Université Lumière Lyon 2
École Doctorale Informatique et Mathématiques de Lyon

Numéro : |_|_|_|_|_|_|_|_|_|

THESE
pour obtenir le grade de
Docteur en Informatique

présentée par

Lionel Robinault

Mosaïque d'images multi résolution et applications

Soutenue le 08 Septembre 2009

JURY

Rapporteurs	Michèle Rombaut	Professeur des Universités	GIPSA lab, Grenoble
	Nicole Vincent	Professeur des Universités	Crip5, Paris
Examinateurs	Thierry Chateau	Maître de conférence	LASMEA, Clermont-Ferrand
Président	Patrick Perez	Directeur de recherche INRIA	IRISA, Rennes
Directeurs	Serge Miguet	Professeur des Universités	Liris, Lyon
	Stéphane Bres	Maître de conférence	Liris, Lyon
Examinateur	Jean-Baptiste Ducatez	Ingénieur	Foxstream, Lyon

Résumé

Le travail de thèse que nous présentons s'articule autour de l'utilisation de caméras motorisées à trois degrés de liberté, également appelées caméras PTZ. Ces caméras peuvent être pilotées suivant deux angles. L'angle de panorama (θ) permet une rotation autour d'un axe vertical et l'angle de tangage (φ) permet une rotation autour d'un axe horizontal. Si, théoriquement, ces caméras permettent donc une vue omnidirectionnelle, elles limitent le plus souvent la rotation suivant l'angle de panorama mais surtout suivant l'angle de tangage. En plus du pilotage des rotations, ces caméras permettent également de contrôler la distance focale permettant ainsi un degré de liberté supplémentaire. Par rapport à d'autres modèles, les caméras PTZ permettent de construire un panorama - représentation étendue d'une scène construite à partir d'une collection d'image - de très grande résolution. La première étape dans la construction d'un panorama est l'acquisition des différentes prises de vue. A cet effet, nous avons réalisé une étude théorique permettant une couverture optimale de la sphère à partir de surfaces rectangulaires en limitant les zones de recouvrement. Cette étude nous permet de calculer une trajectoire optimale de la caméra et de limiter le nombre de prises de vues nécessaires à la représentation de la scène. Nous proposons également différents traitements permettant d'améliorer sensiblement le rendu et de corriger la plupart des défauts liés à l'assemblage d'une collection d'images acquises avec des paramètres de prises de vue différents. Une part importante de notre travail a été consacrée au recalage automatique d'images en temps réel, c'est à dire que chaque étapes est effectuée en moins de 40ms pour permettre le traitement de 25 images par seconde. La technologie que nous avons développée permet d'obtenir un recalage particulièrement précis avec un temps d'exécution de l'ordre de 4ms (AMD1.8MHz). Enfin, nous proposons deux applications de suivi d'objets en mouvement directement issues de nos travaux de recherche. La première associe une caméra PTZ à un miroir sphérique. L'association de ces deux éléments permet de détecter tout objet en mouvement dans la scène puis de se focaliser sur l'un d'eux. Dans le cadre de cette application, nous proposons un algorithme de calibrage automatique de l'ensemble caméra et miroir. La deuxième application n'exploite que la caméra PTZ et permet la segmentation et le suivi des objets dans la scène pendant le mouvement de la caméra. Par rapport aux applications classiques de suivi de cible en mouvement avec une caméra PTZ, notre approche se différencie par le fait que réalisons une segmentation fine des objets permettant leur classification.

Mots clés : analyse vidéo, panorama, mosaïque d'image, recalage, segmentation, suivi

Abstract

The thesis considers the of use motorized cameras with 3 degrees of freedom which are commonly called PTZ cameras. The orientation of such cameras is controlled according to two angles: the panorama angle (θ) describes the degree of rotation around on vertical axis and the tilt angle (φ) refers to rotation along a meridian line. Theoretically, these cameras can cover an omnidirectional field of vision of 4psr. Generally, the panorama angle and especially the tilt angle are limited for such cameras. In addition to control of the orientation of the camera, it is also possible to control focal distance, thus allowing an additional degree of freedom. Compared to other material, PTZ cameras thus allow one to build a panorama of very high resolution. A panorama is a wide representation of a scene built starting from a collection of images. The first stage in the construction of a panorama is the acquisition of the various images. To this end, we made a theoretical study to determine the optimal paving of the sphere with rectangular surfaces to minimize the number of zones of recovery. This study enables us to calculate an optimal trajectory of the camera and to limit the number of images necessary to the representation of the scene. We also propose various processing techniques which appreciably improve the rendering of the mosaic image and correct the majority of the defaults related to the assembly of a collection of images which were acquired with differing image capture parameters. A significant part of our work was used to the automatic image registration in real time, i.e. lower than 40ms. The technology that we developed makes it possible to obtain a particularly precise image registration with an computation time about 4ms (AMD1.8MHz). Our research leads directly to two proposed applications for the tracking of moving objects. The first involves the use of a PTZ camera and a spherical mirror. The combination of these two elements makes it possible to detect any motion object in the scene and to then to focus itself on one of them. Within the framework of this application, we propose an automatic algorithm of calibration of the system. The second application exploits only PTZ camera and allows the segmentation and the tracking of the objects in the scene during the movement of the camera. Compared to the traditional applications of motion detection with a PTZ camera, our approach is different by the fact that it compute a precise segmentation of the objects allowing their classification.

keywords : video analysis, panorama, image mosaicing, registration, segmentation, tracking

Table des matières

Table des illustrations

Liste des tableaux

Mosaïque d'images multi-résolution et applications

Chapitre 1 : Introduction

1. Introduction

1.1. Présentation

Il est manifeste aujourd'hui que la surveillance des lieux et des personnes se généralise actuellement, que ce soit dans des zones du domaine public ou chez des particuliers. Si cette tendance se poursuit, il est bien évident que les besoins d'une analyse automatique des séquences vidéo obtenues seront de plus en plus importants.
Dans un tel contexte, on peut distinguer les analyses qui se font sur la séquence vidéo une fois que celle-ci a été acquise (analyse en temps différé), et les analyses réalisées au moment de l'acquisition, (analyse temps réel). Ces deux types d'analyse visent des applications sensiblement différentes : ainsi, si l'objectif de la surveillance est le déclenchement d'une alarme de sécurité, la nécessité d'une analyse temps réel est évidente. L'exemple le plus commun est celui de la détection d'effractions bien sûr, mais on pourrait aussi citer la détection de malaises chez les personnes âgées par exemple.
D'un point de vue pratique, les deux types d'analyse se distinguent aussi par les contraintes qui y sont attachées : Une analyse en temps différé possède plus de latitude en ce qui concerne le temps du traitement mis en jeu. En revanche, la séquence analysée est figée sur le support d'enregistrement sans adaptation ou interaction possible. Une analyse en temps réel a des contraintes fortes en ce qui concerne la durée d'exécution des algorithmes utilisés. En revanche, elle présente l'avantage de pouvoir interagir sur la prise de vue et donc de piloter la séquence vidéo à l'instant suivant, en adaptant le niveau de zoom ou l'angle de vue, en fonction des résultats de l'analyse de l'instant précédent. Cette possibilité de rétroaction est une des grandes forces de l'analyse temps réel pour permettre une amélioration de la qualité des résultats obtenus, quelles que soient les applications. L'utilisation d'une caméra « Pan-Tilt-Zoom » (PTZ) permet de justement de contrôler interactivement la prise de vue. Le pilotage de la caméra suivant deux axes de rotation permet de sécuriser un champ de vue étendu avec une résolution importante et permet de générer un panorama plus ou moins complet de la scène.

Dans ce travail de thèse, que nous avons réalisé au sein de la société Foxstream et du laboratoire Liris, nous allons aborder les différentes étapes qui entrent dans le processus de réalisation et d'exploitation d'une image panoramique en temps réel : l'acquisition, la projection des prises de vue dans l'image panoramique, l'amélioration du rendu, la visualisation et les applications de détection et de suivi des objets en mouvement dans une scène. Comme précisé dans [GLE03] une image panoramique est une image grand angle visualisant l'environnement autour de l'appareil photo. Habituellement, l'image panoramique parcourt 360° dans un plan horizontal autour de l'appareil photo ou de la caméra et approximativement 120° sur le plan vertical voir 180° pour une vue complète. Un exemple de représentation panoramique est donné ci après.

Figure 1 : Exemple de représentation panoramique (360° suivant le plan horizontal et 120° sur le plan vertical)

L'utilisation des images panoramiques est largement répandue dans des domaines telles que la robotique, la vision par ordinateur, la surveillance et la réalité virtuelle. Elles sont également utilisées dans des applications commerciales comme le divertissement, la TV interactive, l'immobilier et le tourisme virtuel. Ces 15 dernières années, les systèmes panoramiques de formation d'image ont sensiblement progressé. Non seulement les professionnels peuvent créer et afficher des panoramas, mais il existe une grande quantité de logiciels disponibles, dont certains gratuits[1]. Chacun d'entre nous, avec un ordinateur et simple appareil photo numérique, peut créer des panoramas. Même les grandes compagnies sont présentes sur ce marché comme Apple avec son logiciel QuickTime[2] ou Canon avec le logiciel PhotoStitch[3] qui permettent un accès facile à la création d'image panoramique. Il existe également plusieurs méthodes pour capturer des panoramas. La solution la plus simple consiste à utiliser un appareil photo classique monté sur un trépied et en faisant tourner manuellement l'appareil photo autour de son centre optique. Les professionnels auront plutôt tendance à utiliser une caméra motorisée de type PTZ (Pan, Tilt, Zoom). Le principe est exactement le même, à la différence que les rotations de la caméra autour de son centre optique sont pilotées et contrôlées par un ordinateur. Ceci permet de déterminer avec plus de précision les conditions de la prise de vue. L'un des inconvénients de cette approche, qu'elle soit manuelle ou pilotée, est qu'il faut du temps pour parcourir l'ensemble de la scène. La représentation panoramique ne peut pas être réalisée en temps réel. A un instant donné, seule une fraction de la scène est vue par la caméra. Des solutions existent pour capturer l'ensemble de la scène en temps réel. La première consiste à utiliser des équipements photographiques appelés omnidirectionnels. Ils peuvent être dioptriques ou catadioptriques. Les équipements dioptriques sont composés d'un capteur d'image et d'éléments simplement réflectifs (objectif grand angle ou fisheyes par exemple).

[1] http://user.cs.tu-berlin.de/~nowozin/autopano-sift/
http://autostitch.softonic.fr/
http://www.easypano.com/
[2] http://www.apple.com/fr/quicktime/
[3] http://www.canon.fr/

Figure 2 : Objectif Fisheye NIKON FC-E8 Focale 0.21, angle de vue 183°

Dans le cas des équipements catadioptriques, des éléments réfractifs (miroir sphérique ou parabolique) vont être ajoutés. Cette première solution permet effectivement de capturer l'ensemble de la scène en temps réel. Cependant, la résolution obtenue est généralement faible, même en utilisant un capteur à haute résolution.

Figure 3 : Capture d'une image panoramique en utilisant un miroir sphérique

Figure 4 : Projection cylindrique de la scène

Afin d'augmenter la résolution, une autre solution consiste à utiliser plusieurs caméras placées au même endroit mais dirigées dans différentes directions. Ce cas particulier permet de considérer, en première approximation, que le centre optique est unique pour l'ensemble des caméras. La solution plus générale est de placer les différentes caméras dans des endroits différents. Cette technique permet de reconstruire une image 3D de la scène. L'aspect 3D ne sera que partiellement étudié dans ce travail de thèse. Nous nous sommes restreints au cas où le centre optique de chaque prise de vue est unique (ou considéré comme tel). Nous en verrons les raisons par la suite. Dans le cas particulier où le centre optique est unique, la

reconstruction d'image panoramique obéit aux même règles mathématiques que la première solution que nous avons évoquée, c'est à dire une caméra montée sur un trépied ou une caméra motorisée dite PTZ. La différence entre les deux modèles de caméras précités, est bien sur le coût de l'installation mais elle permet de visualiser l'ensemble de la scène avec une meilleure résolution que les caméras omnidirectionnelles.

La construction et l'exploitation d'une image panoramique nécessitent 4 grandes étapes : l'acquisition, la projection des prises de vue dans l'image panoramique, l'amélioration du rendu et la visualisation. Nous allons présenter rapidement ces quatre points.

1.2. Acquisition

La formation d'image panoramique est un exercice particulier qui requiert un nombre d'images différent selon la technologie employée et le but final de l'application. Dans tous les cas, la méthode qui doit être utilisée pour la construction de l'image panoramique est fonction des paramètres suivants : résolution, couverture visuelle et temps d'acquisition. L'utilisation de caméra omnidirectionnelle est fréquemment utilisée dans la robotique. Le but ici est d'obtenir une vue la plus large possible dans le périmètre immédiat du capteur. Les caméras omnidirectionnelles sont à privilégier lorsque le temps d'acquisition est inférieur à la seconde et que la résolution ou plus précisément la profondeur de champ ne sont pas très importants. Avec ces modèles de caméras, sur le plan horizontal, le champ couvert peut aller sans grande difficulté jusqu'à 360°. Par contre, sur le plan vertical, le champ couvert dépasse plus rarement 90°. Avec une caméra unique en rotation autour de son centre optique le champ couvert peut aller jusqu'à 360° sur le plan horizontal et 180° sur le plan vertical. Il dépend essentiellement de la conception mécanique de la mise en rotation de la caméra. La résolution peut être très importante en fonction de la distance focale de l'objectif. Par contre le temps d'acquisition est alors très long et dépend finalement du nombre d'images nécessaires pour une couverture optimale de la scène à observer. Une solution intermédiaire consiste à utiliser ce que l'on appelle traditionnellement un « bouquet » de caméra. Il s'agit d'un ensemble de caméras installées autour d'un axe. Le bouquet de caméra est un compromis intéressant puisqu'il permet d'augmenter la résolution tout en gardant l'aspect temps réel de l'acquisition. Même si le champ couvert peut être de 360° sur le plan horizontal et 180° sur le plan vertical, des contraintes mécaniques limitent généralement celui ci. De plus cette solution est relativement coûteuse. Outre le nombre plus important de caméras, cette solution nécessite des ressources en matériels plus importantes (carte d'acquisition, processeur, mémoire …). Le système GeoView 3000[4] de la société iMove utilise 6 appareils photo, quatre sur le plan horizontal, un vers le bas et un vers le haut.

1.3. Traitement de l'image

Les images capturées par les dispositifs panoramiques vont devoir subir un certain nombre de traitements avant de pouvoir être exploitées. Dans le cas des dispositifs omnidirectionnels, l'image est très déformée. Dans les autres dispositifs, une image ne contient qu'une partie de la scène. La représentation complète de la scène nécessite donc plusieurs images qui vont être projetées sur une image panoramique. Les images ne sont donc pas forcement acquises au même moment ou avec le même capteur. Si les informations de prise de vue ne sont pas précises ou si elles ne sont pas connues, une étape de mise en correspondance sera également nécessaire. Dans le cas où une seule caméra est utilisée, les différentes images formant le panorama ne sont pas prises en même temps. Les conditions

[4] http://www.imoveinc.com

d'illumination peuvent être différentes. De même, la caméra peut disposer d'une correction automatique la luminosité de façon à ajuster le gain du capteur. Les images devrons alors subir un traitement pour supprimer les effets de couture[5]. Ce cas se retrouve également avec les bouquets de caméra. Pour simplifier, les traitements sont : étalonnage du dispositif, correction des aberrations de l'objectif, réduction du bruit, recalage des images, correction du gain et projection. Dans le travail de thèse que nous présentons, nous n'aborderons pas la phase d'étalonnage et de correction des aberrations de l'objectif de notre matériel. Nous avons repris des technique classiques décrites dans la littérature [FAU93, LUO97, BEN01, HAR04, REM06]. Nous allons par contre étudier longuement la phase de mise en correspondance, la correction du gain et la projection.

1.3.1. Mise en correspondance

Une large part de notre travail de thèse a été consacré à ce problème de mise en correspondance. Initialement, nous n'avions pas prévu d'aborder ce sujet. Dans notre cas, le modèle de transformation mathématique était clairement identifié et les paramètres de prise de vue donnés par notre caméra devaient nous permettre de calculer cette transformation. Cependant, la précision de ces paramètres ne s'est pas révélée suffisante et ne nous a pas permis de réaliser des panoramas robustes.

Figures 5 : Image de gauche : fusion de deux images avec uniquement les informations de la caméra ; Image de droite : fusion des deux images après recalage.

La mise en correspondance est un problème difficile qui continue à mobiliser la communauté scientifique. Plusieurs solutions sont proposées dans la littérature et de nombreux articles proposent une synthèse de ces méthodes [BRO92,SAR00,ZIT05]. Le principe de base consiste à mettre en correspondance des zones de l'image en fonction de leurs propriétés radiométriques ou géométrique. Les méthodes sont généralement classées en deux catégories : denses ou éparses. Les méthodes denses cherchent à estimer les paramètres de transformation en exploitant l'intensité de l'ensemble des pixels contenu dans la zone de recouvrement. Les méthodes éparses n'utilisent pas tous les points mais seulement quelques points particuliers (coins, extrema locaux) ou des primitives géométriques (lignes, cercles…). Dans le cas où il n'y a pas de déformation des images, les méthodes denses sont réputées plus précises, mais elles sont souvent moins rapides, moins robustes aux changements de luminosité et à la présence d'objet en mouvements.

[5] couture : variation brusque de la luminosité le long de la frontière entre des images adjacentes

1.3.2. Correction du gain

Le contrôle de la luminosité n'est pas un problème aussi simple qu'il y paraît. Si la luminosité est relativement constante dans toutes les directions alors il suffit de fixer manuellement le gain de la caméra pendant toute la durée de la prise de vue. Par contre, il ne faut pas que cette luminosité évolue pendant cette prise de vue (lors du passage d'un nuage par exemple). Cependant, même si la luminosité générale n'évolue pas au cours de la prise de vue, si les écarts de luminosité dans la scène sont trop important, les zones sombres apparaîtront sous-exposées dans la mosaïque d'image. A l'inverse, d'autres zones beaucoup plus lumineuses risquent de se trouver sur-exposées. La solution pourrait être de laisser la caméra gérer automatiquement le gain en fonction de la luminosité de la portion de scène visée. Cette solution offre bien sûr l'avantage que chaque image est acquise en optimisant la dynamique du capteur. Le problème est qu'une même portion de la scène prise avec deux gains différents n'a plus la même valeur de luminosité. L'exemple suivant est un cas d'école. La caméra est placée à l'intérieur d'une pièce. Cette pièce est éclairée par une lumière artificielle et par une lumière naturelle à travers une fenêtre. La luminosité à l'intérieur est faible par rapport à celle de l'extérieur. Si le gain de la caméra est fixé manuellement pour obtenir une luminosité correcte à l'intérieur de la pièce, la fenêtre apparaîtra sur-exposée et il ne sera pas possible de visualiser l'extérieur. Dans le cas contraire, si le gain est piloté automatiquement par la caméra, alors l'extérieur devient visible, mais le phénomène de « couture » apparaît.

Figures 6 :Image de gauche : Gain fixe, l'extérieur apparaît très largement saturée. Image de droite : Gain automatique, l'extérieur est visible mais on observe des « coutures »

1.4. Assemblage

La mise en correspondance des images est utilisée pour déterminer avec la meilleure précision possible les paramètres de la prise de vue. Cette information permet de projeter correctement l'image dans le panorama. Dans le cas général, une homographie est calculée à partir de surfaces planes contenues dans la zone de recouvrement des deux images [KAN99,SUG04]. Nous verrons que dans le cas particulier des caméras PTZ où le centre de projection est confondu avec le centre des rotations, tous les points de la zone de recouvrement peuvent être utilisés pour calculer l'homographie, même s'ils n'appartiennent pas à des surfaces planes.
Avant toute chose, nous devons définir le modèle de représentation que nous allons utiliser. Lorsqu'il s'agit de fusionner quelques images de façon à obtenir une image globale plus large, il suffit de prendre une image de référence et de projeter toutes les images dans le plan cette image de référence. Lorsque l'angle solide devient trop important, les déformations inhérentes à la projection sont importantes. Au delà d'une certaine limite, l'homographie ne peut plus être calculée où du moins ne permet pas une projection conforme. Il est donc

nécessaire d'utiliser d'autres modes de projection. Nous classons ces projections en deux catégories. Les projections planes et la projection sur polyèdre. Les projections planes permettent de visualiser l'ensemble de la scène sur une seule image mais elles entraînent des déformations importantes. Parmi les projections planes, la plus utilisée est la projection cylindrique.

