur les bords, ce qui empêche une bonne détection de points uniforme dans l'image. Une modélisation de ce phénomène pourrait être utilisée pour effectuer une correction de l'image à l'acquisition rendant le résultat de la détection plus isomorphe. Partant de ce principe, on pourrait également essayer de modéliser les différents bruits liés aux capteurs.

7.2.3 Estimation des matrices fondamentale et d'homographie

Sur le plan théorique, nous l'avons vu, dans le cas des amphores, les points extraits sont pratiquement dispersés sur un plan, ce qui ne facilite pas l'évaluation de la matrice fondamentale, car, en pratique, elle reste toujours estimable (du fait du bruit) mais fausse. En revanche, il est tout à fait possible de calculer la matrice d'homographie. Nous n'avons pas pu trouver de méthode permettant de choisir automatiquement quelle matrice évaluer. La seule approche que nous ayons essayée est d'ajouter dans l'algorithme de calcul un couple de points appariés et de comparer l'erreur résiduelle ainsi obtenue. Ce problème est toujours d'actualité et reste difficile. L'une des pistes à suivre que nous avons commencé à étudier est celle basée sur les travaux d'E. Malis. L'idée est d'estimer systématiquement la matrice d'homographie plutôt que la matrice fondamentale. Nous avons présenté des résultats encourageants avec cette méthode.

7.2.4 Vers l'asservissement visuel en milieu naturel

Enfin, et c'est certainement l'application la plus directe, mais pas nécessairement la plus facile, ces travaux de thèse pourraient être utilisés en vue de faire de la reconstruction d'objets naturels inconnus par vision active en encapsulant nos algorithmes dans une approche de type « asservissement visuel ». Cette problématique est de plus en plus étudiée, mais présuppose deux résultats fondamentaux : des algorithmes « temps-réel » et avoir des points robustes que l'on peut suivre durant l'asservissement. Nous n'avons que partiellement répondu au premier point concernant l'implémentation « temps-réel », mais nous pensons sincèrement que l'évolution du matériel aidant, la possibilité d'avoir des cartes d'acquisitions performantes et une implémentation réellement dédiée au temps réel, permettront de pallier ce problème. Par contre, nous avons montré qu'il était possible de suivre des points robustes dans des séquences naturelles inconnues. Peut-être faudra-t-il utiliser des schémas d'asservissement visuel spécifiques à ce type d'images comme par exemple celui développé par E. Malis dans (Malis et al., 1999) ?

Bibliographie

ALLEZARD, N., DHOME, M., et JURIE, F. (1999). « Mise en correspondance multi-échelles ». Dans *Orasis 99*, pages 73–84, Aussois. (page 29)

ASADA, H. et BRADY, M. (1986). « The Curvature Primal Sketch ». *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1) :2–14. (page 25)

AYACHE, N. (1983). « *Un système de vision bidimensionnelle en robotique industrielle* ». Thèse de Doctorat, Université de Paris-Sud, Centre d'Orsay. (page 60)

BALLARD, D. et BROWN, C. (1982). *Computer Vision*. Prentice Hall. (page 16)

BEAUDET, P. R. (1978). « Rotationally Invariant Image Operators ». Dans *Proceedings of International Joint Conf. on Pattern Recognition*, pages 579–583. (pages 25, 26)

BLAKE, A. et ISARD, M. (1998). *Active Contours*. Springer. (page 24)

BLANC, J. (1998). « *Synthèse de nouvelles vues d'une scène 3D à partir d'images existantes* ». Thèse de Doctorat, Institut National Polytechnique de Grenoble, France. (page 42)

BLASZKA, T. et DERICHE, R. (1994). « Recovering and Characterizing Image Features Using An Efficient Model Based Approach ». Rapport technique 2422, INRIA. (page 25)

CANNY, J. (1986). « A Computational Approach to Edge Detection ». *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6) :679–698. (page 17)

CHABAT, F., YANG, G., et HANSELL, D. (1999). « A Corner Orientation Detector ». *Image and Vision Computing*, 17(10) :761–769. (page 27)

CHRISTY, S. (1998). « *Modélisation tridimensionnelle d'objets quelconques par vision dynamique* ». Thèse de Doctorat, Institut National Polytechnique de Grenoble, France. (page 43)

COCQUEREZ, P. et PHILIPP, S., éditeurs (1995). *Analyse d'Images : Filtrage et Segmentation*. Masson. (page 21)