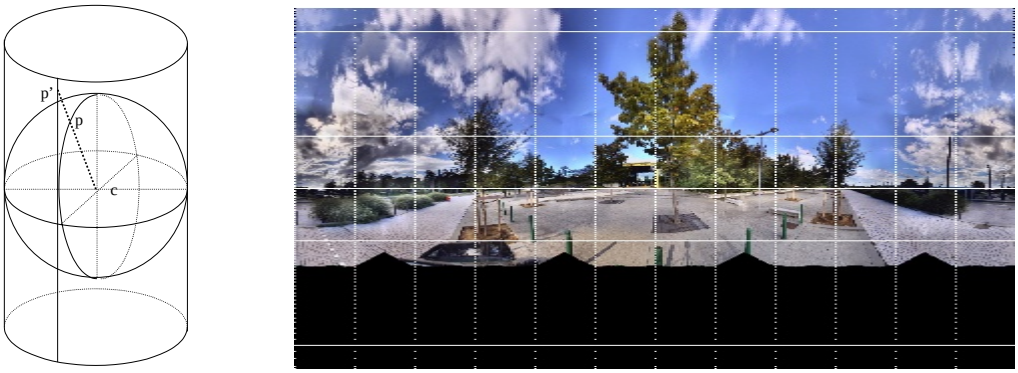


Figure 7 : Principe de la projection cylindrique avec une visualisation par développement

Un autre mode de représentation est la projection de la scène sur un polyèdre. Ces projections ne permettent pas la représentation de la scène en une seule image mais elles limitent les déformations ainsi que le coût du stockage. Le principe est de projeter les images sur les faces du polyèdre. Un autre avantage de ce type de représentation est que la projection d'une image sur une face, se calcule à partir d'une homographie. Habituellement, le polyèdre utilisé est le cube. Nous lui préférons l'icosaèdre tronqué pour deux raisons. La première est qu'à résolution identique, l'espace mémoire est moins important dans le cas de l'icosaèdre tronqué. Le deuxième est que les angles sont moins aigus (plus « plats ») que sur le cube, ce qui limite les défauts lors du rendu.

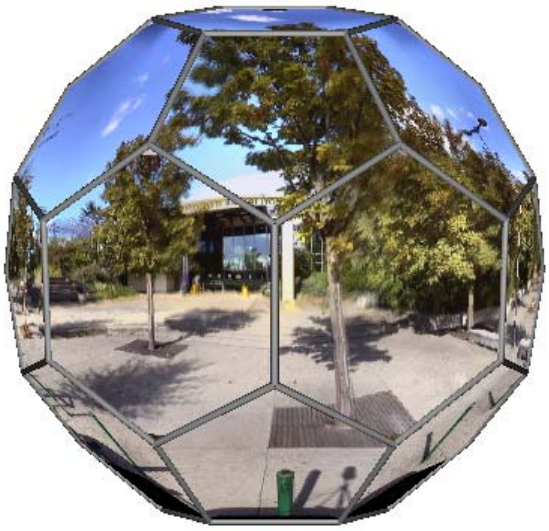


Figure 8 : Projection sur un icosaèdre tronqué

1.5. Immersion et Visualisation

La visualisation est souvent l'étape ultime du mosaïquage. Bien que les techniques de base soient connues depuis très longtemps, ce n'est que récemment que les mosaïques d'images servent à autre chose qu'à la visualisation. Cependant, cela reste pour beaucoup, le seul intérêt de ces constructions. Comme nous venons de le voir, la projection d'une image sur un cylindre permet de visualiser l'image à partir d'une seule vue avec une limitation au niveau des pôles mais surtout en déformant la scène. Une fois le mosaïquage réalisé, plusieurs logiciels permettent de visualiser une portion de la scène en redressant l'image dans les mêmes conditions que si elle avait été acquise par la caméra. Il est alors possible de se déplacer dans la scène et d'effectuer des changements de focale afin d'augmenter ou diminuer l'angle de visualisation. Le logiciel le plus répandu est QuickTimeVr. En interne, il utilise une structure de donnée basée sur la projection sur un cube. S'il est relativement facile de réaliser un fichier compatible avec QuickTimeVr, l'étape la plus délicate est de construire correctement les faces du cube en supprimant tous les défauts que nous avons évoqués précédemment. L'inconvénient de la plupart des logiciels est qu'ils ne gèrent pas l'aspect multi-résolution ou alors indirectement. Ce que nous entendons par « gestion de la multi-résolution » est la possibilité de ne pas stocker l'ensemble du panorama avec la même résolution. En effet, l'ensemble de la scène n'est pas forcément d'un intérêt égal. Lorsque nous réalisons un panorama, nous aimerions avoir plus de détail sur certaines zones de l'image et moins sur d'autre (le ciel par exemple). Avec la plupart des logiciels disponibles, c'est le niveau de détail le plus fin qui conditionne la résolution du panorama tout entier. Le temps de calcul et surtout la taille du panorama ne sont pas optimisé. Cela peut poser un problème lorsque l'on transmet un panorama sur le réseau. C'est la raison pour laquelle, certains logiciels gèrent la multi-résolution en utilisant plusieurs panoramas complets de la même scène avec des niveaux de résolutions différents. Lors d'une transmission sur Internet, cette solution permet rapidement d'obtenir une première représentation de la scène à basse résolution de façon à ce que l'utilisateur puisse naviguer. Les résolutions plus élevées arrivant ensuite au gré du débit de la connexion, mais uniquement pour les zones observées.

1.6. Caméras et Applications

Pendant longtemps, les caméras IP ont eu une mauvaise réputation. Il est vrai que sur les premières caméras IP la qualité de la vidéo n'avait rien à voir avec les caméras analogiques. Les progrès réalisés sur la taille des capteurs, la qualité de la compression et la bande passante ont grandement amélioré le rendu des caméras IP. Ces dernières années ont vu l'essor d'une nouvelle classe de caméras IP, les caméras PTZ. Des caméras PTZ analogiques existent depuis presque aussi longtemps que les caméras analogiques fixes, mais la baisse des coûts de fabrication à permis une démocratisation de ces caméras. Du coup, les applications se sont multipliées. L'un des objectifs de ce travail de thèse, en accord avec la société Foxstream, est d'étudier les applications de détection et de suivi automatique des objets en mouvement dans une scène à partir d'une caméra PTZ. L'intérêt étant de sécuriser une zone étendue en limitant le nombre de caméras.

Le matériel que nous avons utilisé au cours de ce travail de thèse est une caméra PTZ Sony SNC-RZ25P. Cette caméra motorisée est particulièrement bien adaptée à notre application puisqu'elle dispose d'une tourelle et d'un zoom optique pouvant aller jusqu'à un grossissement x18. C'est de plus une caméra IP équipée d'un capteur CCD Sony ExwaveHAD et de la fonction jour/nuit.

Figure 9 : Caméra Sony RZ25P

Le serveur Web et l'interface réseau intégrée à la caméra permettent une prise en main et une configuration rapide. Cependant, pour notre application, nous avons développé notre propre interface de pilotage et de visualisation que nous ne détaillerons pas ici.

Les principales caractéristiques de cette caméra sont :

- Angle panoramique : –170° à +170°
- Angle de tangage : -30° à +90°
- Zoom optique : x 18
- Distance focale : 4,1 mm à 73,8mm
- Nombre d'ouverture : 1,4 à 3.0
- Capteur : CCD ExwaweHAD type ¼
  444 000 pixels (752 H x 582 V)
- Sensibilité : 0,07 lux en couleur et jusqu'à 0,01 lux en NB
- Résolution image: jusqu'à 640 x 480
- Compression image : JPEG
- Compression flux : MJPEG ou MPEG4

1.7. Organisation du document

Nous avons scindé notre travail de recherche en 4 grandes étapes qui font l'objet des chapitres 2 à 5. Le chapitre 2 est consacré aux différentes représentations possibles des mosaïques d'image. Dans ce chapitre nous nous placerons dans le cas idéal où la caméra correspond exactement au modèle et où il n'y a aucune distorsion de quelque nature que ce soit. Bien évidemment, nous serons rapidement confrontés aux limites de ce cas idéal. Dans ce chapitre, nous introduisons également les principes mathématiques de base permettant de calculer une mosaïque d'images. Une description plus complète est donnée en annexe. Enfin, dans ce chapitre nous présentons une étude théorique permettant un pavage optimal de la sphère à partir de surfaces rectangulaires en limitant les zones de recouvrement. Cette étude nous permet de calculer une trajectoire optimale de la caméra et de limiter le nombre de prises de vues nécessaire à la représentation de la scène. Ce travail a fait l'objet d'une publication dans les actes de la conférence ICIAP 2007 qui a eu lieu à Modène en Italie.
Dans le chapitre 3, nous allons aborder les principales distorsions que nous rencontrons lors de la création d'une mosaïque d'image. Le but de ce chapitre est double. Le premier est de vérifier l'écart des caméras par rapport au modèle que nous utilisons et surtout d'en définir

les limites de validité. Le deuxième est de proposer des outils permettant de réaliser des panoramas robustes en temps réel. La contrainte forte du temps réel, nous oblige à faire des choix entre la qualité de la construction et la rapidité de sa mise en œuvre. Nous évoquons essentiellement les problèmes d'alignement photométrique et la suppression des fantômes.
Le chapitre 4 est consacré lui aussi à l'amélioration de la qualité du panorama. Cependant, ce chapitre se focalise exclusivement sur le problème de recalage d'images auquel nous sommes invariablement confrontés lorsque les paramètres de prise de vue ne sont pas disponibles ou ne sont pas suffisamment précis. Une bonne part de notre travail de recherche s'est attaché à résoudre ce problème en temps réel. Le recalage d'images n'est pas un problème récent et plusieurs solutions ont déjà été apportées. Cependant, les expérimentations que nous avons effectuées ne se sont pas révélées concluantes en termes de qualité du recalage mais surtout en temps de calcul. Pour les applications que nous envisageons, le recalage d'image n'est qu'une étape intermédiaire. L'ensemble du processus, recalage compris, devant s'exécuter en temps réel. Dans ce chapitre, nous présentons tout d'abord une étude des principales mesures de similarité utilisées dans le recalage d'image afin de déterminer la mesure la mieux adaptée à nos besoins. Ensuite nous passons en revues les principales méthodes de recalage denses et éparses décrites dans la littérature. Nous proposons également notre propre méthode de recalage que nous comparons aux solutions existantes.
Le chapitre 5 est consacré aux applications que nous avons étudiées. Nous présentons une application de détection d'objets en mouvement, mettant en collaboration une caméra PTZ et un miroir sphérique. Nous proposons une méthode de calibration automatique de l'ensemble composé de la caméra et du miroir. Un article décrivant l'étape de calibration a été accepté à la conférence AVSS 2009 et sera présenté à Gènes en septembre 2009. Nous présentons également une modélisation originale du fond adaptée au cas des caméras PTZ. Cette modélisation permet de réaliser une segmentation fine des objets en mouvement, rendant possible leur suivi et leur classification en temps réel. Ce travail de recherche a également fait l'objet d'une publication dans les actes de la conférence VISAPP 2009 qui s'est déroulée à Lisbonne.

Mosaïque d'images multi-résolution et applications

Chapitre 2 : Mosaïque d'images

2. Mosaïque d'images

2.1. Présentation

Une mosaïque d'images est une collection d'images prises suivant des angles de vue différents et ramenées à un même repère. Un panorama est une représentation d'une mosaïque d'images, permettant de voir l'intégralité d'une scène à 360° voir à 4π stéradians. Le terme panorama est dérivée du grec est signifie « tout-voir ». L'un des premiers panoramas répertoriés a été réalisé par Robert Barker. Il fit d'ailleurs breveter en 1787 un dispositif qu'il nomma «La nature d'un coup d'œil» où les spectateurs, placés sur une estrade, se trouvent au cœur d'un paysage ou d'un champ de bataille.

En décembre 1900, Louis Lumière dépose le brevet du Photorama. Il s'agit d'un procédé photographique permettant de prendre, en une seule prise de vue, une scène sur 360°. Le brevet inclut la projection intégrale de ce panorama sur un cylindre. Les spectateurs sont, là aussi, placés au centre de l'estrade.

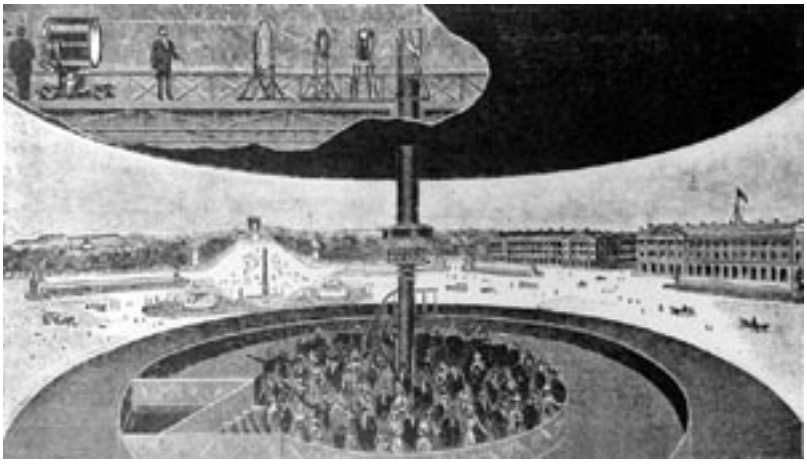


Figure 10 :Vue en coupe de la salle du photorama lumière.

Crédit photo : http://www.institut-lumiere.org

Les développements des techniques de visualisation de ces dix dernières années et notamment les travaux de Chen [CHE95] et de Szeliski [SZE94] ont permis la création et l'exploitation de panoramas virtuels à partir d'ordinateurs personnels. Il existe aujourd'hui plusieurs logiciels commerciaux permettant de créer un panorama à partir de simples photos en quelques clics de souris. Les utilisateurs de matériel Canon connaissent le logiciel

PhotoStitch. Mais il existe bien d'autres logiciels disponibles sur internet comme PanoramaFactory, PanoramaComposer, AutoStitch ou PanoramaTools pour les utilisateurs avancés.

La création d'un panorama ou d'une mosaïque d'images de façon automatique et en contrôlant éventuellement les conditions de prise de vue est un défi qui motive les industriels comme le monde de la recherche. Comme il est précisé dans [TAN04], nous pouvons classer les méthodes de construction d'une mosaïque d'images en deux catégories : dioptrique et catadioptrique. Dans les méthodes dioptriques seuls des éléments réfractifs (lentilles) sont utilisés. Dans les méthodes catadioptriques, des éléments réflectifs (miroirs) vont être ajoutés. Parmi les méthodes dioptriques, nous allons retrouver les ensembles de caméras (bouquet), les lentilles panoramiques [NAY97], les lentilles « fish-eyes » [XIO97], les caméras linéaires [DOU03] ou plus classiquement les caméras rotatives dont, entre autres, les caméras PTZ. Dans les méthodes catadioptriques, nous trouvons généralement une caméra associée à un miroir conique [YAG90], sphérique [HON91], parabolique [STU02] ou a double courbure [SOU96, FIA02], ou alors plusieurs caméras associées à des miroirs plans [TAN04].

Les applications sont diverses. Chen [CHE95] est l'un des premiers à utiliser un panorama cylindrique à 360° pour représenter une scène et permettre la navigation en temps réel dans une application de réalité virtuelle. Son approche a été utilisée dans le produit commercial QuickTimeVR.

Pour Lee [LEE99], la construction du panorama sert de support à la compression et à la transmission d'un flux vidéo. Le panorama correspondant aux composantes statiques de la scène est transmis une fois. Ensuite, les objets en mouvement sont extraits de la vidéo et transmis séparément. La première étape de leur approche consiste à créer le panorama en projetant les images sur un cylindre. Les auteurs utilisent pour cela le logiciel PhotoVista™. Une importante limitation de leur approche est que dans cette première étape, les images acquises ne doivent comporter que les composantes statiques du fond. Une fois cette étape réalisée, ils utilisent la mosaïque d'image pour segmenter les objets en mouvement. Lors de cette deuxième phase, les auteurs doivent estimer la position de la camera de façon à reconstruire l'image de fond utilisée pour la segmentation. Pour réaliser ce calcul, leur méthode nécessite de sélectionner manuellement et préalablement dans le panorama des petits blocs carrés qui ne devront pas être occultés pendant le mouvement. Ces blocs sont alors recherchés dans l'image courante.

Hsu et Anandan [HSU96] décrivent ce qu'ils appellent la compression basée sur la mosaïque d'image (MBC pour Mosaic-Based Compression) et décrivent plusieurs types de représentations permettant d'éviter la redondance d'informations dans les données vidéo.

Douze et al. [DOU02, DOU03] proposent une méthode de suivi robuste dans une séquence vidéo en utilisant un panorama. L'originalité de leur approche repose sur l'utilisation d'une caméra linéaire de leur fabrication. Cette caméra leur permet rapidement, et sans calcul particulier, de construire le panorama. Une caméra classique est ensuite utilisée pour le suivi. Les auteurs proposent une solution de suivi à base de point contrôle et d'un modèle homographique.

Kang et al [KAN03] utilisent une caméra PTZ pour réaliser le suivi des personnes. La première étape de leur algorithme consiste à construire un panorama sans les objets en mouvement et à sélectionner des points statiques de la scène de façon à ce que, quel que soit le déplacement de la caméra et les objets en mouvement dans la scène, ils puissent en

retrouver au moins quatre dans l'image et ainsi calculer correctement l'homographie. Dans la phase d'exploitation, ils recherchent les points clés dans l'image, calculent l'homographie et par simple soustraction de l'image courante et de l'image du fond, ils déterminent les pixels en mouvement. Bartoli et all. [BAR03] utilisent une technique similaire pour créer le panorama du mouvement d'une séquence d'images. Le principe réside dans la construction d'une couche statique et d'une couche dynamique et ceci en deux étapes. La première étape consiste à créer un modèle de fond sous la forme d'un panorama. Ce fond ne contient alors que les composantes statiques de la scène. Dans une deuxième phase, chaque image de la séquence est projetée dans le panorama statique et comparée avec le modèle de fond de façon à extraire les objets en mouvement.

D'autres applications sont plus marginales et sortent du cadre de la visualisation. Peer et al. [PEE02] présentent un système permettant de déterminer la carte de profondeur d'une image panoramique. Ils utilisent pour cela une caméra placée sur un bras tournant. En raison du décalage entre le centre optique de la caméra et l'axe de rotation du bras, les auteurs peuvent ainsi estimer l'effet de parallaxe sur deux images prises suivant deux angles différents. Ceci rend donc possible la reconstruction stéréo. Pour Sato [SAT04] la mosaïque d'images peut servir de support à l'analyse de document.

Si le centre optique de la caméra est confondu avec les axes de rotations, le monde vu par la caméra peut être considéré comme une sphère dont le centre est justement le centre optique. Si deux points sont alignés avec le centre de rotation de la caméra et que le centre optique est confondu avec le centre de rotation, quel que soit l'angle de rotation que l'on fait subir à la caméra les deux points restent alignés avec le centre optique puisque ce dernier ne se déplace pas. Ceci implique que quel que soit l'angle de rotation, un seul rayon passe par ces deux points et que donc ces deux points sont projetés sur un point unique dans l'image. En conséquence, lorsque le centre optique de la caméra est confondu avec le (ou les) centre(s) de rotation, il n'est pas possible d'obtenir une mesure de la profondeur par triangulation même en multipliant les prises de vue pour des angles différents. Nous pouvons donc considérer que tous les points sont à la même distance du centre c'est à dire sur la surface d'une sphère de rayon fixé.

2.2. Construction d'une mosaïque d'images

Nous allons aborder dans ce chapitre les différentes techniques qui vont nous permettre de construire une mosaïque d'images dans le cas idéal. C'est à dire que la caméra peut être modélisée avec le modèle sténopé que nous présentons en annexe et que le centre de projection est confondu avec le centre des rotations, que les paramètres de la prise de vue sont connus avec précision, qu'il n'y a pas de distorsion dans l'image et que la luminosité est constante dans toute la scène. Nous pouvons déjà imaginer que la réalité sera quelque peu éloignée de ce cas idéal.

2.2.1. Projection d'un point dans l'image

Si l'on se place dans les conditions idéales décrites ci-dessous, nous pouvons établir les équations qui relient un point quelconque de la scène et sa projection dans le plan image. Soit un point *P* quelconque du monde dans le repère de la caméra qui se projette en un point *p* sur le plan image. En fonction de la distance *f* entre le centre optique Oc et le plan image, les coordonnées 2D (u, v) de ce point de l'image peuvent s'exprimer à partir de deux angles (θ_p et φ_p) comme représenté sur la figure ci-dessous :

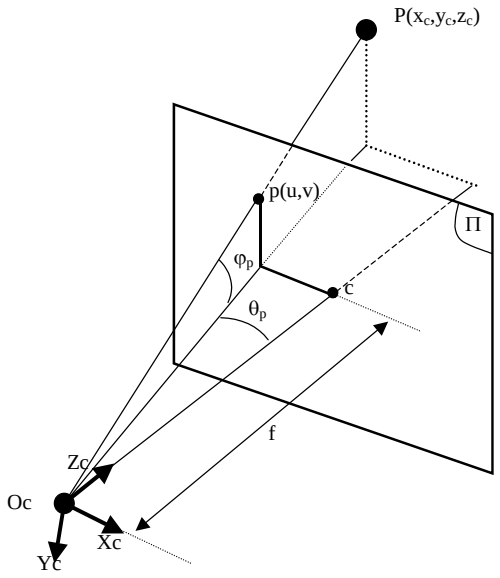


Figure 11 : Schéma de la projection d'un point du monde dans l'image

Les relations qui relient les coordonnées (u, v) d'un point aux angles (θ_p et φ_p) en fonction de f sont données par :

$$\theta_p = \tan^{-1}\left(\frac{u}{f}\right) \quad \text{et} \quad \varphi_p = \tan^{-1}\left(\frac{v}{\sqrt{u^2+f^2}}\right) \tag{2.1}$$

Dans la pratique, f représente la distance focale exprimée en unités pixels (up).

2.2.2. Unité pixel

L'unité pixel (notée up) que nous utilisons n'a pas de réalité physique. Cependant, nous l'utilisons pour faciliter l'écriture des équations. Nous partons de l'hypothèse que les cellules composant le capteur CCD sont rangées sous la forme d'une matrice rectangulaire comme les pixels de l'image et qu'il y a le même nombre de pixels que de cellules. Dans ce cas, l'unité pixel correspond à la taille d'une cellule sur le capteur.

$$1up\,(mm/pixel) = \frac{Largeur\ du\ capteur(mm)}{Nombre\ de\ colonnes\ de\ l'image} \tag{2.2}$$

Le tableau suivant indique l'ordre de grandeur des principaux capteurs CCD.

Format CCD	Hauteur (mm)	Largeur (mm)	Diagonale (mm)
1/4"	2.4	3.2	4
1/3"	3.6	4.8	6
1/2"	4.8	6.4	8
2/3"	6.6	8.8	11
1"	9.6	12.8	16

Tableau 1 : Ordre de grandeur de la taille des principaux capteurs CCD

Dans la plupart des applications nous faisons également l'hypothèse que les pixels sont carrés. Dans la réalité, les cellules composants le capteur ne sont pas carrées mais ont plutôt une forme rectangulaire. L'unité pixel n'a donc pas la même valeur suivant les deux axes. Dans le reste de ce manuscrit, sauf indication contraire, l'unité pixel que nous utilisons est identique selon les deux axes.

2.2.3. Projection d'un pixel de l'image sur la sphère

Ensuite, nous cherchons à établir la relation entre un pixel de l'image et sa projection sur la sphère de rayon 1. Dans le repère caméra, le vecteur unité du segment [O_cp] est défini par :

$$X^T = \left(\cos\varphi_p \cdot \sin\theta_p \quad \sin\varphi_p \quad \cos\varphi_p \cdot \cos\theta_p\right)^T .$$

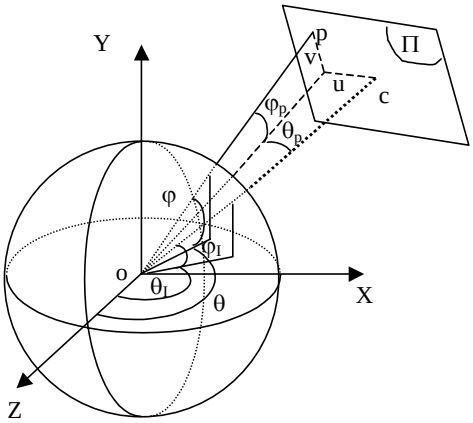


Figure 12 : Schéma de la projection d'un pixel de l'image sur la sphère

Ce vecteur peut représenter n'importe quel point dans le plan image. Dans le repère du monde, il correspond également aux coordonnées X du point d'intersection du segment [O,P] avec la surface de la sphère de rayon 1. Si l'origine du repère de la caméra est confondu avec l'origine du repère du monde et si on applique une rotation d'angle θ_I suivi d'une rotation d'angle φ_I au centre optique de l'image, les coordonnées X' d'un point quelconque de l'image dans ce nouveau repère, s'exprime par :

$$X = R^{-1}.X' \qquad \text{avec} \qquad R = R_{\varphi I}.R_{\theta I}$$

$$\text{et où } R_{\varphi i} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi_I & \sin\varphi_I \\ 0 & -\sin\varphi_I & \cos\varphi_I \end{bmatrix} \text{ et } R_{\varphi I} = \begin{bmatrix} \cos\theta_I & 0 & -\sin\theta_I \\ 0 & 1 & 0 \\ \sin\theta_I & 0 & \cos\theta_I \end{bmatrix}$$

Or, le vecteur unité du segment [OP] dans le repère du monde peut être est défini par deux angles θ et φ :

$$X^T = \left(\cos\varphi \cdot \sin\theta \quad \sin\varphi \quad \cos\varphi \cdot \cos\theta\right)^T$$

Nous pouvons en déduire la relation qui relie les coordonnées (u,v) d'un pixel de l'image avec les coordonnées (θ, φ) de la projection de ce pixel sur une sphère centrée sur le centre de projection. Nous obtenons :

$$\begin{bmatrix} \cos\varphi\cdot\sin\theta \\ \sin\varphi \\ \cos\varphi\cdot\cos\theta \end{bmatrix} = \begin{bmatrix} \cos\theta_I & 0 & \sin\theta_I \\ 0 & 1 & 0 \\ -\sin\theta_I & 0 & \cos\theta_I \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi_I & -\sin\varphi_I \\ 0 & \sin\varphi_I & \cos\varphi_I \end{bmatrix} \cdot \begin{bmatrix} \cos\varphi_P\cdot\sin\theta_P \\ \sin\varphi_P \\ \cos\varphi_P\cdot\cos\theta_P \end{bmatrix} \quad (2.3)$$

avec :

$$\theta_P = \tan^{-1}\left(\frac{u}{f}\right) \qquad \text{et} \qquad \varphi_P = \tan^{-1}\left(\frac{v}{\sqrt{u^2+f^2}}\right)$$

Ce système d'équations nous permet de projeter n'importe quel point d'une image sur une sphère, quelles que soient les conditions de prise de vue.

2.2.4. Relation entre deux images

Nous pouvons également utiliser cette expression pour représenter une autre image prise avec un angle de vue différent. Soit deux plans images π_1 et π_2 et un rayon passant par le centre optique Oc qui intersecte ces deux plans respectivement en deux points p_1 et p_2. θ_1 et θ_2 représentent les angles de rotation panoramique autour de l'axe Y des axes optiques des deux images et φ_1 et φ_2 les angles d'inclinations des deux axes.

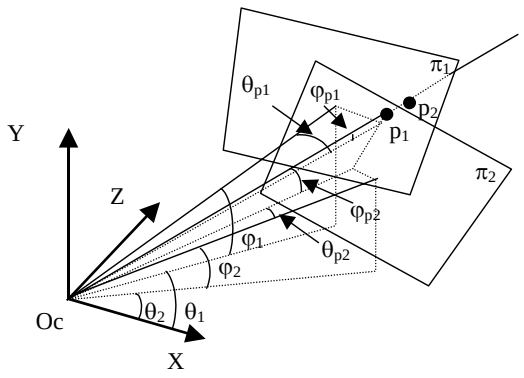


Figure 13 : Schéma de la relation entre deux images

Nous avons donc :

$$X_1 = R_{\varphi 1}.R_{\theta 1}.X \qquad \text{et} \qquad X_2 = R_{\varphi 2}.R_{\theta 2}.X$$

Nous en déduisons que :

$$X_1 = R_{\varphi 1}.R_{\theta 1}.R_{\theta 2}^{-1}.R_{\varphi 2}^{-1}.X_2 \quad (2.4)$$

Les matrices $R_{\theta 2}$ et $R_{\varphi 2}$ sont orthogonales, donc $R_{\varphi 2}^{-1} = R_{\varphi 2}^{T}$ et $R_{\theta 2}^{-1} = R_{\theta 2}^{T}$.

La relation qui relie les angles θ_{p1} et φ_{p1} de l'image 1 aux angles θ_{p2} et φ_{p2} de l'image 2 d'un point P s'exprime de la façon suivante :

$$\begin{bmatrix} \cos\varphi_{p1}\cdot\sin\theta_{p1} \\ \sin\varphi_{p1} \\ \cos\varphi_{p1}\cdot\cos\theta_{p1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi_1 & -\sin\varphi_1 \\ 0 & \sin\varphi_1 & \cos\varphi_1 \end{bmatrix} \bullet \begin{bmatrix} \cos(\theta_1-\theta_2) & 0 & -\sin(\theta_1-\theta_2) \\ 0 & 1 & 0 \\ \sin(\theta_1-\theta_2) & 0 & \cos(\theta_1-\theta_2) \end{bmatrix} \bullet \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi_2 & \sin\varphi_2 \\ 0 & -\sin\varphi_2 & \cos\varphi_2 \end{bmatrix} \bullet \begin{bmatrix} \cos\varphi_{p2}\cdot\sin\theta_{p2} \\ \sin\varphi_{p2} \\ \cos\varphi_{p2}\cdot\cos\theta_{p2} \end{bmatrix}$$

Cette expression peux se résumer de la façon suivante :

$$\begin{bmatrix} \cos\varphi_{p1}\cdot\sin\theta_{p1} \\ \sin\varphi_{p1} \\ \cos\varphi_{p1}\cdot\cos\theta_{p1} \end{bmatrix} = \begin{bmatrix} A \\ B \\ C \end{bmatrix}$$

Ou A, B et C sont les coefficients de la matrice ligne résultant du calcul numérique de l'expression précédente. A partir de cette relation, il est facile de retrouver les angles θ_{p1} et φ_{p1} :

$$\varphi_{p1} = \sin^{-1}(B) \text{ et } \theta_{p1} = \tan^{-1}\left(\frac{A}{C}\right) \text{ pour C} \neq 0 \qquad (2.5)$$

Ces deux angles permettent à présent de calculer les coordonnées (u_1, v_1) du point p dans l'image 1 :

$$u_1 = f\cdot\tan\theta_{p1}, \qquad v_1 = \tan\varphi_{p1}\cdot\sqrt{u_1^2+f^2}$$

2.2.5. Homographie entre deux images

Dans le paragraphe précédent, nous avons présenté une relation permettant de calculer la projection d'un pixel d'une image 1 dans une image 2. Cependant, si le centre des rotations de la caméra est confondu avec le centre optique (i.e. pas de translation de la caméra) et que le centre optique correspond à l'origine du repère de la caméra, alors, la relation qui relie deux images peut s'exprimer sous la forme d'une transformation homographique H [MAN94]. Une homographie 2D s'exprime avec 8 coefficients puisqu'elle est définie à un facteur d'échelle près (cf. annexe).

$$X_1 \sim H\cdot X_2 = \begin{bmatrix} m_0 & m_1 & m_2 \\ m_3 & m_4 & m_5 \\ m_6 & m_7 & 1 \end{bmatrix} \bullet \begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix} \qquad (2.6)$$

où X_1 est un point de l'image défini par ses coordonnées homogènes (u_1, v_1, 1) dans l'image et X_2 sa projection homographique dans l'image 2 est défini par ses coordonnées homogènes (u_2, v_2, 1). Le signe ~ indique la relation d'équivalence, à un facteur d'échelle près, entre le point X_1 et la transformation du point X_2.

Les coordonnées (u_1, v_1) du point X_1 dans l'image 1 se déduisent simplement :

$$u_1 = \frac{m_0\cdot u_2 + m_1\cdot v_2 + m_2}{m_6\cdot u_2 + m_7\cdot v_2 + 1}, \quad v_1 = \frac{m_3\cdot u_2 + m_4\cdot v_2 + m_5}{m_6\cdot u_2 + m_7\cdot v_2 + 1} \qquad (2.7)$$

Il existe plusieurs solutions permettant de déterminer la matrice H en fonction des données que l'on possède. La matrice H contenant 8 coefficients, cela signifie que seulement 4 points sont nécessaires pour résoudre le système linéaire . Ces quatre points minimum pourront être saisis par l'utilisateur ou obtenus de façon automatique avec un détecteur de Harris ou d'autres détecteurs plus robustes [TOR96, LOW04]. Lorsque le nombre d'équations est supérieur au nombre d'inconnues, le système n'a en général par de solution exacte et une solution approchée peut être obtenue au sens des moindres carrés

Il est également possible de déterminer les coefficients de la matrice de transformation à partir des paramètres intrinsèques et extrinsèques des prises de vue des deux images. Dans ce cas, la relation entre un point P de coordonnées (x, y, z) dans le repère de la caméra et un point X de coordonnée homogène (u, v, 1) dans le plan image, s'écrit :

$$X \sim T.K.R.P \tag{2.8}$$

où $T=\begin{bmatrix} 1 & 0 & u_{10} \\ 0 & 1 & v_{10} \\ 0 & 0 & 1 \end{bmatrix}$ est la matrice de changement d'unité et de translation du repère image,

$K=\begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$ est la matrice de changement d'échelle,

et $R=[r_{ij}]$ la matrice de rotation 3D.

Cette relation entre X et P correspond à un modèle simplifié de la projection perspective pour lequel nous considérons que d'une part le repère image est orthogonal (i.e. les pixels du capteur CCD sont rectangulaires) et que d'autre part, il n'y a pas de distorsion de l'image. Pour simplifier encore les équations, Szeliski [SZE97] fait l'hypothèse que le centre optique de la caméra passe par l'origine de l'image (i.e. $u_0 = v_0 = 0$) et que l'origine de l'image correspond au centre de l'image. Il fait en outre remarquer qu'un léger décalage entre l'axe optique et le centre de l'image n'a que peu d'incidence sur le résultat final. La relation simplifiée s'écrit alors : X ~ K.R.p.

La projection perspective d'un même point p du repère caméra donne $X_1 \sim K_1.R_1.p$ dans une image 1 et $X_2 \sim V_2.R_2.p$ dans une image 2. On en déduit rapidement la relation homographique entre les deux images :

$$H \sim V_2 \cdot R_2 \cdot R_1^{-1} \cdot V_1^{-1} \tag{2.9}$$

2.2.6. Calcul de la trajectoire optimale

L'acquisition de l'ensemble de la scène peut être abordée de deux façons. La première solution consiste à piloter la caméra en permanence et à procéder à l'acquisition du flux vidéo en temps réel. Chaque image est alors projetée sur le panorama. En optimisant un peu le code, il est relativement simple de ne projeter que la portion de la scène qui n'était pas présente dans l'image précédente. Cependant, les modèles de caméra que nous trouvons généralement dans le commerce ne sont pas des modèles idéaux et nous devons faire face à

deux problèmes. Le premier est que la vitesse de déplacement doit être particulièrement lente. Dès que la vitesse augmente, des phénomènes d'entrelacement entre les lignes paires et impaires apparaissent rapidement. Le deuxième inconvénient est que les paramètres extrinsèques de l'acquisition ne sont pas forcément disponibles en temps réel. Sur le modèle SONY RZ25P utilisée pour nos premières expérimentations, les angles de pan et de tilt ne sont mis à jour qu'une fois par seconde. Sur la caméra AXIS 233D utilisée dans un second temps, ces informations ne sont pas disponibles. Dans ces conditions, nous ne pouvons donc pas déterminer précisément la projection des pixels dans le panorama. En l'absence de techniques de recalage, qui seront abordées longuement dans le chapitre 4, nous cherchons donc à minimiser le nombre de prises de vue. L'acquisition de l'ensemble de la scène est réalisée en déplaçant rapidement la caméra d'une prise de vue à l'autre, puis en maintenant la position pendant au moins une seconde de façon à stabiliser l'image et permettre à la caméra de mettre à jour les paramètres extrinsèques. Notre problématique est donc de recouvrir l'ensemble des points d'une sphère à partir d'un nombre minimal d'images rectangulaires.

Le constructeur de la caméra SONY RZ25P, propose un balayage de la sphère en secteurs. C'est à dire que pour un angle de panorama donné, il réalise plusieurs acquisitions en faisant varier l'angle de tangage. L'optimisation simple utilisée par le constructeur consiste à faire varier les angles de tangage par ordre croissant puis décroissant à chaque décalage de panorama.

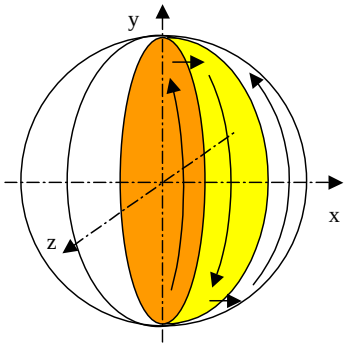


Figure 14 : Parcours de la sphère utilisé sur les caméras SONY RZ25P

Cette première solution offre quelques avantages. Le premier est que le parcours est simple à calculer puisqu'il suffit de s'assurer qu'à l'équateur, le nombre de pas de l'angle de panorama assure un recouvrement partiel des images. Dans une application de vidéo-surveillance, le deuxième intérêt est que la nature des objets mobiles (essentiellement des personnes) ont une forme allongée verticalement. Si les images sont prises relativement rapidement, le découpage de ces objets sera limité. Le principal inconvénient est que lorsqu'on s'approche des pôles, le recouvrement des images est de plus en plus important. L'information à traiter est dupliquée inutilement. Nous proposons une solution permettant définir un parcours limitant le nombre de prises de vue. Parmi les algorithmes permettant un parcours optimisé de la sphère, nous avons fait le choix d'étudier une approche par bandes sphériques horizontales. Il s'agit, pour une valeur de l'angle de tangage φ_0 donnée, de parcourir l'ensemble des angles de panorama θ de façon à obtenir une couverture complète entre deux valeurs φ_{max} et φ_{min} tout en maximisant la largeur de cette bande.

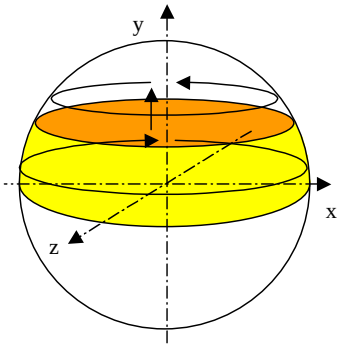


Figure 15 : Parcours de la sphère à partir d'une approche par bande sphérique horizontale

Dans cette classe d'algorithme nous allons étudier deux stratégies. Dans la première, nous considérons que l'écart entre les valeurs φ_0 de deux bandes successives est constant. Nous devons alors calculer le pas permettant d'assurer un recouvrement minimal entre chaque bande quelle que soit la latitude de la bande. Dans la deuxième, nous utilisons un pas variable qui est directement calculé à partir de la portion de la sphère déjà parcourue.

Dans un premier temps, nous allons étudier la surface occupée par une image en fonction des angles de panorama et de tangage. Nous allons particulièrement nous intéresser aux équations qui définissent les bords de l'image. Ces équations nous permettront de calculer plus simplement les intersections entre les images. Dans un chapitre précédent, nous avons démontré les relations qui lient les valeurs des angles à n'importe quel point de l'image. Cependant ces relations matricielles ne permettent pas de retrouver simplement la relation analytique de l'angle de tangage en fonction de l'angle de panorama. Nous allons donc établir, dans les paragraphes suivants, les équations des bords de l'image sous la forme :

$$\varphi = f(\theta)$$

A partir de ces expressions analytiques, il sera plus aisé de déterminer la surface occupée par la projection d'une image sur la sphère ainsi que les intersections entre les images.

2.2.6.1 Equations de la projection des bords haut et bas de l'image sur la sphère

Les équations qui délimitent les bords bas et haut de l'image sont les plus simples à calculer puisque ce sont des grands cercles d'inclinaison ($\varphi_0 \pm \varphi_h$), décalés de l'angle θ_0 où φ_0 et θ_0 sont les angles de la prise de vue et φ_h l'angle correspondant à la hauteur de l'image,

$$\varphi_h = \tan^{-1}\left(\frac{H}{2f}\right)$$

avec H = hauteur de l'image et f = distance focale.