CSURKA, G. (1996). « *Modélisation projective des objets tridimensionnels en vision par ordinateur* ». Thèse de Doctorat, Université de Nice - Sophia-Antipolis, France. (page 61)

DERICHE, R. (1987). « Using Canny's Criteria to Derive a Recursively Implemented Optimal Edge Detector ». *The International Journal of Computer Vision*, 1(2) :167–187. (pages 17, 30, 37)

DERICHE, R. et FAUGERAS, O. (1990). « 2D-Curves Matching Using High Curvatures Points : Applications to Stereovision ». Dans *Proceedings of the 10th International Conference om Pattern Recognition*, volume 1, pages 240–242, Atlantic City. (page 25)

DERICHE, R. et FAUGERAS, O. (1995). « Les EDP en traitement des images et vision par ordinateur ». Rapport technique RR 2697, INRIA, France. (page 21)

DERICHE, R. et GIRAUDON, G. (1991). « Accurate Corner Detection : an Analytical Study ». Rapport technique RR 1420, INRIA, France. (page 26)

DERICHE, R. et GIRAUDON, G. (1993). « A Computational Approach For Corner And Vertex Detection ». *The International Journal of Computer Vision*, 10(2) :101–124. (page 26)

DEVERNAY, F. (1997). « *Vision stéréoscopique et propriétés différentielles des surfaces* ». Thèse de Doctorat, École Polytechnique, France. (page 42)

DEVERNAY, F. et FAUGERAS, O. (1995). « From Projective to Euclidean Reconstruction ». Rapport de Recherche 2725, INRIA, France. (page 70)

DRESCHLER, L. et NAGEL, H. (1982). « Volumetric model and 3d trajectory of a moving car derived from monocular tv frame sequence of a street scene ». Dans *Proceedings of Computer Vision, Graphics and Image Processing*, volume 20, pages 199–228. (pages 25, 26)

DUFOURNAUD, Y. (2001). « *Navigation aérienne et guidage terminal à partir de données bidimensionnelles* ». Thèse de Doctorat, Institut National Polytechnique de Grenoble, France. (pages 29, 45)

DUFOURNAUD, Y., SCHMID, C., et HORAUD, R. (2000). « Matching Images with Different Resolutions ». Dans *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 612–618, South Carolina, USA. (page 45)

ESPIAU, F.-X. (2000). « Extraction de points robustes dans des images naturelles complexes ». Dans *13èmes Journées des Jeunes Chercheurs en Robotique*, Rennes. (page 114)

FAUGERAS, O. (1992). « What can be seen in three dimensions with an uncalibrated stereo rig ? ». Dans *Proceeding of the 2nd European Conference on Computer Vision*, pages 563–578, Santa Margherita Ligure, Italy. (pages 67, 68)

FAUGERAS, O. (1993). *Three-Dimensional Computer Vision : A Geometric Viewpoint*. MIT Press. (pages 17, 64, 68)

FAUGERAS, O. et HERMOSILLO, G. (2001). « Well-posedness of eight problems of multi-modal statistical image-matching ». Rapport technique 4235, INRIA, France. (page 81)

FAUGERAS, O., LUONG, Q.-T., et PAPADOPOULO, T. (2001). *The Geometry Multiple Images*. MIT Press. (page 64)

FAUGERAS, O. et PAPADOPOULO, T. (1998a). « Grassmann-Cayley Algebra for Modeling Systems of Cameras and the Algebraic Equations of the Manifold of Trifocal Tensors ». *Transaction of the Royal Society*, pages 1123–1152. (page 71)

FAUGERAS, O. et PAPADOPOULO, T. (1998b). « A nonlinear method for estimating the projective geometry of three views ». Dans *International Conference on Computer Vision*, pages 477–484. (page 71)

FISCHLER, M. et BOLLES, R. (1981). « Random sample consensus : A paradigm for model fitting with applications to image analysis and automated cartography ». *Communications of the ACM*, 24 :381–385. (page 70)

FÖRSTNER, W. et GÜLCH, E. (1987). « A fast operator for detection and precise location of distinct points, corners and centres of circular features ». Dans *Proceedings of Intercommission Conference on Fast Processing of Photogrammetric Data*, pages 281–305, Switzerland. (pages 26, 28)

GIBSON, J. (1950). *The Perception of the Visual World*. Houghton Mifflin, Boston, Mass. (page 46)