L'équation polaire avec $\theta_0 = 0$ est donnée par :

$$\rho = \frac{a \cdot b}{\sqrt{b^2 \cdot \cos^2\theta + a^2 \cdot \sin^2\theta}} \text{ avec ici, } \left| \begin{matrix} a = \cos(\varphi_0 \pm \varphi_h) \\ b = 1 \end{matrix} \right.$$

donc,

$$\rho = \frac{\cos(\varphi_0 \pm \varphi_h)}{\sqrt{\cos^2(\theta) + \cos^2(\varphi_0 \pm \varphi_h) \cdot \sin^2(\theta)}} \quad (2.10)$$

ρ est aussi le cosinus de l'angle φ au point d'abscisse θ pour un grand cercle d'inclinaison φ_0. L'équation des bords haut et bas de l'image est donc :

$$\varphi = \cos^{-1}\left(\frac{\cos(\varphi_0 \pm \varphi_h)}{\sqrt{\cos^2(\theta) + \cos^2(\varphi_0 \pm \varphi_h) \cdot \sin^2(\theta)}} \right) \quad (2.11)$$

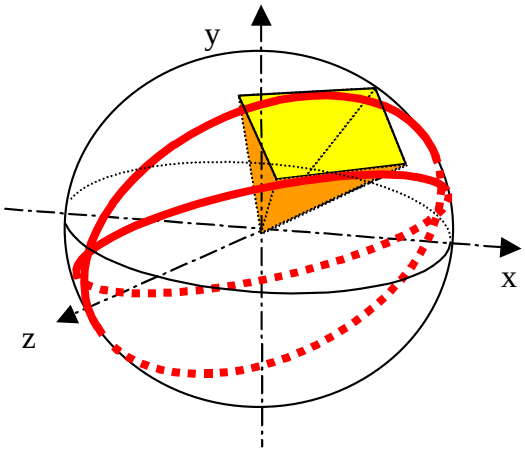


Figure 16 : Représentation des grands cercles (traits fort) des bords bas et haut de l'image

Sur la Figure 16, nous représentons les grands cercles confondus avec les portions de courbes correspondant à la projection des bords haut et bas de l'image. Sur cette figure, nous avons également représenté une image dont le centre est tangent à la sphère.

La Figure 17 représente ces mêmes grands cercles sur repère (θ, φ) pour une image de taille 640 x 480.

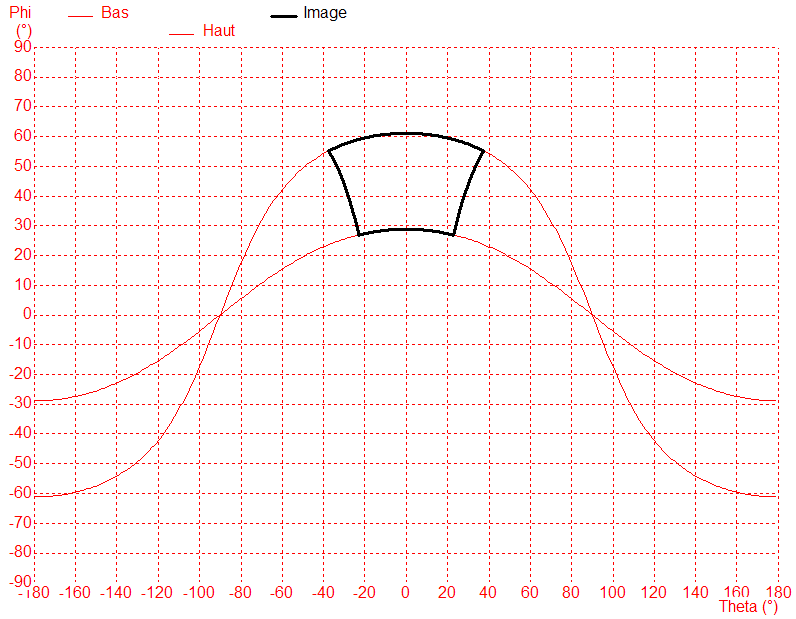


Figure 17 : Courbes des bords haut et bas d'une image de dimension 640x480 avec $\theta_0 = 0$, $\varphi_0 = 45°$ et $f = 830$.

2.2.6.2 Equations de la projection des bords droit et gauche de l'image sur la sphère

L'équation des bords droit et gauche de l'image est un peu plus délicate à exprimer. Ce sont également des grands cercles qui correspondent à l'intersection de la sphère et d'un plan perpendiculaire au plan $(\vec{x},\vec{z})$ auquel on a fait subir une rotation d'angle $(\theta_0 \pm \theta_l)$ autour de l'axe z et d'angle φ_0 autour de l'axe y.

Soit $\vec{n}=\begin{pmatrix}0\\1\\0\end{pmatrix} \perp x\circ z$. Les rotations $(\theta_0 \pm \theta_l)$ autour de l'axe z et φ_0 autour de l'axe y donnent :

$$\vec{n}=\begin{bmatrix}\cos\varphi_0 & 0 & -\sin\varphi_0\\ 0 & 1 & 0\\ \sin\varphi_0 & 0 & \cos\varphi_0\end{bmatrix}\cdot\begin{bmatrix}\cos(\theta_0\pm\theta_l) & -\sin(\theta_0\pm\theta_l) & 0\\ \sin(\theta_0\pm\theta_l) & \cos(\theta_0\pm\theta_l) & 0\\ 0 & 0 & 1\end{bmatrix}\cdot\begin{bmatrix}0\\1\\0\end{bmatrix}=\begin{bmatrix}-\sin(\theta_0\pm\theta_l)\cos\varphi_0\\ \cos(\theta_0\pm\theta_l)\\ -\sin(\theta_0\pm\theta_l)\sin\varphi_0\end{bmatrix}$$

L'équation du plan est donc :

$$-\sin(\theta_0\pm\theta_l)\cos\varphi_0\cdot x+\cos(\theta_0\pm\theta_l)y-\sin(\theta_0\pm\theta_l)\sin\varphi_0\cdot z=0 \qquad (2.12)$$

L'équation de la sphère de rayon 1 centrée sur O est donnée par :

$$x^2+y^2+z^2=1 \qquad (2.13)$$

En isolant z dans l'équation du plan (2.12) et l'injectant dans celle de la sphère (2.13), on obtient l'expression suivante:

$$x^2+y^2+\left(-\frac{\cos\varphi_0}{\sin\varphi_0}x+\frac{\cos(\theta_0\pm\theta_l)}{\sin(\theta_0\pm\theta_l)\sin\varphi_0}y\right)^2=1$$

Expression que l'on peut écrire :

$$\left(1+\frac{\cos^2\varphi_0}{\sin^2\varphi_0}\right)\cdot x^2+\left(1+\frac{\cos^2(\theta_0\pm\theta_l)}{\sin^2(\theta_0\pm\theta_l)\sin^2\varphi_0}\right)\cdot y^2-2\cdot\frac{\cos\varphi_0\cdot\cos(\theta_0\pm\theta_l)}{\sin(\theta_0\pm\theta_l)\sin^2\varphi_0}\cdot x\cdot y=1 \qquad (2.14)$$

Cette équation représente la projection du grand cercle sur le plan $(\vec{x},\vec{y})$.

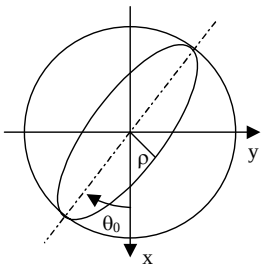


Figure 18 : Projection du grand cercle sur le plan $(\vec{x},\vec{y})$.

En coordonnées polaires, cette ellipse s'exprime aussi de la façon suivante :

$$\begin{cases} x=\rho\cdot\cos\theta \\ y=\rho\cdot\sin\theta \\ \rho=\left|\cos\varphi\right| \end{cases} \qquad (2.15)$$

On remarque ici que l'on introduit une ambiguïté liée au signe de ρ. Cette ambiguïté sera levée un peu plus loin. Quoi qu'il en soit, nous avons les expressions de x et y en fonction de θ et φ.

$$\begin{cases} x=\cos\varphi\cdot\cos\theta \\ y=\cos\varphi\cdot\sin\theta \end{cases} \qquad (2.16)$$

Nous pouvons utiliser ces expressions dans l'équation de la projection définie ci-dessus. Après réduction et en divisant le tout par $\cos^2\varphi$ sachant que $\frac{1}{\cos^2\varphi}=\tan^2\varphi+1$, on obtient :

$$\tan^2\varphi=\left(1+\frac{\cos^2\varphi_0}{\sin^2\varphi_0}\right)\cdot\cos^2\theta+\left(1+\frac{\cos^2(\theta_0\pm\theta_l)}{\sin^2(\theta_0\pm\theta_l)\cdot\sin^2\varphi_0}\right)\cdot\sin^2\theta-2\cdot\frac{\cos\varphi_0\cdot\cos(\theta_0\pm\theta_l)}{\sin(\theta_0\pm\theta_l)\cdot\sin^2\varphi_0}\cdot\sin\theta\cdot\cos\theta-1$$

L'équation des bords droit et gauche de l'image est donc donnée par :

$$\varphi=\tan^{-1}\left(\sqrt{A\cdot\cos^2\theta\ +\ B\cdot\sin^2\theta\ +C\cdot\sin\theta\cdot\cos\theta-1}\right) \qquad (2.17)$$

avec $A=\frac{1}{\sin^2\varphi_0}$; $B=1+\frac{\cos^2(\theta_0\pm\theta_l)}{\sin^2(\theta_0\pm\theta_l)\cdot\sin^2\varphi_0}$ et $C=-2\cdot\frac{\cos\varphi_0\cdot\cos(\theta_0\pm\theta_l)}{\sin(\theta_0\pm\theta_l)\cdot\sin^2\varphi_0}$

Afin de régler le problème de signe introduit plus haut, nous pouvons remarquer que le signe de φ est le même que celui de z. Comme nous avons isolé z dans l'équation du plan (2.12), nous savons que :

$$z = \frac{-\cos\varphi_0}{\sin\varphi_0}\cdot x + \frac{\cos(\theta_0 \pm \theta_l)}{\sin(\theta_0 \pm \theta_l)\cdot\sin\varphi_0}\cdot y \quad \text{avec} \quad x = \cos\varphi\cdot\cos\theta \text{ et } y = \cos\varphi\cdot\sin\theta \qquad (2.18)$$

Nous en déduisons donc que :

$$z = \frac{-\cos\varphi_0}{\sin\varphi_0}\cdot\cos\varphi\cdot\cos\theta + \frac{\cos(\theta_0 \pm \theta_l)}{\sin(\theta_0 \pm \theta_l)\cdot\sin\varphi_0}\cdot\cos\varphi\cdot\sin\theta \qquad (2.19)$$

Le changement de signe s'opère pour z = 0, c'est à dire pour $\theta = \tan^{-1}(\cos\varphi_0\cdot\tan\theta_0)$

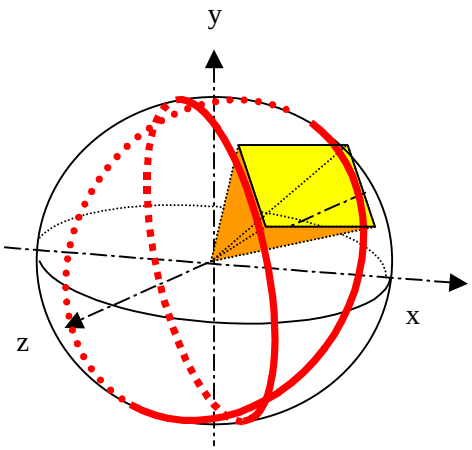


Figure 19 :Représentation des grands cercles des bords droit et gauche de l'image

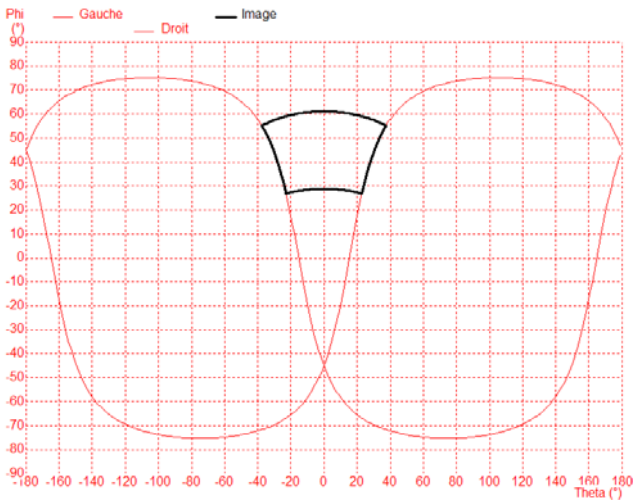
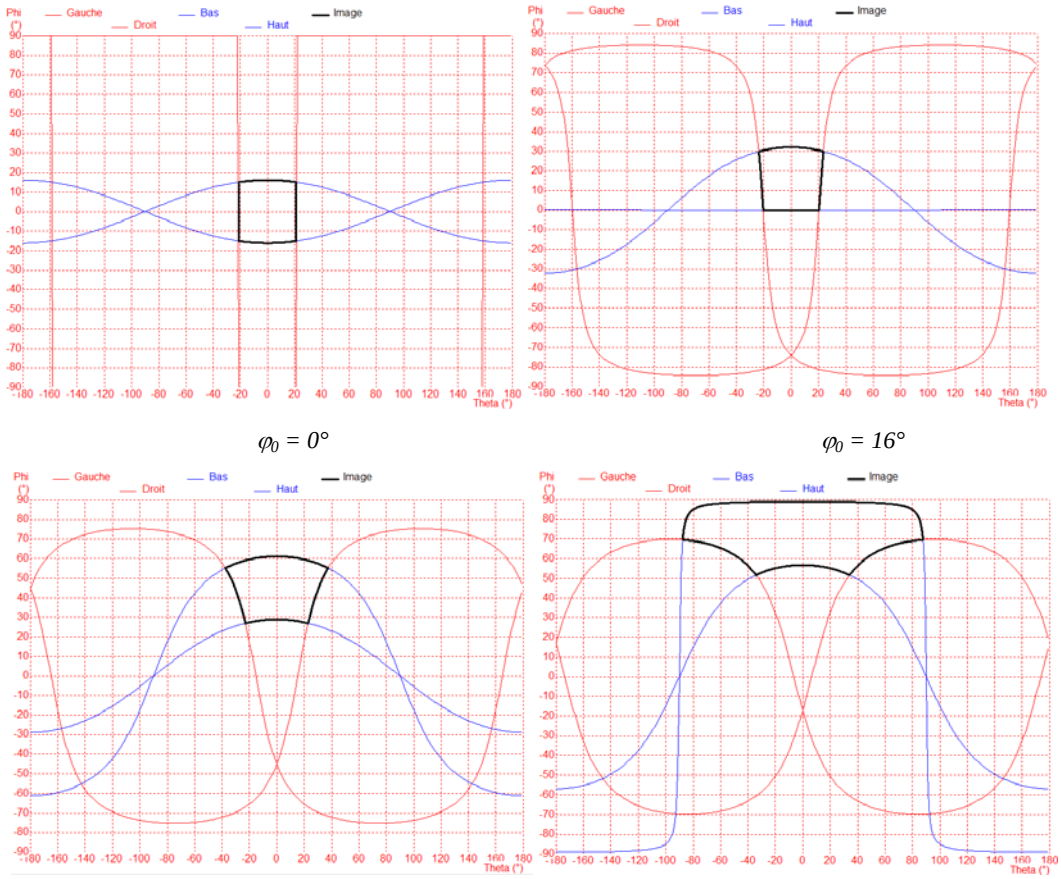


Figure 20 :Courbes des bords droit et gauche d'une image de dimension 640x480 avec θ_0 = 0, φ_0 = 45° et f = 830.



$\varphi_0 = 0°$ $\varphi_0 = 16°$

$\varphi_0 = 45°$ $\varphi_0 = 73°$

Figure 21 : Représentation des projections des quatre bords de l'image pour différentes valeur de φ_0

2.2.6.3 Hauteur de la bande couverte

Les quatre équations des bords de l'image nous permettent d'analyser plus finement les projections des angles solides de chaque image en fonction des conditions de prise de vue. Dans les deux approches par bande sphérique horizontale que nous proposons, pour une valeur φ_0 donnée, nous faisons varier l'angle de panorama. Le pas de variation $\Delta\theta$ est alors optimisé pour chaque φ_0 de façon à limiter les recouvrements et donc limiter le nombre d'images à acquérir pour balayer l'ensemble de la scène. Donc pour un φ_0 et un $\Delta\theta$ nous obtenons une bande comme représentée ci-dessous.

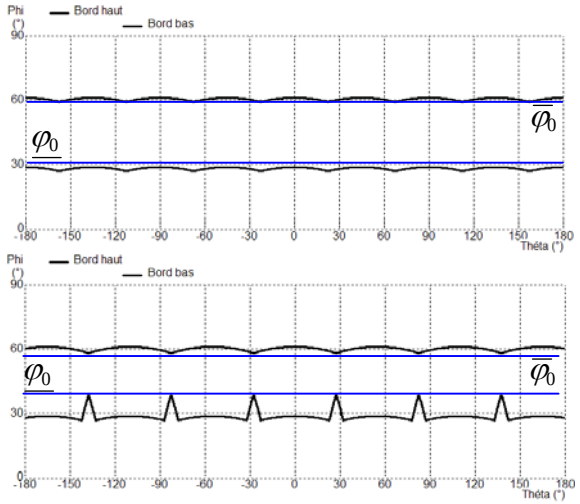


Figure 22 : Représentation des bords hauts et bas pour des images de 640x480 avec φ_0 = 45°, f = 830 et $\Delta\theta$ = 45° (à gauche) et $\Delta\theta$ = 55° (à droite)

En recherchant la valeur minimale $\overline{\varphi_0}$ du bord haut et la valeur maximale $\underline{\varphi_0}$ du bord bas, nous obtenons la hauteur maximale de la bande que nous sommes assurés de couvrir sans laisser de trous. La valeur du pas $\Delta\theta$ doit être choisie judicieusement. Un pas trop petit entraîne un recouvrement inutile alors qu'un pas trop grand risque de diminuer fortement la hauteur de la bande, voire ne pas assurer une connexité entre les images. Nous considérons 4 domaines représentés ci-dessous et dont les limites sont fixées par $\Delta\theta_1$, $\Delta\theta_2$, $\Delta\theta_3$ et $\Delta\theta_4$.

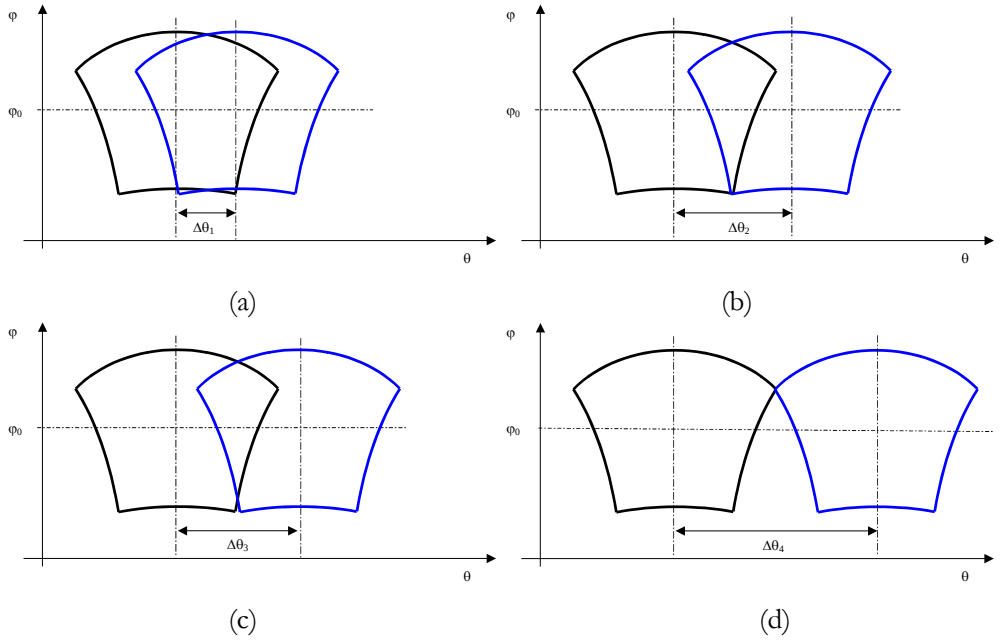


Figure 23 : Evolution du recouvrement entre les images pour différentes valeurs de $\Delta\theta$

Les paramètres (φ_1, $\Delta\theta_1$) de la bande suivante sont alors déterminés de façon à ce que pour $\varphi_1 > \varphi_0$, $\underline{\varphi_1} \leq \overline{\varphi_0}$. La hauteur de la bande dépend de φ_0 bien sur, mais surtout du pas $\Delta\theta$ qui conditionne le recouvrement des images sur une même bande. Il s'agit d'une heuristique permettant d'obtenir une condition suffisante au recouvrement mais pas nécessaire. Pour le bord haut de l'image, l'analyse est assez simple puisque tant que l'on a $0 \leq \Delta\theta \leq \Delta\theta_4$ (voir Figure 24), la valeur minimale du bord haut, correspondant à l'intersection des deux images, se situe toujours sur le bord haut de l'image pour une valeur $\theta = \theta_0 + \frac{\Delta\theta}{2}$. Pour le bord bas, le calcul est un peu plus compliqué. Nous pouvons définir 4 domaines. Pour $0 \leq \Delta\theta \leq \Delta\theta_1$ (Figure 23a), la valeur maximale du bord bas correspond à l'intersection entre le bord bas de l'image 1 et le coté gauche de l'image 2. La valeur $\Delta\theta_1$ correspond à l'angle pour lequel l'intersection entre les deux images se situe sur l'axe de symétrie de l'image 1. Dans cet intervalle la fonction $\underline{\varphi_0} = f(\Delta\theta)$ est croissante et évolue de $\underline{\varphi_0} = \varphi_0 - \tan^{-1}\left(\frac{h}{\sqrt{l^2+f^2}}\right)$ pour $\Delta\theta = 0$ à $\underline{\varphi_0} = \varphi_0 - \tan^{-1}\left(\frac{h}{2 \cdot f}\right)$ pour $\Delta\theta = \Delta\theta_1$. Si $\Delta\theta_1 < \Delta\theta \leq \Delta\theta_2$ (Figure 23b), la fonction est continue. En effet l'intersection entre les deux images se situe toujours sur le bord bas de l'image 1 après l'axe de symétrie. Dans cet intervalle la fonction du bord bas de l'image est décroissante. La valeur maximale atteinte reste donc $\underline{\varphi_0} = \varphi_0 - \tan^{-1}\left(\frac{h}{2 \cdot f}\right)$. A $\Delta\theta = \Delta\theta_2$, le coin inférieur droit de l'image 1 est confondu au coin inférieur gauche de l'image 2. Si on augmente encore la valeur de $\Delta\theta$, les deux images s'intersectent le long de leur bord verticaux. Cependant si $\Delta\theta_2 < \Delta\theta \leq \Delta\theta_3$ (Figure 23c), la valeur de l'angle φ reste inférieure à $\underline{\varphi_0} = \varphi_0 - \tan^{-1}\left(\frac{h}{2 \cdot f}\right)$, donc, la valeur maximale reste inchangée. Ensuite, si $\Delta\theta_3 < \Delta\theta \leq \Delta\theta_4$ (Figure 23d), la valeur est supérieure est augmente régulièrement jusqu'à atteindre la valeur $\underline{\varphi_0} = \varphi_0 + \tan^{-1}\left(\frac{h}{\sqrt{l^2+f^2}}\right)$ pour $\Delta\theta = \Delta\theta_4$. Si $\Delta\theta > \Delta\theta_4$, il n'y a plus d'intersection entre les images. Nous ne pouvons donc plus assurer un balayage sans trous.

Pour terminer, il convient de préciser que cette étude est valable lorsque $\overline{\varphi_0}$ et $\underline{\varphi_0}$ sont tous les deux strictement supérieurs à zéro. Dans le cas où leurs valeurs sont strictement inférieures, il suffit de considérer leur valeur absolue pour retomber dans le cas décrit plus haut. Lorsque les valeurs de $\overline{\varphi_0}$ et $\underline{\varphi_0}$ ne sont pas du même signe, c'est à dire lorsque $\overline{\varphi_0} > 0$ et $\underline{\varphi_0} < 0$, la fonction $\underline{\varphi_0} = f(\Delta\theta)$, s'analyse comme la valeur absolue d'un bord haut.

Le graphique suivant représente les fonctions $\overline{\varphi_0}$ et $\underline{\varphi_0}$ pour une image 640x480 avec $\varphi_0 = 45°$ et f = 830. Nous pouvons constater que la hauteur de la bande couverte est maximale pour $\Delta\theta = 0$, ce qui n'a pas beaucoup d'intérêt, et que cette hauteur diminue lorsque $\Delta\theta$ augmente. Nous rappelons que $\Delta\theta$ est inversement proportionnel au nombre d'images nécessaires pour couvrir une bande complète.

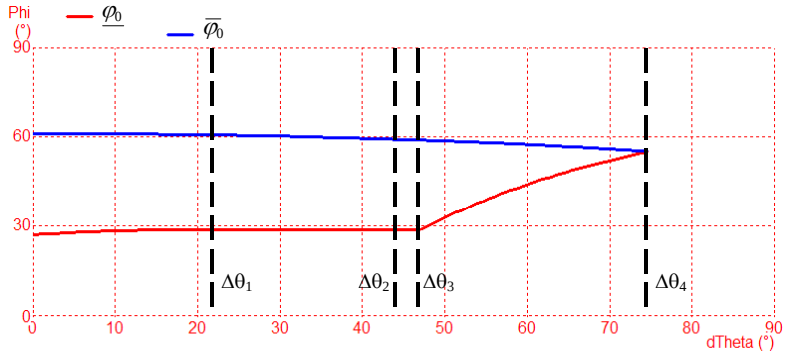


Figure 24 : fonctions $\overline{\varphi_0}$ et $\underline{\varphi_0}$ d'une image de dimension 640x480 avec φ_0 = 45° et f = 830

Tant que $\Delta\theta < \Delta\theta_3$ la décroissance de la hauteur de la bande reste faible en fonction des valeurs croissantes de $\Delta\theta$. Par contre, des que $\Delta\theta > \Delta\theta_3$ la décroissante est plus rapide.

2.2.6.4 Optimisation de la trajectoire avec φ fixe

Nous avons à présent tous les outils nous permettant de rechercher une heuristiques d'optimisation du recouvrement de la sphère dans une approche par bandes sphériques horizontales. La première solution que nous proposons consiste à fixer l'incrément $\Delta\varphi$ à une valeur unique entre toutes les bandes successives. Ceci implique que quelle que soit la latitude d'une bande, nous devons assurer un recouvrement minimum entre la bande considérée et la bande précédente.

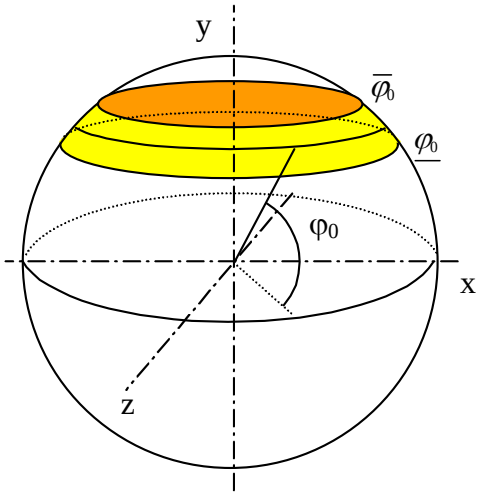


Figure 25 :Représentation de la largeur de la bande sphérique horizontale

A partir des équations et des différentes remarques que nous avons formulées précédemment, nous étudions l'évolution de la largeur maximale de la bande en fonction de la latitude. Comme nous l'avons indiqué dans le paragraphe précédent, pour une valeur de φ_0 donnée, la largeur de la bande atteint un maximum lorsque l'incrément de l'angle de panorama $\Delta\theta$ tend vers 0. Cependant, lorsque l'incrément diminue, le nombre d'images nécessaire à la couverture de la bande augmente. Le pas de l'incrément doit donc être voisin

de $\Delta\theta_2$ de façon à optimiser le nombre d'images et la hauteur de la bande. Nous devons également tenir compte du fait que le nombre d'images par bandes est forcément un nombre entier. Ce nombre d'images est donc l'entier immédiatement supérieur correspondant au rapport entre l'angle de panorama couvert et le pas d'incrément théorique. Fort de ces différentes considérations, le graphique suivant représente la variation de la hauteur de la bande en fonction de l'angle de tangage pour différents facteurs de zoom.

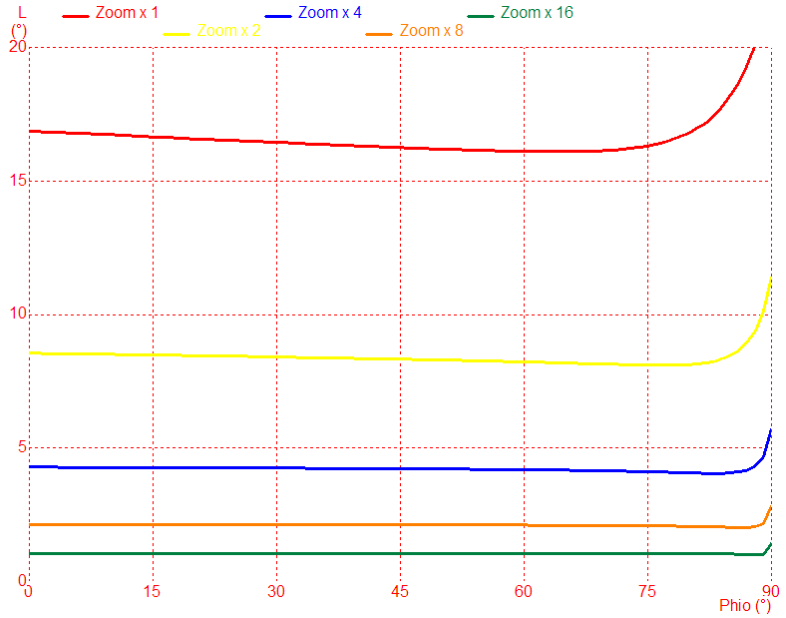


Figure 26 :Variation de la hauteur de la bande sphérique horizontale en fonction de la latitude pour différentes valeurs de zoom. Le zoom x 1 correspond à une focale de 8mm et les images acquises ont une résolution de 640x480.

Sur ce graphique, nous pouvons remarquer que la hauteur maximale de la bande décroît sensiblement en fonction de φ_0 jusqu'à atteindre un minimum puis augmente pour atteindre un maximum à $\varphi_0 = 90°$. Si nous calculons l'incrément de l'angle de tangage de chaque bande sur la base de la bande $\varphi_0 = 0°$, alors nous ne pouvons pas garantir un recouvrement optimal puisque les bandes de part et d'autre de la bande de référence auront une hauteur légèrement inférieure. L'incrément doit donc être calculé à partir de la valeur de φ_0 pour laquelle la largeur est la plus faible.
Nous remarquons également que le minimum de la largeur n'est pas atteint pour la même valeur de φ_0 suivant le facteur de zoom. Cela n'apparaît pas sur ce graphique, mais nous pouvons faire le même constat en fonction de la taille de l'image. Nous présentons sous la forme de l'abaque suivant, la valeur optimale de l'incrément en fonction du facteur de zoom et pour plusieurs tailles d'image standards :

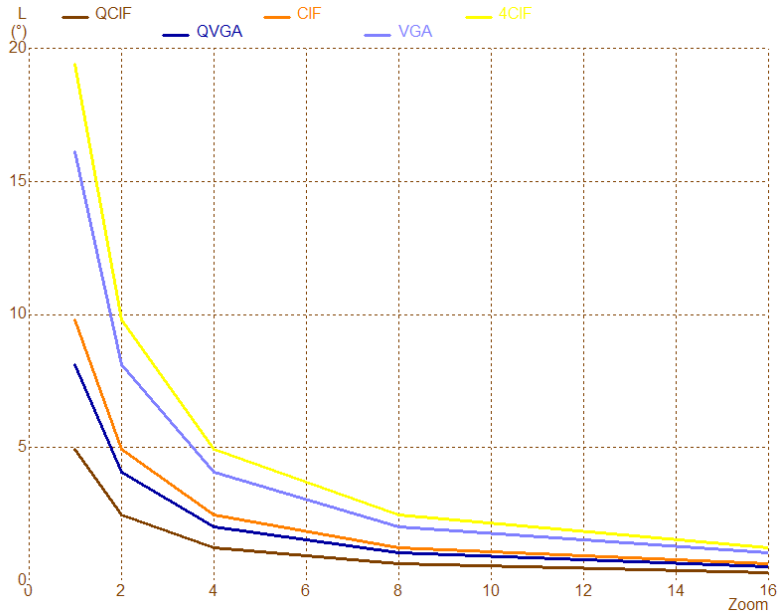


Figure 27 : Abaque de la hauteur optimale de la bande sphérique horizontale en fonction du facteur de zoom pour plusieurs tailles d'image.

A partir de ce graphique nous pouvons déterminer l'incrément maximal de l'angle de tangage $\Delta\varphi_{max}$ qui assure un recouvrement minimal entre les bandes successives en fonction de la résolution de l'image et du facteur de zoom utilisé. L'incrément que nous allons réellement utiliser n'a pas forcément la valeur de $\Delta\varphi_{max}$ mais une valeur qui peut être plus faible. En effet, la valeur de $\Delta\varphi_{max}$ nous permet de calculer le nombre de bande que nous allons devoir parcourir.

$$Nbbandes = \frac{\left|\varphi_{arrivé} - \varphi_{départ}\right|}{\Delta\varphi_{max}} \tag{2.20}$$

Cependant, il est peu probable que le nombre de bande ainsi calculé soit un nombre entier. Le nombre de bande réel est donc la valeur entière immédiatement supérieure. En définitive, l'incrément sera alors :

$$\Delta\varphi = \frac{\left|\varphi_{arrivé} - \varphi_{départ}\right|}{Nbbandes\,réel} \tag{2.21}$$

2.2.6.5 Trajectoire optimale avec φ variable

La première approche que nous venons d'étudier est une heuristique qui permet d'assurer le parcours complet de la sphère en se basant sur la largeur la plus faible pour une bande. Cependant, nous avons vu que cette largeur de bande pouvait être plus grande pour les angles de tangages proches de 0° et proches de 90°. L'approche que nous étudions à présent consiste à déterminer pour chaque bande, la valeur maximale de l'incrément en fonction de ce qui a déjà été parcouru. Notre stratégie vise localement à maximiser le rapport entre l'angle de tangage couvert $\Delta\varphi$ et le nombre d'images. Nous avons vu dans le paragraphe précédent qu'un bon compromis entre le nombre d'images et la largeur de bande se situe

pour un $\Delta\theta$ au voisinage de θ_2. Il serait tentant d'imaginer que le nombre d'images optimal est celui pour lequel $\Delta\theta$ est le plus proche de θ_2. Ce n'est pas forcement le cas et ce pour deux raisons. La première est que le nombre d'images étant un nombre entier, il y a finalement peu de chance de trouver un nombre tel que $\Delta\theta = \theta_2$. La seconde raison est qu'il est possible que dans certaines conditions, le fait d'augmenter légèrement $\Delta\theta$, entraînant inévitablement une diminution de la limite haute, nous fasse gagner une image sur la bande. A l'inverse, diminuer $\Delta\theta$ revient à augmenter le nombre d'images mais permet également d'augmenter légèrement la limite haute. Cette légère augmentation pouvant, au final, nous permettre de supprimer une bande.

A partir d'une valeur φ_0 de départ, qui correspond à une des limites du matériel utilisé, nous pouvons calculer les limites $\overline{\varphi}_{0,n}$ pour plusieurs nombre d'images n. Ces limites correspondent aux hauteurs maximales que nous sommes assurés de couvrir en fonction du nombre d'images utilisées, c'est à dire du pas $\Delta\theta$ entre chaque image. Pour chaque valeur de $\overline{\varphi}_{0,n}$, nous pouvons calculer les valeurs $\varphi_{1,n}$, pour lesquelles $\overline{\varphi}_{1,n} \leq \underline{\varphi_{0,n}}$. Nous obtenons ainsi un arbre représentant l'ensemble des solutions possibles.

Dans le graphique ci-dessous nous avons représenté l'ensemble des solutions possibles par des croix. Les liens qui relient ces croix correspondent aux branches de l'arbre. Ce graphique permet de déterminer le nombre d'images nécessaires pour atteindre une certaine valeur de φ ainsi que les différentes étapes qui permettent d'y accéder. Le tracé en trait continu fort indique la solution optimale en nombre d'images pour atteindre une certaine valeur de φ. Le graphe ci-dessous est celui du matériel que nous utilisons et qui ne permet pas de descendre en dessous de –30°. Nous pouvons remarquer que dans certains cas, notamment entre 11 et 15 images, le fait d'augmenter le nombre d'images ne permet pas d'augmenter significativement l'étendue du φ couvert.

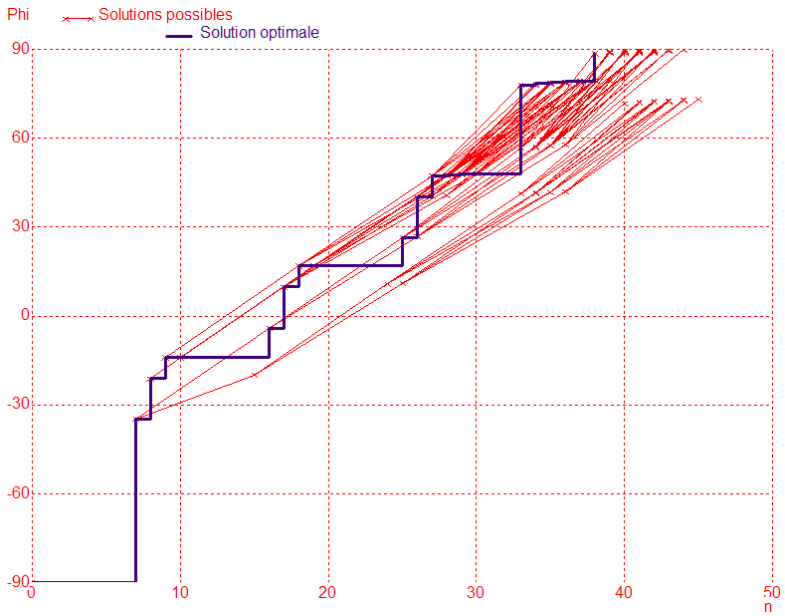


Figure 28 : Etendue du φ couvert en fonction du nombre d'image.

Nous pouvons également constater que, bien que notre matériel ne permet pas de piloter l'angle de tangage à des valeurs inférieures à –30°, la limite φ_0 est inférieure à cette valeur. Ceci est du à un cas particulièrement défavorable pour lequel le nombre d'images choisi (içi n = 7) conduit à une valeur de $\Delta\theta$ comprise entre $\Delta\theta_3$ et $\Delta\theta_4$ et visiblement, plus proche de $\Delta\theta_4$ que de $\Delta\theta_3$. Dans le cas de notre caméra et plus généralement lorsque l'on ne cherche pas une couverture complète de l'angle de tangage, la limite basse de la première bande ou la limite haute de la dernière (sauf si elles correspondent aux pôles), est un problème mal posé. En effet, cette limite théorique dépend de l'angle de tangage mais également de la taille de l'image et du facteur de zoom. Cependant, la forme de la projection d'une image sur la sphère (cf. équation des bords haut et bas) est telle que cette limite n'est atteinte que par un pixel de l'image voire deux si les angles $\varphi_{0,\text{ debut}}$ et $\varphi_{0,\text{ fin}}$ sont de même signe. Nous pouvons quand même atteindre cette limite théorique avec un nombre fini d'image. Il faut pour cela que l'incrément $\Delta\theta$ entre deux images corresponde à l'angle sous lequel est vu un pixel. Ceci implique un nombre important d'image (plusieurs centaines). A titre d'exemple, nous présentons sur le graphique suivant la couverture d'une bande pour φ_0 = -30°, f = 800 up et avec des images dont la résolution est 640 x 480. Dans ce cas de figure, la limite théorique minimum que l'on peut atteindre est $\varphi_{0,th}$ = -45.1°. Le premier exemple est réalisé avec 7 images. La valeur limite atteinte est $\varphi_{0,7img}$ = -41.7°. Dans le deuxième exemple réalisé avec 13 images, elle est $\varphi_{0,13img}$ = -44.7°.

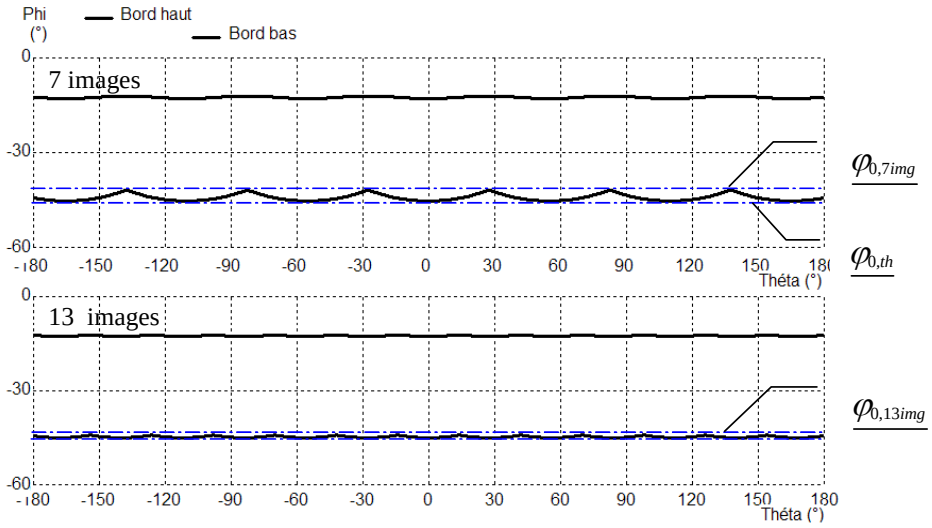


Figure 29 : Exemple de deux bandes sphériques horizontales réalisées avec 7 images (en haut) et 13 images (en bas).