HARRIS, C. et STEPHENS, M. (1988). « A Combined Corner and Edge Detector ». Dans *Proceedings of the 4th Alvey Vision Conference*, pages 147–151. (pages 25, 26, 28)

HARTLEY, R. (1995). « In defense of the 8-point algorithm ». Dans *Proceedings of the 5th ICCV*, pages 1064–1070. (page 70)

HARTLEY, R. (1997). « Lines and points in three views and the trifocal tensor ». *The International Journal of Computer Vision*, 22(2) :125–140. (page 71)

HARTLEY, R. I. et ZISSERMAN, A. (2000). *Multiple View Geometry in Computer Vision*. Cambridge University Press. (pages 64, 71)

HERMOSILLO, G. et FAUGERAS, O. (2001). « Dense Image Matching with Global and Local Statistical Criteria : a Variational Approach ». Dans *Proceedings of the International Conference on Computer Vision*. (page 81)

HORAUD, R. et MONGA, O. (1995). *Vision par Ordinateur : outils fondamentaux, 2ème édition*. Hermès. (page 16)

HORN, B. et SCHUNCK, B. (1981). « Determining Optical Flow ». *Artificial Intelligence*, 17 :185–204. (page 46)

JOLION, J.-M. et ROSENFELD, A. (1994). *A Pyramid Framework for Early Vision*. Kluwer Academic Publisher. (page 48)

KASS, M., WITKIN, A., et TERZOPPOULOS, D. (1988). « Snakes : Active Contour Models ». *The International Journal of Computer Vision*, 1 :321–331. (pages 18, 24)

KITCHEN, L. et ROSENFELD, A. (1982). « Gray-level corner detection ». *Pattern Recognition Letters*, 1(2) :95–102. (pages 25, 26)

KOENDERINCK, J. J. et DOORN, A. J. V. (1987). « Representation of local geometry in the visual system ». Dans *Biological Cybernetics*, volume 55, pages 367–375. (page 44)

KORNPROBST, P. (1998). « *Contributions à la restauration d'images et à l'analyse de séquences : Approches Variationnelles et Equations aux Dérivées Partielles* ». Thèse de Doctorat, Université de Nice-Sophia Antipolis. (page 46)

Kumar, V. et Desai, U. (1996). « Image interpretation using bayesian networks ». *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(1) :74–77. (page 21)

Laganière, R. (1998). « Morphological Corner Detection ». Dans *Proceedings of the 6th International Conference on Computer Vision*, pages 280–285. (page 27)

Lhuillier, M. (1999). « Joint View Triangulation for two Views ». Dans *Orasis 99*, pages 151–158, Aussois, France. (page 45)

Lhuillier, M. et Quan, L. (2000). « Appariement dense robuste à l'aide de contraintes géométriques locales et globales ». Dans *RFIA*, volume III, pages 215–223, Paris. (page 45)

Lingrand, D. (1999). « *Analyse adaptative du mouvement dans des séquances monoculaires non calibrées* ». Thèse de Doctorat, Université de Nice - Sophia-Antipolis, France. (pages 64, 67, 71)

Longuet-Higgins, H. C. (1981). « A computer algorithm for reconstructing a scene from two projections ». *Nature*, 293 :133–135. (pages 68, 70)

Lorette, A. (1999). « *Analyse de texture par méthodes markoviennes et par morphologie mathématique : application à l'analyse des zones urbaines sur des images satellitales* ». Thèse de Doctorat, Université de Nice - Sophia-Antipolis, France. (page 21)

Lucido, L., Opderbecke, J., Rigaud, V., Deriche, R., et Zhang, Z. (1996). « A terrain referenced underwater positioning using sonar bathymetric profiles and multiscale analysi ». Dans *Proc. OCEANS 96 MTS-IEE*, Fort Lauderdale, Florida (US). (page 46)

Luong, Q.-T. (1992). « *Matrice Fondamentale et Calibration Visuelle sur l'Environnement-Vers une plus grande autonomie des systèmes robotiques* ». Thèse de Doctorat, Université de Paris-Sud, Centre d'Orsay. (page 68)

Malis, E. (1998). « *Contributions à la modélisation et à la commande en asservissement visuel* ». Thèse de Doctorat, Université de Rennes 1. (page 71)