2.2.6.6 Résultats

Nous présentons les résultats obtenus avec les 3 algorithmes :

- celui du constructeur, où le parcours est réalisé en secteur (Bande V.),
- le parcours en bandes sphériques horizontales avec un incrément $\Delta\varphi$ fixe (Bande H. fixe),
- le parcours en bandes sphériques horizontales avec un incrément $\Delta\varphi$ variable (Bande H. variable).

Nous comparons ces résultats avec la limite théorique (Borne) que nous pourrions atteindre si il était possible de paver une sphère (sans recouvrement) à partir de surfaces rectangulaires.

Cette valeur est obtenue en divisant l'angle solide de la sphère par l'angle solide sous lequel est vue une image.

Facteur zoom	Borne	Bande V.	Bande H. fixe	Bande H. variable
1	27	48	44	44
2	105	176	151	139
4	419	672	524	488
8	1676	2646	1970	1823
16	6703	10584	7544	7100

Tableau 2 : Comparaison du nombre d'images nécessaire pour recouvrir la sphère à partir des différents algorithmes

Les résultats présentés ci dessus sont calculés pour une résolution d'image de 640 x 480 pixels. Ils indiquent le nombre d'images nécessaires pour recouvrir les $4\pi sr$ de la sphère en fonction du facteur de zoom. Le zoom x 1 correspond à une focale de 4.1 mm soit une distance de 800up. La lecture de ce tableau montre que pour le facteur de zoom le plus faible, les trois algorithmes sont comparables. Par contre, l'écart par rapport à la borne inférieur est important. A partir d'un facteur de zoom = 2, l'apport de l'approche par bandes sphériques horizontales avec un $\Delta\varphi$ variable est significatif. Il permet un gain de 12 images par rapport à l'approche fixe et de 37 images par rapport à l'approche du constructeur.

2.2.7. Polygone d'intersection

Une problématique récurrente à laquelle nous allons être confrontés est le calcul de la zone de recouvrement entre deux images liées par une homographie. Le calcul de ce polygone d'intersection nous permettra d'assurer un minimum de recouvrement entre deux images, d'optimiser les calculs lors de la mise en correspondance ou encore d'améliorer le rendu de la mosaïque d'images en lissant les différences de luminosité entre les images.

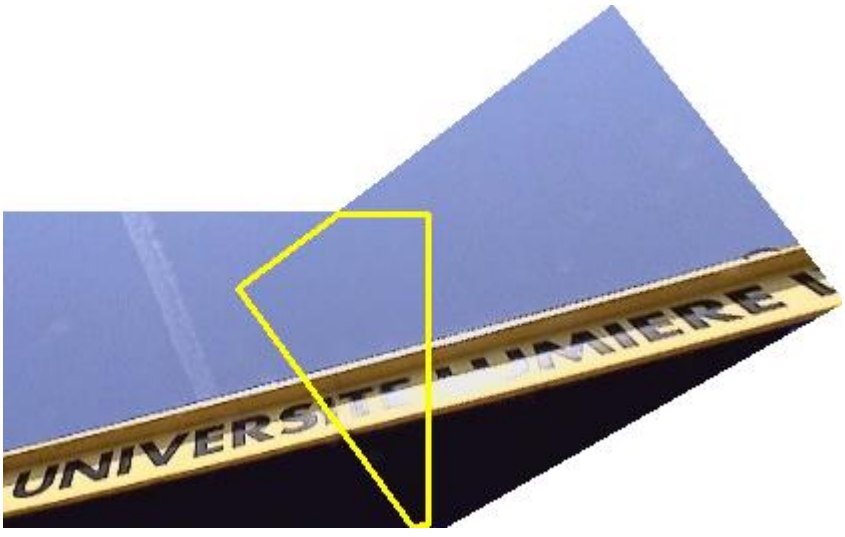


Figure 30 : Exemple d'une zone de recouvrement entre deux images

Ce problème complexe analytiquement peut être résolu par un algorithme relativement simple. Soient deux images A et B dont les paramètres de prise de vue θ_A, φ_A, f_A et θ_B, φ_B, f_B, sont connus et où θ et l'angle de panorama, φ l'angle de tangage et f la distance focale. Le polygone P_A est formé à partir des 4 coins de l'image A et le polygone P_B est formé à partir de la projection homographique des 4 sommets de l'image B dans l'image A. A partir de ces deux polygones nous pouvons déterminer les différents sommets du polygone d'intersection en utilisant l'algorithme suivant :

```
Pour chaque segment Seg_ai de A faire
      Pour chaque segment Seg_bj de B faire
            Si intersection entre Seg_ai et Seg_bj
                  Ajouter le point d'intersection

Pour chaque sommet Som_ai de A faire
      Si Som_ai est à l'intérieur de B
            Ajouter le sommet

Pour chaque sommet Som_bi de B faire
      Si Som_bi est à l'intérieur de A
            Ajouter le sommet
```

Cet algorithme se composent de 3 séries de deux boucles imbriquées, c'est à dire que sa complexité s'exprime en O(mn). Il est possible de réduire la complexité de cet algorithme en O(m+n). Toutefois, comme m = n = 4, l'optimisation du code n'est pas indispensable. Pour plus de détail sur l'implémentation de cet algorithme et notamment sur le calcul des intersections et la position d'un point par rapport à un polygone quelconque, le lecteur pourra se référer au livre « Computational Geometry in C » de Joseph O'Rourke [ROU98].

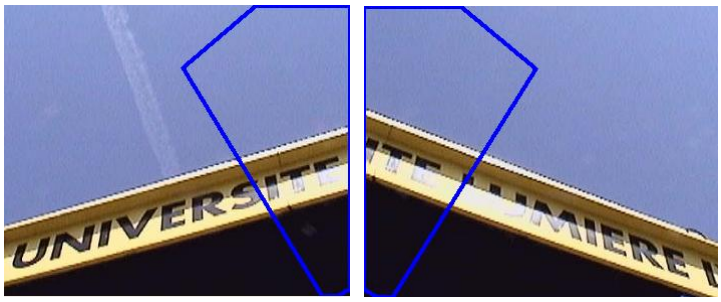


Figure 31 : Exemple de calcul du polygone d'intersection entre deux images liées par une homographie

2.3. Visualisation des mosaïques d'images

Nous avons à présent tous les outils nous permettant de projeter les images et d'optimiser leur acquisition. Nous allons pouvoir aborder les différents modes de représentation. La création d'une mosaïque d'images consiste donc à représenter sur une seule structure de données la vue d'ensemble de la caméra. A partir d'une mosaïque d'images, les applications sont nombreuses. Quelle que soit l'application, l'une des finalités les plus courantes reste la restitution d'une prise de vue particulière qui n'aurait pas été directement acquise. Plus concrètement, à partir de deux images prises avec des angles de vue différents, il doit être possible de calculer l'image équivalente à une prise de vue intermédiaire, pour peu qu'il y ait recouvrement partiel entre les deux images de départ.

Dans le cas d'une projection centrale et où le centre optique de la caméra est confondu avec les axes de rotations, le monde vu par la caméra peut être considéré comme une sphère. Nous cherchons donc à exprimer le nombre de pixels optimal permettant la représentation de cette sphère qui limite la perte d'information en fonction de la résolution du capteur et de la distance focale. Donc en fonction de la taille du capteur et de la distance focale, un pixel de l'image couvre un certain angle solide. Une approximation de cet angle solide est donnée par :

$$\Omega = \frac{l \cdot h}{f^2}$$

où l est la largeur du pixel, h sa hauteur et f la distance focale. Cette approximation est valable si l et h sont très inférieurs à f. Pour un capteur 1/3", l et h sont de l'ordre du micromètre alors que f est de l'ordre du millimètre. Il y a donc un rapport 1000 entre la taille du capteur et la distance focale. Exprimé en unité pixel, l'angle solide est donné par :

$$\Omega(up) = \frac{1}{f^2}$$

L'angle solide d'une sphère étant de 4πsr, ceci implique que le nombre de pixels nécessaire à la représentation de la sphère est donnée par :

$$nb\ pixels = 4 \cdot \pi \cdot f^2 \tag{2.22}$$

Ce nombre de pixel est une approximation dans laquelle nous considérons que chaque pixel représente une unité de surface. Nous pouvons donc en déduire que pour représenter un panorama en limitant les pertes d'information, nous devons utiliser une sphère de rayon égale à la distance focale exprimée en pixels. Si nous utilisons une sphère plus petite, nous allons concentrer l'information. Si nous utilisons une sphère plus grande, nous n'aurons pas assez de résolution pour un pavage complet.

Pour simplifier le problème, nous avons calculé l'angle solide couvert par le pixel du centre de l'image ou plus exactement du pixel situé à l'intersection du plan image et de l'axe optique. L'angle solide de n'importe quel pixel du plan image perpendiculaire à l'axe optique est donnée par :

$$\Omega = \cos\theta \cdot \sin\varphi \cdot \frac{l \cdot h}{f^2} \tag{2.23}$$

où θ et φ sont les angles entre l'axe optique et la droite passant par le centre optique et le centre du pixel. En définitive, ce sont donc les pixels des coins de l'image qui couvrent l'angle solide le plus faible. En toute rigueur nous devrions donc utiliser cette expression. Avec un capteur 1/3", une focale de 4.1mm et une résolution d'image de 640 x 480, l'erreur sur le calcul de l'angle solide est de l'ordre de 11%. Elle n'est plus que de 3% dans les mêmes conditions mais avec une distance focale deux fois plus grande.

La résolution optimale de la représentation du monde vue par la caméra qui limite autant que faire se peut les pertes d'informations, est une sphère dont le rayon est de l'ordre de la distance focale exprimée en pixels. Les deux défis à relever sont d'une part la complexité algorithmique nécessaire au calcul de cette représentation et d'autre part la minimisation de l'espace de stockage. L'espace de stockage optimal est simple à déterminer. Il correspond simplement, comme nous l'avons vu, à la surface de la sphère exprimée en nombre de pixels.

$$s = 4 \cdot \pi \cdot f^2 \tag{2.24}$$

C'est cette limite que l'on va chercher à approcher. Cependant dans ce type de représentation, la complexité algorithmique sera importante puisqu'il faudra calculer la projection de chaque pixel.

Le mode de représentation le plus classique est la projection sur un cylindre. Cette projection offre notamment la possibilité de visualiser en une seule image l'ensemble de la scène. Par contre la représentation des pôles est impossible et la complexité algorithmique est la même que la sphère puisqu'il est nécessaire de calculer la projection de chaque pixel.

Afin d'optimiser le temps de calcul, nous recherchons une représentation à base de faces planes de façon à pouvoir calculer une matrice de projection pour un ensemble de pixels. Une première solution consiste à utiliser des polyèdres réguliers qui approximent au mieux la sphère. L'intérêt d'utiliser des polyèdres réguliers est qu'un seul mode de calcul est utilisé quelle que soit la face pour calculer la projection d'une image. Une autre solution est d'utiliser des polyèdres non réguliers, c'est-à-dire construits à partir d'un ou de plusieurs type de polygones.

2.4. Représentation plane

Nous regroupons sous le terme « représentation plane », différentes représentations d'un panorama permettant de visualiser l'ensemble de la scène en une seule image. Ce type de représentation est réalisé à partir d'une projection par développement. Comme il n'est pas possible d'étendre la surface d'une sphère, ou d'un ellipsoïde en général, sur un plan sans déchirure ni déformation, la représentation ne pourra pas être rigoureuse. La représentation la plus utilisée est la projection de la scène sur un cylindre que l'on va dérouler. Nous allons rapidement présenter quelques unes des ces projections ayant un intérêt dans le cas d'une mosaïque d'images. Nous présentons en annexe d'autres constructions, issues du monde de la cartographie, esthétiquement très réussie mais dont l'intérêt pour la visualisation d'un panorama est plus discutable.

Afin de visualiser les déformations, nous avons dessiné les méridiens et les parallèles par pas de 30°. Notre caméra ne nous permettant pas de réaliser un panorama complet, le pilotage de l'angle de tangage est compris en –30° et 90°. Les projections que nous présentons ont donc une bande noire pour les angles de tangage inférieurs à –30°.

2.4.1. Projection cylindrique

La projection cylindrique est le mode de représentation le plus couramment utilisé pour la visualisation. Nous verrons par la suite que pour l'analyse, les auteurs préfèrent utiliser une projection sur les faces d'un cube. Cependant, pour la visualisation, la projection cylindrique permet de visualiser l'ensemble de la scène en une seule image. Les déformations sont bien sûr importantes mais l'œil humain et surtout le cerveau arrivent à reconstituer mentalement la scène. Le principe de ce type de projection consiste à projeter la sphère sur un cylindre tangent à la sphère.

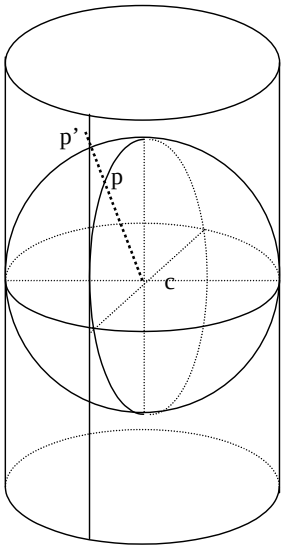


Figure 32 : Principe de la projection cylindrique

2.4.1.1 Projection cylindrique classique

L'équation de la projection cylindrique classique est la suivante :

$$\begin{cases} X = \theta \\ Y = \tan\varphi \end{cases} \tag{2.25}$$

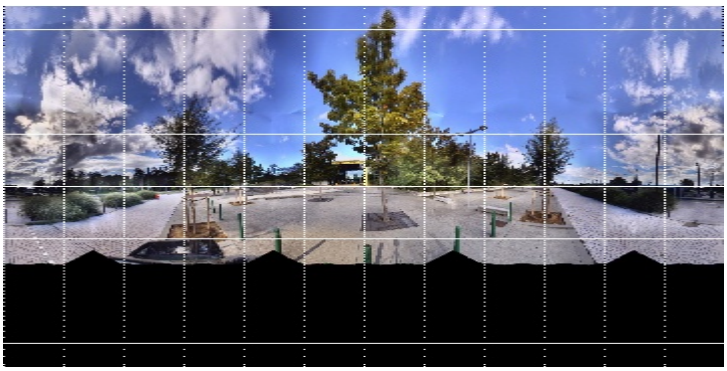


Figure 33 : Projection cylindrique avec –63° < φ < 63°

Visuellement cette solution est intéressante puisqu'elle permet en une seule image de représenter l'ensemble de la scène. Cependant, elle engendre deux inconvénients principaux. Le premier est qu'elle déforme les lignes droites. Le deuxième est qu'elle ne permet pas une définition optimale pour des angles de tangage proche des pôles. Au delà de 60°, le coût de stockage d'un stéradian devient trop important.

2.4.1.2 Projection plate carré cylindrique équidistante (PPCE)

Cette projection a été utilisée pour la première fois par Anaximandre vers 550 avant notre ère. L'équation utilisée est donnée ci-dessous. Comme nous pouvons le remarquer elle est extrêmement simple à mettre en œuvre :

$$\begin{cases} X &= \theta \\ Y &= \varphi \end{cases} \tag{2.26}$$

Par rapport à la projection cylindrique classique, celle-ci permet une représentation des pôles pour un coût de stockage raisonnable. Cependant, elle déforme également les droites horizontales. Parmi les représentations cylindriques, c'est la représentation qui est généralement utilisée pour représenter une mosaïque d'images.

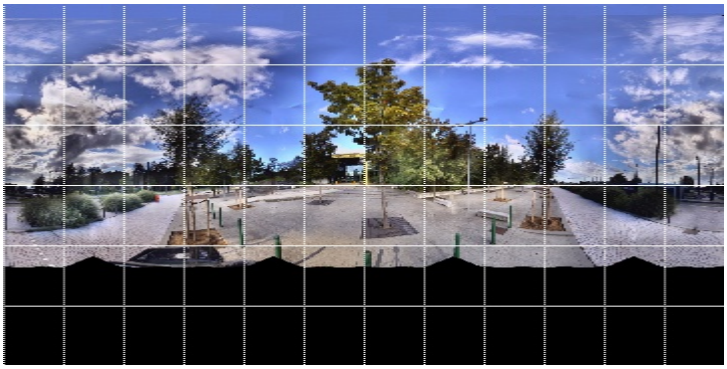


Figure 34 : Représentation cylindrique équidistante

En lien avec la cartographie, nous présentons en annexe deux autre projections cylindriques. Une projection équivalente, celle de Lambert et une conforme, celle de Mercator. Une projection équivalente permet de conserver les surfaces alors qu'une projection conforme permet de conserver les angles.

2.4.2. Projection azimutales

Les projections azimutales sont des projections au sens mathématique du terme, qui transforment les méridiens en rayons également espacés. La projection s'opère à partir d'un centre de projection sur un plan tangent à la sphère.

2.4.2.1 Projection gnomonique

Cette projection n'est ni conforme, ni équivalente, c'est à dire qu'elle ne conserve respectivement ni les angles ni les surfaces. Elle est réalisée en projetant la sphère sur un plan Π tangent en un point p de la sphère et en utilisant la projection radiale centrée sur le centre de la sphère c. Dans le cas d'une représentation de la terre, si le pôle nord est le point de tangence alors l'équateur est renvoyé à l'infini. Cette représentation ne pourra pas être utilisée pour représenter la scène complète en une seule image. Par contre, elle est souvent utilisée dans la navigation parce que l'une des ses propriétés intéressante est qu'elle conserve les droites. Une droite sur la carte correspond à une droite sur le terrain.

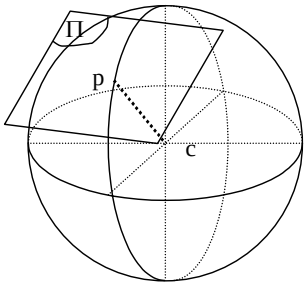


Figure 35 :Principe de la projection gnomonique

La formule de la projection gnomonique est donnée ci dessous :

$$\begin{cases} X = \dfrac{\cos\varphi \cdot \sin(\theta-\theta_0)}{\sin\varphi_0 \cdot \sin\varphi + \cos\varphi_0 \cdot \cos\varphi \cdot \cos(\theta-\theta_0)} \\ Y = \dfrac{\cos\varphi_0 \cdot \sin\varphi - \sin\varphi_0 \cdot \cos\varphi \cdot \cos(\theta-\theta_0)}{\sin\varphi_0 \cdot \sin\varphi + \cos\varphi_0 \cdot \cos\varphi \cdot \cos(\theta-\theta_0)} \end{cases} \quad (2.27)$$

où (θ_0, φ_0) sont les coordonnées du point p de tangence du plan et la de sphère. La figure suivante donne une représentation de la projection gnomonique pour ($\theta_0 = 0°$, $\varphi_0 = 0°$) :

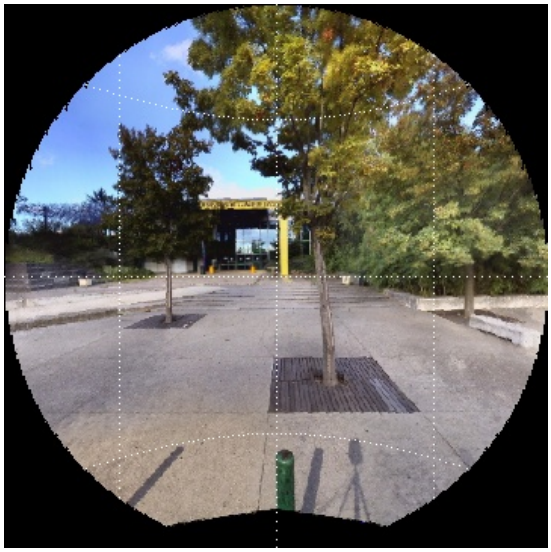


Figure 36 : Projection gnomonique avec ($\theta_0 = 0°$, $\varphi_0 = 0°$)

2.4.2.2 Projection stéréographique

Le principe de la projection stéréographique est le même que celui de la projection gnomonique. Dans le cas de la projection stéréographique, le centre c de projection est aux antipodes du point p de tangence. De plus c'est une transformation conforme, c'est à dire qu'elle conserve les angles.