Malis, E., Chaumette, F., et Boudet, S. (1999). « 2 1/2 D Visual Servoing ». *IEEE Trans. on Robotics and Automation*, 15(2) :234–246. (pages 75, 115)

Malis, E., Chaumette, F., et Boudet, S. (2000). « 2 1/2 D Visual Servoing with Respect to Unknown Objects Through a New Estimation Scheme of Camera Displacement ». *The International Journal of Computer Vision*, 37(1) :79–97. (page 74)

Meer, P., Mintz, D., Rosenfeld, A., et Kim, D. (1991). « Robust Regression Methods for Computer Vision : A Review ». *The International Journal of Computer Vision*, 6(1) :59–70. (page 70)

Mokhtarian, F. et Suomela, R. (1998). « Robust Image Corner Detection Through Curvature Scale Space ». *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12) :1376–1381. (pages 25, 28)

Montagnat, J. (1999). « *Modèles déformables pour la segmentation et la modélisation d'images médicales 3D et 4D* ». Thèse de Doctorat, Université de Nice - Sophia-Antipolis, France. (page 46)

MORAVEC, H. (1977). « Towards Automatic Visual Obstacle Avoidance ». Dans *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, pages 584–586, Cambridge. (pages 25, 26, 28)

NAGEL, H. (1983). « Displacement Vectors Derived from Second Order Intensity Variations in Image Sequences ». *Computer Vision, Graphics and Image Processing*, 21 :85–117. (page 26)

NOBLE, J. A. (1988). « Finding Corners ». *Image and Vision Computing*, 6(2) :121–128. (pages 25, 29)

ODOBEZ, J.-M. et BOUTHEMY, P. (1995). « Robust multiresolution estimation of parametric motion models ». *Journal of Visual Communication and Image Representation*, 6(4) :348–365. (page 46)

PAPADOPOULO, T. (1995). « *Analyse du mouvement de courbes rigides 3D à partir de séquences d'images monoculaires* ». Thèse de Doctorat, Université de Paris-Sud Centre d'Orsay. (page 46)

PARAGIOS, N. (2000). « *Geodesic Active Regions and Level Set Methods : Contributions and Applications in Artificial Vision* ». Thèse de Doctorat, Université de Nice - Sophia-Antipolis, France. (page 24)

PARAGIOS, N. et DERICHE, R. (1998). « Geodesic active regions for texture segmentation ». Rapport technique RR 3440, INRIA, France. (page 24)

ROHR, K. (1992). « Recognizing corners by fitting parametric models ». *The International Journal of Computer Vision*, 9(3) :213–230. (page 25)

ROUSSEEUW, P. et LEROY, A. (1987). *Robust Regression and Outlier Detection*. John Wiley & Sons, New York. (page 70)

SAMSON, C. (2000). « *Contribution à la classification d'images satellitaires par approche variationnelle et équations aux dérivées partielles* ». Thèse de Doctorat, Université de Nice - Sophia-Antipolis, France. (page 21)

SCHMID, C. (1998). « *Appariement d'images par invariants locaux de niveaux de gris* ». Thèse de Doctorat, Institut National Polytechnique de Grenoble, France. (page 43)

SCHMID, C. et MOHR, R. (1997). « Local Greyvalue Invariants for Image Retrieval ». *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19 :530–534. (page 43)

SCHMID, C., MOHR, R., et BAUCKHAGE, C. (1998). « Comparing and Evaluating Interest Points ». Dans *Proceedings of the 6th International Conference on Computer Vision*, pages 230–235. (pages 26, 30)

SHI, J. et TOMASI, C. (1994). « Good Features to Track ». Dans *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, Seattle. (page 43)

SMITH, S. M. et BRADY, J. M. (1997). « SUSAN - A New Approach to Low Level Image Processing ». *The International Journal of Computer Vision*, 23(1) :45–78. (pages 25, 27)

Sturm, P. (1997). « *Vision 3D non calibrée : Contributions à la reconstruction projective et étude des mouvements critiques pour l'auto-calibrage* ». Thèse de Doctorat, Institut National Polytechnique de Grenoble, France. (pages 67, 69, 71, 73)

Tomasi, C. et Kanade, T. (1991). « Detection and Tracking of Point Features ». Rapport technique CS-91-132, Carnegie Mellon University. (page 43)

Torr, P., Fitzgibbon, A., et Zisserman, A. (1998). « Maintaining Multiple Motion Model Hypotheses Over Many Views to Recover Matching and Structure ». Dans *International Conference on Computer Vision*, pages 485–491, Bombay, India. (page 74)