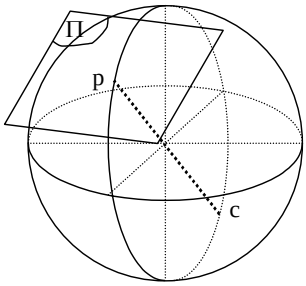


Figure 37 : Principe de la projection stéréographique

La formule de la projection stéréographique est donnée ci dessous :

$$\begin{cases} X = 2 \cdot \dfrac{\cos\varphi \cdot \sin(\theta - \theta_0)}{1 + \sin\varphi_0 \cdot \sin\varphi + \cos\varphi_0 \cdot \cos\varphi \cdot \cos(\theta - \theta_0)} \\ Y = 2 \cdot \dfrac{\cos\varphi_0 \cdot \sin\varphi - \sin\varphi_0 \cdot \cos\varphi \cdot \cos(\theta - \theta_0)}{1 + \sin\varphi_0 \cdot \sin\varphi + \cos\varphi_0 \cdot \cos\varphi \cdot \cos(\theta - \theta_0)} \end{cases} \quad (2.28)$$

où (θ_0, φ_0) sont les coordonnées du point p de tangence du plan et de la sphère. Dans le cas d'une représentation de la scène vue par la caméra, l'intérêt de cette représentation est qu'elle permet de visualiser l'ensemble de la scène à partir de 2 images. La figure suivante donne une représentation de la projection stéréographique pour ($\theta_0 = 0°$, $\varphi_0 = 0°$) et ($\theta_0 = 180°$, $\varphi_0 = 0°$).

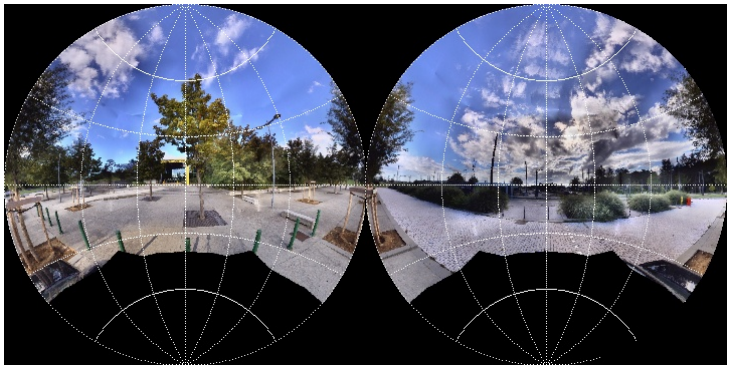


Figure 38 : Projection stéréographique pour ($\theta_0 = 0°$, $\varphi_0 = 0°$) et ($\theta_0 = 180°$, $\varphi_0 = 0°$)

La projection stéréographique a été imaginée 130 ans avant J.C. par Hipparque qui lui donna le nom de *planisphère*. C'est en 1643 que le jésuite François Aiguillon la renomma *projection stéréographique*.

2.4.2.3 Projection orthographique

Le principe de la projection orthographique est le même que les deux précédentes. Dans ce cas, le centre de projection est rejeté à l'infini. Cette projection n'est ni conforme, ni équivalente. L'équation de la projection orthographique est la suivante :

$$\begin{cases} X = \cos\varphi \cdot \sin(\theta - \theta_0) \\ Y = \cos\varphi_0 \cdot \sin\varphi - \sin\varphi_0 \cdot \cos\varphi \cdot \cos(\theta - \theta_0) \end{cases} \quad (2.29)$$

Cette représentation est souvent utilisée pour plaquer une texture sur une sphère. Elle permet de donner un « *effet 3D* » à l'image.

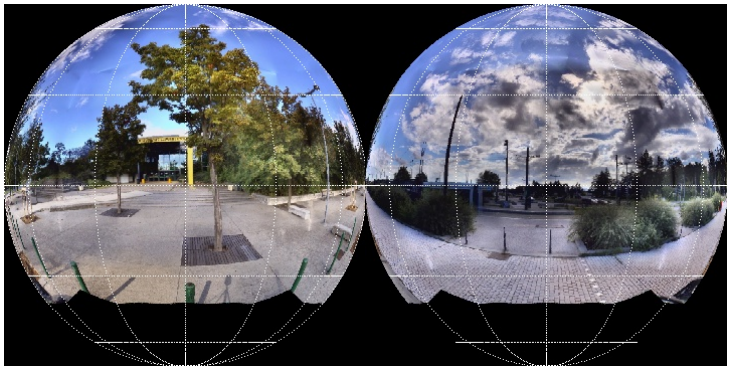


Figure 39 : Projection orthographique pour (θ_0 = 0°, φ_0 = 0°) et (θ_0 = 180°, φ_0 = 0°)

2.4.3. Conclusion

Les projections planes apportent une solution élégante et pertinente pour représenter une sphère sur un plan. Comme nous l'avons vu, tout au long des chapitres précédents, il existe un grand nombre de projections. Cependant, pour représenter une mosaïque d'images, les projections utilisées habituellement se limitent à la projection cylindrique classique avec le problème de la représentation des pôles et la projection plate carré cylindrique équidistante (PPCCE) proposée par Anaximandre. Ces projections présentent néanmoins quelques inconvénients.

Tout d'abord le coût de stockage pour une représentation sans perte d'informations est relativement importants. Le stockage en mémoire vive n'est aujourd'hui plus vraiment un problème. Par contre le transfert du panorama à travers le réseau internet ou autre peut être un frein. De même, il n'en reste pas moins que la quantité d'informations à traiter est importante ce qui peut être pénalisant pour le traitement en temps réel. Nous pouvons calculer rapidement l'espace mémoire nécessaire pour stocker un panorama complet sur une PPCCE. Dans le cas d'une PPCCE, le cylindre est tangent à la sphère, c'est à dire que les pixels de l'équateur sont représentés sans déformation. Il en va de même pour les pixels le long du méridien de référence. Tous les autres pixels sont dilatés par la transformation mathématique. Donc les incréments $\Delta\theta$ et $\Delta\varphi$ des angles respectivement θ et φ permettant une représentation limitant les pertes d'informations correspondent à l'angle solide du pixel situé sur l'axe de projection. Une approximation de ces angles est donnée par :

$$\theta = \varphi = 2 \cdot \tan^{-1}\left(\frac{1}{2 \cdot f}\right)$$

Une PPCE couvrant une scène complète de 2π x π nécessite une matrice rectangulaire de résolution $\frac{2.\pi}{\Delta\theta}$ x $\frac{\pi}{\Delta\varphi}$. Le calcul numérique montre que le nombre de pixels de cette matrice est équivalent à l'expression suivante pour f > 50up :

$$nb\ pixels = 19.7\ f^2$$

où la distance focale f est donnée en unité pixel. Cette valeur est à comparer avec la borne théorique $4\pi f^2$, soit un surcoût de 57%.

Prenons le cas le plus favorable, c'est à dire avec une distance focale faible f = 4.1mm et un capteur 1/3". En unité pixel, la distance focale f = 800up. Un panorama représenté par une PPCCE nécessite environ 12.6 10^6 pixels. Un pixel couleur étant généralement codé sur 3 octets, l'espace de stockage nécessaire pour représenter une PPCCE est d'environ 38MO.

L'autre inconvénient des représentations sur forme de planisphère est le temps de calcul. Pour chaque pixel d'une image il faut tout d'abord calculer la valeur des angles θ_{pi} et φ_{pi} en fonction des conditions de prise de vue puis de calculer la projection du pixel dans l'image panoramique. Toutes ces transformations sont à base de fonctions trigonométriques assez lourdes en temps de calcul. Comme nous l'avons vu précédemment, lorsque le centre de projection est confondu avec le centre des rotations, la transformation qui relie deux images est une homographie. Nous pouvons donc plaquer une image sur une face de polyèdre à l'aide d'une homographie. Une fois le calcul de la transformation effectué nous pouvons donc appliquer cette transformation à tous les pixels de l'image. Nous allons étudier à présent quelques polyèdres.

2.5. Projection polyédrique

Nous appelons projection polyédrique l'opération qui consiste à plaquer les images non pas sur une sphère mais sur une ou plusieurs faces d'un polyèdre à l'intérieur duquel la sphère est inscrite. La transformation reliant une face du polyèdre et l'image correspond à une homographie 2D. Comme nous l'avons évoqué, l'intérêt de cette transformation est qu'elle s'applique à tous les pixels d'une même image. De plus le stockage de surface plane est plus adapté à la mémorisation des données que les surfaces sphériques. L'idéal étant d'utiliser des faces carrés ou rectangulaires. Le calcul de l'adresse d'un pixel particulier à partir de ses coordonnées est alors triviale. Il existe un nombre important de polyèdres permettant d'approximer une sphère. Nous allons tout d'abord étudier les cinq polyèdres réguliers, c'est à dire les cinq polyèdres formés à partir d'un même polygone. Nous verrons ensuite un exemple de polyèdre semi régulier particulièrement intéressant puis une dernière représentation à base de polygones quelconques. La forme obtenue n'est plus un polyèdre mais nous la traitons dans cette catégorie.

2.5.1. Les solides de Platon

Les polyèdres réguliers sont au nombre de 5 et sont également appelés « solides de Platon » en référence au philosophe grec qui les a décrits au 5ième siècle avant notre ère. Ils sont dits *régulier*s parce qu'ils sont formés à partir de polygones réguliers isométriques. Un polygone régulier a tous ses côtés isométriques et tous ses angles de même mesure. Dans l'ouvrage « le timée », Platon décrit les cinq solides « parfaits » en ces termes : « … *La première chose à expliquer ensuite, c'est la forme que chacun d'eux a reçue et la combinaison de nombres dont elle est issue. Je commencerai par la première espèce, qui est composée des éléments les plus petits. Elle a pour élément le triangle (…) qui est équilatéral. Quatre de ces triangles équilatéraux réunis selon trois angles plans forment un seul angle solide, qui vient immédiatement après le plus obtus des angles plans. Si l'on compose quatre angles solides, on a la première forme de solide, qui a la propriété de diviser la sphère dans laquelle il est inscrit en parties égales et semblables. La seconde espèce est composée des mêmes triangles. Quand ils ont été combinés pour former huit triangles équilatéraux, ils composent un angle solide unique, fait de quatre angles plans. Quand on a construit six de ces angles solides, le deuxième corps se trouve achevé. Le troisième est formé de la combinaison de deux fois dix triangles élémentaires, c'est-à-dire de douze angles solides, dont*

chacun est enclos par cinq triangles plans équilatéraux, et il y a vingt faces qui sont des triangles équilatéraux. (…) Six de ces quadrangles, en s'accolant, ont donné naissance à huit angles solides, composés chacun de trois angles plans droits, et la figure obtenue par cet assemblage est le cube (…) Il restait encore une cinquième combinaison. Dieu s'en est servi pour achever le dessin de l'univers. » Platon, Timée, 54d-55d.

Euclide termina son oeuvre « Les Eléments » en prouvant qu'il existe exactement 5 polyèdres convexes réguliers : le tétraèdre, le cube, l'octaèdre, le dodécaèdre et l'icosaèdre. A travers Platon, les grecs ont accordé une signification mystique aux cinq solides réguliers en les rattachant aux grandes entités qui selon eux façonnaient le monde : le feu est associé au tétraèdre, l'air à l'octaèdre, la terre au cube, l'univers au dodécaèdre et l'eau à l'icosaèdre. Lorsqu'il publie «Mysterium Cosmographicum » en 1596, Kepler propose un modèle de représentation du système solaire composé des 5 solides de Platon. Dans son modèle, les trajectoires des planètes du système solaire connues à l'époque s'inscrivent dans une sphère dont le centre est occupé par le soleil. Chaque sphère est inscrite dans un polyèdre et circonscrite à un autre. La planète Mercure étant la planète la plus proche du soleil, son orbite est donc inscrite dans une première sphère. Kepler montre alors que cette première sphère est inscrite dans un octaèdre lui même circonscrit à une deuxième sphère et que l'orbite de Venus est justement inscrite dans cette deuxième sphère. Il associe donc Vénus à l'octaèdre, la Terre à l'icosaèdre, Mars au dodécaèdre, Jupiter au tétraèdre et Saturne au cube. Nous allons étudier la projection de la sphère sur ces cinq polyèdres réguliers par ordre croissant du nombre de faces. Le premier d'entre eux est donc le tétraèdre.

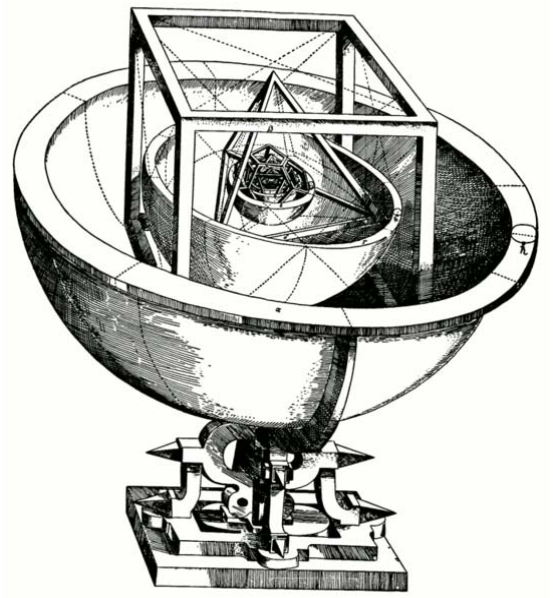


Figure 40 : Modèle de l'univers de Kepler

2.5.1.1 Le tétraèdre

Dans la cosmogonie Platonicienne, le feu est représenté par un tétraèdre régulier constitué de 4 triangles équilatéraux (les plus beaux selon Platon), provenant eux mêmes de la réunion de 6 triangles rectangles. Loin de ces considérations métaphysiques, notre problématique consiste à plaquer les images d'une caméra PTZ sur les 4 faces du tétraèdre et à étudier l'espace mémoire nécessaire pour mémoriser la construction. Comme nous l'avons précisé précédemment, si nous voulons obtenir une représentation sans perte d'information, nous devons utiliser le polyèdre dont la sphère est tangente aux faces. Dans le cas du tétraèdre, le rayon de la sphère inscrite est donné par :

$$r=\frac{a\sqrt{6}}{12}$$

où *a* est longueur d'une arête, et la surface d'un tétraèdre est donnée par :

$$s=\sqrt{3}\cdot a^2$$

L'espace de stockage nécessaire pour la représentation est égale à la surface du polyèdre. Comme dans notre cas, le rayon de la sphère correspond à la distance focale, l'espace de stockage optimal peut donc s'exprimer directement en fonction de la distance focale à partir de l'expression suivante :

$$s \approx 41.5 \cdot f^2$$

Le résultat de la projection d'une sphère sur un tétraèdre est donnée ci-dessous :

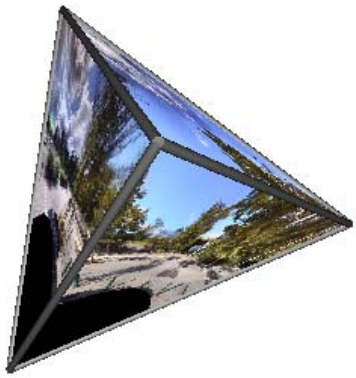


Figure 41 : Exemple de projection d'une sphère sur un tétraèdre

2.5.1.2 Le cube

Si nous reprenons la cosmologie Platonicienne, le cube représente la Terre en raison de sa stabilité. En ce qui nous concerne, le cube est le polyèdre le plus facile à utiliser puisque ses 6 faces sont carrées. Le stockage ne pose donc pas de soucis particuliers. De même, l'orientation des 6 faces est triviale, ce qui rend les calculs plus simples. C'est sans surprise le polyèdre généralement utilisé par les différents auteurs, voire le seul. Cependant, il n'est pas le polyèdre le mieux adapté. Son principal défaut est que l'angle entre les faces est important, ce qui peut entraîner des défauts lors de la restitution d'une image. Nous y reviendrons par la suite. Dans le cas du cube, le rayon de la sphère inscrite est donné par :

$$r=\frac{a}{2}$$

où *a* est la longueur d'une arête. La surface du cube est donnée par :

$$s=6\cdot a^2$$

En tenant compte des remarques précédentes, nous obtenons un espace de stockage correspondant à :

$$s = 24 \cdot f^2$$

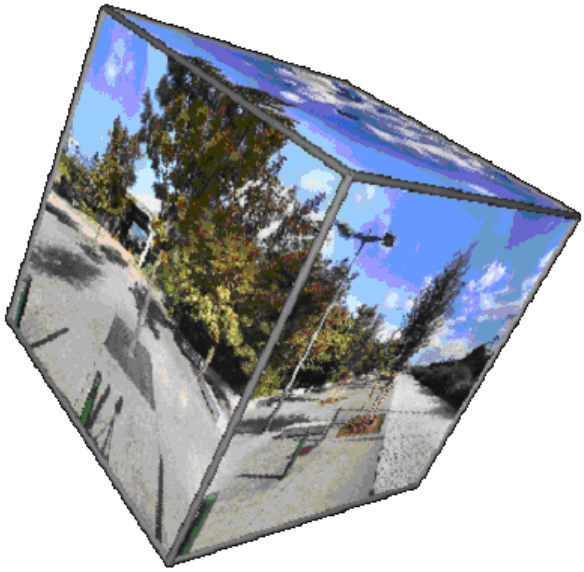


Figure 42 : Exemple de projection d'une sphère sur un cube

Nous reviendrons plus en détail sur la construction panoramique à l'aide d'un cube par la suite.

2.5.1.3 L'octaèdre

L'octaèdre est formé de 8 triangles équilatéraux et il était associé à l'air par les Platoniciens. Pour le stockage nous pouvons utiliser le même raisonnement que pour le tétraèdre, c'est-à-dire un bandeau rectangulaire où nous plaçons les 7 triangles en quinconce et le 8ième coupé en deux et placé à chaque extrémité. Comme pour le tétraèdre, si l'on ne procède pas à cette optimisation, il est nécessaire d'utiliser 9 triangles pour le stockage.

Le rayon de la sphère inscrite dans l'octaèdre est donné par :

$$r = \frac{a}{\sqrt{6}}$$

où a est la longueur d'une arête. La surface de l'octaèdre, exprimé en fonction de la distance focale est donné par :

$$s \approx 20.8 \cdot f^2$$

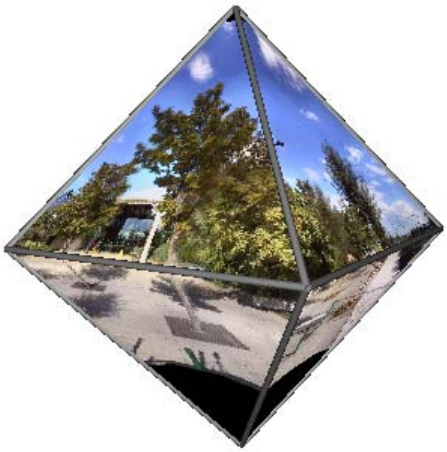


Figure 43 : Exemple de projection d'une sphère sur un octaèdre

2.5.1.4 L'icosaèdre

L'icosaèdre est formé de 20 triangles équilatéraux et était associé à l'eau par les Platoniciens.