Triggs, B., McLauchlan, P., Hartley, R., et Fitzgibbon, A. (1999). « Bundle Adjustment - A Modern Synthesis ». Dans *Vision Algorithms'99*. (page 73)

Witkin, A. P. (1987). Scale-Space Filtering. Dans Fischler, M. A. et Firschein, O., éditeurs, *Readings in Computer Vision : Issues, Problems, Principles, and Paradigms*, pages 329–332. Kaufmann, Los Altos, CA. (page 48)

Zhang, Z., Deriche, R., Faugeras, O., et Luong, Q. (1995). « A Robust Technique for Matching Two Uncalibrated Images Through the Recovery of the Unknown Epipolar Geometry ». *Artificial Intelligence Journal*, 78 :87–119. (page 70)

Zhang, Z., Deriche, R., Luong, Q.-T., et Faugeras, O. (1994). « Robust Recovery of the Epipolar Geometry for an Uncalibrated Stereo Rig ». Dans *Proceedings ECCV 94*, volume I, pages 567–576. (page 70)

Zuniga, O. et Haralick, R. (1983). « Corner detection using the facet model ». Dans *Proceedings of the International Conference on Computer Vision*, pages 30–37. (page 26)

Métrologie 3D par vision active sur des objets naturels sous-marins

Ce travail de thèse se place dans le cadre d'une application pour le traitement de scènes naturelles sous-marines. Si le formalisme apporté par la géométrie projective dans le monde de la vision par ordinateur a permis de mieux appréhender cette discipline, l'étude de scènes naturelles pose encore de nombreux problèmes.

Nous présentons ici une méthodologie complète pour l'étude de scènes sous-marines acquises avec une seule caméra non calibrée en vue de faire de la reconstruction projective. Dans un premier temps, nous nous intéressons à l'extraction de caractéristiques robustes, phase préliminaire indispensable pour tout traitement d'images. Du fait du caractère propre des scènes observées (pas de formes géométriques simples, bruit important, non connaissance de l'environnement), nous avons choisi une implémentation robuste d'un détecteur de points d'intérêt en se basant sur une représentation multi-échelles des images. Celle-ci nous permet, via un algorithme d'appariement pyramidal, d'effectuer un classement des points reposant sur deux critères : une bonne robustesse aux bruits et une bonne localisation. Il est possible alors d'effectuer l'appariement de ces points entre différentes images et de construire ainsi une modèle projectif de la scène, par des méthodes robustes classiques.

Cependant, dans notre problématique, il n'est pas rare d'avoir des images de piètre qualité et les algorithmes de traitement d'images sont mis en échec. Nous proposons alors d'appliquer un pré-traitement qui, couplé à notre approche multi-échelles, permet d'obtenir de bons résultats. Enfin, ce travail a donné lieu au développement d'un outil logiciel permettant aux utilisateurs, spécialistes ou non du domaine, de manipuler des techniques avancées de traitement d'image.

Mots-clés : vision par ordinateur, reconstruction projective, points d'intérêt, approche multi-échelles, appariement robuste, images sous-marines

3D Metrology using active vision with natural underwater objects

This PhD Thesis concerns the application of computer vision techniques to natural underwater images. Recently, advances in projective geometry have given a strong formalism to computer vision reconstruction algorithms and has allowed real improvements. Nevertheless, many algorithms may have problems with natural scenes.

We present here a complete methodology to make a projective reconstruction of natural scenes from underwater images taken with one uncalibrated camera. In the first step, we are interested in extracting robust features which is a necessary step of image processing. Due to the particular scenes we observe (no geometric simple forms, high noise, no knowledge of the environment), we choose a robust implementation of a point detector based on a multi-scale representation of the images. This one allows us to classify points depending on two criteria : robustness against noise and good localization. It is then possible to match these points between images and compute the projective model of the scene with standard robust methods.

In our case, we often have really noisy images and standard algorithms cannot be efficient. We propose to apply a preliminary data processing, which used with our multi-scale approach gives good results. Finally, this work has permitted to develop a software for users, experimented or not, to use advanced techniques for computer vision.

Keywords : computer vision, projective reconstruction, interest points, multi-scale approach, robust matching, underwater images

Thèse préparée à l'INRIA Sophia Antipolis dans le projet ICARE