Le rayon de la sphère inscrite dans l'icosaèdre est donné par :

$$r=\frac{a\cdot\phi^2\cdot\sqrt{3}}{6}$$

où ϕ est le nombre d'or $\phi=\frac{1+\sqrt{5}}{2}$ et *a* est la longueur d'une arête. La surface de l'icosaèdre est donné par :

$$s \approx 15.2 \cdot f^2$$

2.5.1.5 le dodécaèdre pentagonal

Le dodécaèdre pentagonal est le cinquième et dernier solide de Platon. Il représente l'univers. Il est formé de 12 pentagones réguliers. Le rayon de la sphère inscrite est donnée par :

$$r=\frac{1}{2}\cdot\sqrt{\frac{\phi^5}{\sqrt{5}}}\cdot a$$

où ϕ est le nombre d'or et *a* est la longueur d'une arête. La surface du dodécaèdre pentagonal exprimée en fonction de la distance focale est donnée par :

$$s \approx 16.7 \cdot f^2$$

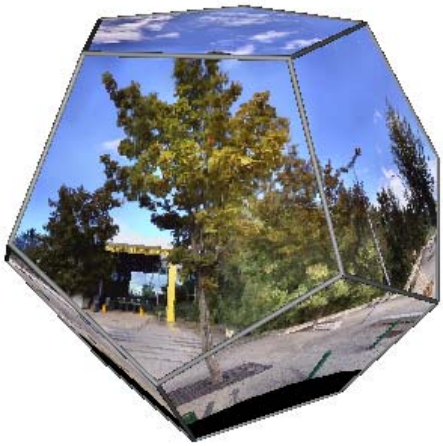


Figure 44 : Exemple de projection d'une sphère sur un dodécaèdre pentagonal

2.5.2. Polyèdre semi-régulier

La définition d'un polyèdre semi régulier est la suivante : « *Un polyèdre semi-régulier est un polyèdre tel que pour tout couple de sommets, il existe une isométrie conservant globalement le polyèdre et transformant un sommet en l'autre, autrement dit, tel que le groupe des isométries du polyèdre agit transitivement sur l'ensemble des sommets.* » Pour la projection d'une sphère sur un polyèdre, nous cherchons un polyèdre qui approxime le mieux possible la sphère tout en limitant à la fois le nombre de faces et le nombre de polygones différents utilisés pour la construction. L'idéal est de n'avoir qu'une forme de polygone. Comme nous l'avons vu, il n'y a que cinq polygones qui répondent à cette demande. Trois d'entre eux sont à base de triangles, ce qui ne facilite pas le stockage. Un est à base de carrés ce qui est le plus intéressant et un est à base de pentagones. Dans ce dernier cas, le plus simple pour le stockage consiste à mémoriser le pentagone de chaque face dans un rectangle englobant. Nous n'allons pas passer en revue tous les polyèdres semi réguliers mais nous allons nous intéresser à l'icosaèdre tronqué. Il est formé à partir de 12 pentagones et 20 hexagones. L'icosaèdre tronqué, répond bien à nos exigences. Il approxime assez bien la sphère, le nombre de faces reste raisonnable et il n'est composé de deux types de polygone. Il existe des structures de cette forme dans la nature ou dans la vie courante. Les chimistes, par exemple, connaissent la molécule de buckminsterfullerène composée de 60 atomes de carbone et dont la structure est en forme d'icosaèdre tronqué. Cependant, ce polyèdre est surtout connu des terrains de football puisque c'est à partir de cette structure que l'on forme les ballons de foot.

Contrairement aux polyèdres réguliers que nous avons déjà étudiés, il n'est pas possible d'obtenir une sphère inscrite qui soit à la fois tangente aux faces des pentagones et tangente aux faces des hexagones. Le calcul du rayon de la sphère inscrite tangente aux faces des pentagones est donné par :

$$r_p = \frac{\sqrt{1250+410\cdot\sqrt{5}}}{20}\cdot a_p$$

et le rayon de la sphère inscrite et tangente aux faces des hexagones est donné par :

$$r_h = \frac{\sqrt{3}\cdot\left(3+\sqrt{5}\right)}{4}\cdot a_h$$

Pour mémoriser l'ensemble de la scène sans perte d'informations, nous devons nous mettre dans les conditions les plus défavorables, c'est à dire utiliser le calcul qui maximise la taille de l'arête. La taille d'arête la plus grande implique automatiquement un polyèdre plus gros mais nous assure un stockage sans pertes. Nous devons donc utiliser la sphère tangente aux pentagones.

La surface de l'icosaèdre tronqué est donnée par :

$$s \approx 13.4 \cdot f^2$$

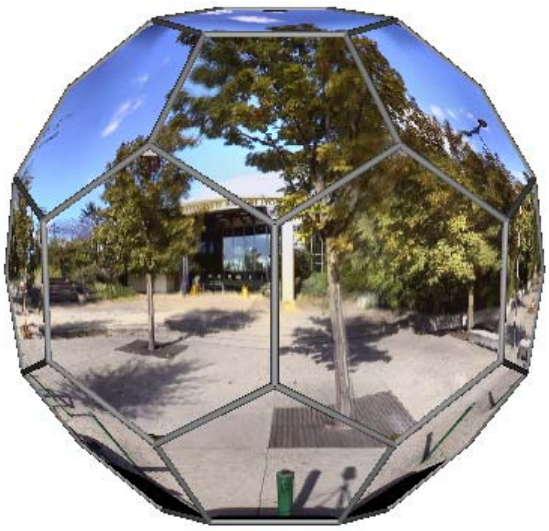


Figure 45 : Exemple de projection d'une sphère sur un icosaèdre tronqué

2.5.3. Polygones quelconques

Parmi tous les polyèdres que nous avons présentés, seul le cube présente l'avantage d'avoir une structure de données en parfaite adéquation avec le stockage. Pour les autres, il faut soit disposer les faces d'une certaine manière soit limiter la complexité algorithmique au détriment de l'espace de stockage. Finalement, pourquoi ne pas mémoriser que des rectangles et trouver le recouvrement optimal de la sphère qui minimise le nombre d'images en limitant les chevauchements entre les images connexes. Nous retrouvons ici une autre application du calcul de la trajectoire optimale que nous avons abordée précédemment. Les polygones utilisés sont alors quelconques ; le plan qui les contient est le plan tangeant à la sphère au point ayant pour coordonnée sphérique θ et φ. Une arête « verticale » est créée entre deux polygones successif d'une même bande horizontale. L'espace de stockage est alors simplement égal à la somme de toutes les images nécessaires au recouvrement.

A partir de l'algorithme de balayage par bandes, nous présentons ci-dessous quelques uns des polygones obtenus.

Figure 46 : Polygones de la bande sphérique horizontale à φ_0 = -30°

Pour une focale de 4.1mm (800 en pixels), un total de 44 images a été nécessaire. Le recouvrement de la sphère est donnée ci-dessous.

Figure 47 : Exemple de projection d'une sphère sur des polygones quelconques

Il faut noter que l'objet construit n'est pas forcement un polyèdre.

2.6. Comparatif du nombre de pixels nécessaire aux représentations panoramiques

Nous proposons un tableau récapitulatif du nombre de pixels nécessaires à la construction des principales représentations panoramiques que nous avons présentées dans les paragraphes précédents. Pour chacune de ces représentations panoramiques, le nombre de pixels est calculé de façon à minimiser la perte d'information entre l'acquisition et la construction en fonction de la résolution de l'image et de la distance focale. Dans le cas de l'algorithme de parcours optimal, nous calculons l'espace nécessaire à la mémorisation de l'ensemble des images même si une partie de l'information est dupliquée du fait du recouvrement partiel entre les images, ceci afin de limiter la complexité algorithmique. Le rapport entre la surface et la distance focale au carré que nous présentons est simplement un indicateur de la qualité d'une représentation basée sur la place mémoire utilisée. Un autre critère que nous avons rapidement abordé et qui n'apparaît pas dans le tableau est le temps d'accès aux données.

Représentation	Nom	Type de polygones	Surface / f^2
Sphère	-	-	12,6
Plane	PPCCE	Rectangle 360°x180°	19,7
Polyèdre	Tétraèdre	4 triangles	41,6
	Cube	6 carrés	24,0
	Octaèdre	8 triangles	20,8
	Dodécaèdre pentagonal	12 pentagones	16,7
	Icosaèdre	20 triangles	15,2
	Icosaèdre tronqué	12 pent. + 20 hex.	13,4
Parcours optimal	f =800up	44 images	21,1
	f = 1600up	139 images	16,7
	f = 3200up	488 images	14,6
	f = 6400up	1823 images	13,8
	f = 12800up	7100 images	13,3

Tableau 3 : Comparaison du nombre de pixels nécessaires à la construction de plusieurs représentations panoramiques

A la lecture de ce tableau, nous pouvons remarquer que l'icosaèdre tronqué est la structure qui nécessite le moins de pixels pour représenter une scène. En toute rigueur, l'espace minimum est donné par l'algorithme de parcours optimal avec une focale de 12800 up ce qui correspond approximativement à une focale de 64mm pour un capteur 1/3". Toutefois, avec une telle distance focale, nous utiliserions 100TO de mémoire pour représenter le panorama complet en couleur.
Nous pouvons également remarquer que les deux représentations les plus utilisées, c'est à dire la PPCCE et le cube sont globalement équivalente. La PPCCE occupe un espace un peu moins important mais nécessite un temps de calcul plus lourd.

2.7. Multi résolution

2.7.1. Avant propos

L'aspect multi-résolution est un point que nous n'avons pas encore abordé. Cependant il peut être considéré comme un paramètre intrinsèque de la construction d'un panorama comme de la visualisation. Nous avons vu dans le chapitre précédent que sous certaines conditions, nous pouvons déterminer l'homographie entre deux plans correspondant à deux prises de vue. Cette homographie peut être calculée à partir des paramètres intrinsèques et extrinsèques des deux prises de vue. Parmi les paramètres intrinsèques, la distance focale, exprimée en pixel, correspond justement à la résolution. Nous pouvons donc calculer l'homographie entre deux images dont la longueur focale, c'est à dire la résolution, n'est pas la même. Nous traitons donc implicitement l'aspect multi-résolution. Ce même constat s'applique lors de la visualisation. Si le panorama est projeté sur un polyèdre, nous pouvons reconstruire une image correspondant à une prise de vue quelconque. La limite étant, cette fois ci, la résolution avec laquelle est construit le panorama. Si la distance focale de l'image à construire est plus faible que celle du panorama, l'image pourra être construite sans perte d'information. En revanche, si la distance focale de l'image à construire est plus grande que celle du panorama, l'image ne pourra être construite que par interpolation.

2.7.2. Problématique

Nous avons présenté plusieurs modes de représentation possibles d'une mosaïque d'images. Toutes ces représentations ont leurs avantages et leur inconvénients. Cependant, elles sont toutes basées sur le principe d'une focale fixe. C'est à dire que l'ensemble de la scène est stockée avec la même résolution. Le logiciel QuickTime VR ainsi que d'autre logiciels du commerce permettent de créer un panorama avec plusieurs résolutions. Cependant, la résolution la plus haute est utilisée pour créer plusieurs panoramas de résolution plus basse. L'intérêt est d'optimiser les calculs lors des visualisations à basse résolution et d'optimiser les transferts sur le réseau.

Figure 48 : Plusieurs résolutions d'un même panorama

Or, nous pouvons imaginer que sur l'ensemble de la scène toutes les zones ne présentent pas le même intérêt. Avec les méthodes que nous avons présentées, si une zone particulièrement intéressante nécessite une certaine résolution, c'est l'ensemble de la scène qui devra être stockée dans cette résolution. Prenons l'exemple du château de Couzan dans la Loire, dont le panorama cylindrique est donné ci-dessous.

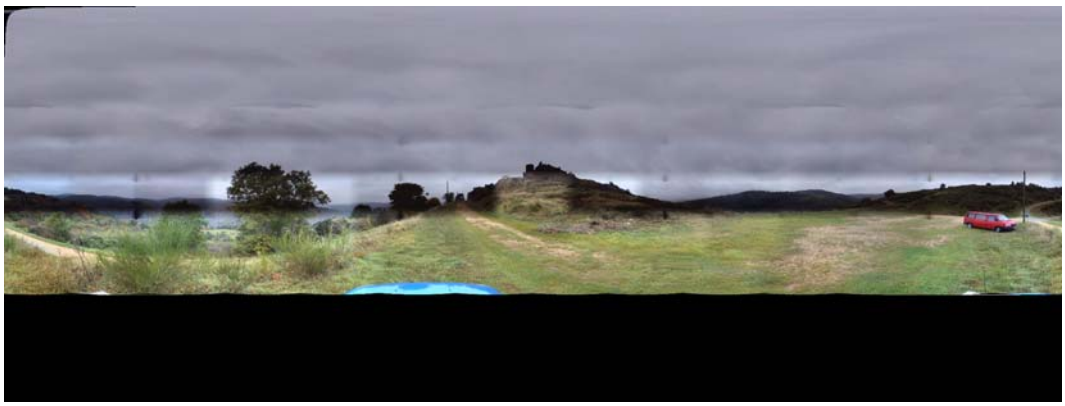


Figure 49 : Panorama cylindrique du château de Couzan

Sur l'ensemble du panorama, le château en lui-même ne représente qu'une toute petite partie au centre de l'image. La scène en elle-même, ne présente que peu d'intérêt. Elle permet simplement de visualiser dans quel contexte se situe le château. Le château en lui même est plus intéressant et dans le cadre d'une application de visualisation, nous aimerions avoir plus de détails, comme par exemple la tour ci dessous.

Figure 50 : Détail de la tour du château de Couzan (focale 50mm)

Afin d'obtenir ce niveau de détail sur la tour, la prise de vue doit être réalisée avec une focale de 50mm. Sur un cube et avec une telle distance focale, l'espace de stockage serait de l'ordre de 2.4GO. En utilisant l'algorithme de parcours optimisé, il faudrait pas moins de 4179 images 640x480, soit 2h15 de temps d'acquisition sur la base d'une image toutes les deux secondes (1 seconde pour le déplacement et 1 seconde de stabilisation). Si théoriquement cette approche est valable, dans la pratique elle montre rapidement ses limites.

2.7.3. Approche proposée

L'approche que nous proposons est une généralisation de la projection du panorama sur des polygones quelconques. Lorsque nous avons présenté cette projection, nous avons considéré implicitement que toutes les images avaient été acquises avec la même distance focale. Or, nous pouvons très bien garder la même structure de données mais en utilisant des images dont la distance focale est différente. Dans le chapitre suivant, nous verrons que les données issues de la caméra ne permettent pas toujours de calculer précisément l'homographie entre les images. Une étape de recalage d'image est alors nécessaire pour obtenir un panorama robuste. Cette étape de recalage nécessite un certain recouvrement entre les image qui

implique une multiplication des prises de vue. Comme nous ne souhaitons pas multiplier le nombre de faces, la solution que nous avons adoptée consiste à regrouper sur un même polygone les images acquises avec la même résolution. Dans la limite, bien sur, où l'homographie entre l'image et le polygone peut être déterminée. Si cela n'est pas le cas, un nouveau polygone est généré.

Dans l'exemple du château de Couzan, nous pouvons construire la scène à partir des images suivantes :

Figure 51 : Ensemble des faces utilisées pour le panorama multirésolution

Dans cet exemple les images (Figure 51e) et (Figure 51f) ne sont pas représentées. Avec les images (Figure 51a), (Figure 51b), (Figure 51c) et (Figure 51d), elles correspondent aux six faces du cube de plus basse résolution. Les images (Figure 51e) et (Figure 51f) correspondent aux deux pôles et ne présentent pas d'intérêt. Les six faces du cubes sont de taille 2534 x 2534 pixels. L'image (Figure 51g) qui correspond à un détail de l'image (Figure 51b) à une résolution de 5068 x 3889 pixels. L'image (Figure 51h) qui représente un détail de l'image (Figure 51g) à une résolution de 2669 x 2669 pixels. L'ensemble du panorama multi résolution occupe donc un espace mémoire de l'ordre de 200MO soit 10 fois moins que la solution traditionnelle.

Pour reconstruire une vue particulière, il suffit de considérer les faces dans lesquelles l'image se projette et dont la distance focale utilisée pour la création est supérieure ou égale à la vue demandée.

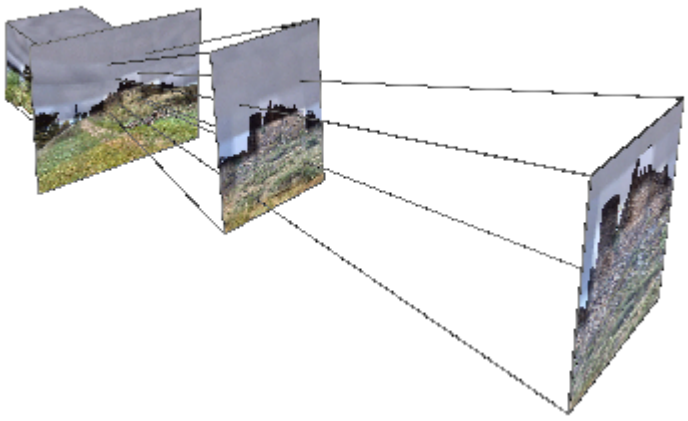


Figure 52 : Exemple de panorama multi-résolution

2.8. Conclusion

Dans ce chapitre, nous avons présenté plusieurs méthodes permettant de représenter un panorama. Toutes ces approches ont leurs avantages et leurs inconvénients et finalement seront choisies en fonction du besoin. Pour une visualisation de l'ensemble de la scène en une seule vue, nous choisirons une projection cylindrique. Habituellement, c'est la projection plate carré cylindrique équidistante qui est utilisée. Si la région d'intérêt de la scène est comprise pour des angles de tangage de ± 60°, la projection cylindrique classique sera privilégiée puisqu'elle permet une dilatation des distance verticale en fonction de l'angle de tangage. Par rapport à la projection précédente, les objets semblent moins tassés. Lorsque nous avons beaucoup de traitements à faire sur l'image panoramique, notamment pour calculer des vues intermédiaires, nous utilisons préférentiellement une représentation sous la forme de polyèdre. Le polyèdre les plus utilisé dans la littérature est le cube. Cependant, nous lui préférons l'icosaèdre tronqué qui utilise la même transformation mathématique mais qui permet de limiter la place mémoire utilisée et le problème de rendu au niveau des coins. Enfin, si la limitation de l'espace mémoire est un critère important, nous utiliserons les images issues du parcours optimisé que nous avons proposé.

Dans ce chapitre, nous avons fait l'hypothèse que les paramètres de la projection sont parfaitement connus. De même, nous avons considéré que les conditions de luminosité ou que le gain de la caméra n'ont pas varié au cours de la prise de vue des différentes images. Nous nous sommes placés dans un cas idéal auquel il est possible de se ramener. Il suffit que la caméra soit suffisamment bien instrumentée (précision et temps de réponse) pour obtenir les paramètres précis de la prise de vue. De même, le gain de la caméra peut être fixé. Nous allons à présent aborder le cas un peu moins idéal et présenter différentes méthodes permettant d'obtenir un panorama robuste.

Mosaïque d'images multi-résolution et Applications

Chapitre 3 : Création d'un panorama robuste en temps réel

3. Création d'un panorama robuste en temps réel

3.1. Objectif

Nous avons présenté dans le chapitre précédent les principes mathématiques de base permettant de calculer et de représenter une mosaïque d'images. Nous avons alors fait l'hypothèse que la caméra correspond exactement au modèle sténopé, que les paramètres intrinsèques et extrinsèques de la caméra sont parfaitement connus et qu'il n'y a pas de changement de luminosité de la scène au cours de l'acquisition. Dans la pratique, nous allons être confrontés à des situations où nous ne sommes pas dans le cas « idéal » . Nous allons donc avoir à résoudre un certain nombre de difficultés. Bien qu'il existe plusieurs solutions à ces différents problèmes, elles sont bien souvent incompatibles avec le temps réel. La définition du temps réel reste une notion subjective. Notre définition du temps réel est le délai qui sépare deux acquisitions. Cela va donc dépendre des applications visées. Dans la mesure où la fréquence d'acquisition classique des caméras est de 25 ou 30 images par seconde, le temps de calcul ne devrait pas excéder respectivement 40ms ou 30ms. Cependant, dans certaines applications que nous présentons, une fréquence de 5 images par seconde suffit, ce qui porte ce délai à 200ms. Si le but final de la création du panorama se limite à la visualisation *a posteriori* de la scène, de nombreux logiciels sont disponibles. Comme nous l'avons indiqué en introduction, certain d'entre eux sont même gratuits. Le logiciel AutoStitch™ que l'on peut se procurer gratuitement sur Internet, permet un recalage précis des images et donne un rendu excellent. Cependant, pour créer un panorama à partir de 72 images couleur de taille 640x480, le temps de calcul est d'environ 65s soit près d'une seconde par image. Pour une application de visualisation, ce délai est très honorable. Cependant, pour des applications en temps réel que nous présentons dans le dernier chapitre, ce temps de calcul est trop important.
Nous avons globalement, 3 défis à relever : corriger le défaut de positionnement de la caméra, compenser les changements de luminosité et supprimer ce que les auteurs appellent les « fantômes ». Pour chacune de ces opérations, nous allons présenter un état de l'art des solutions existantes et dans la mesure du possible, apporter notre contribution.

3.2. Interpolation

Avant toute chose, nous devons résoudre une première difficulté liée à la discrétisation des images numériques. La définition classique d'une image numérique donnée par la géométrie discrète est la suivante. Une image numérique est constituée d'un ensemble de cellules dénombrables appelées pixels dans le cas des images en 2 dimensions. Ces cellules forment une partition de la portion de plan que représente l'image. Ce pavage de l'image est appelé espace discret ou grille discrète. Le centre de gravité de chaque cellule est appelé point discret.

Or, il est peu probable qu'un pixel x_c de l'image courante C se projette, après transformation, exactement sur la grille de l'image panoramique, d'une face du polyèdre ou simplement de l'image précédente P. Il est donc nécessaire de diffuser la valeur du pixel vers les plus proches voisin sur la grille discrète. Classiquement ce problème est résolu par une interpolation bilinéaire en « backward mapping ».

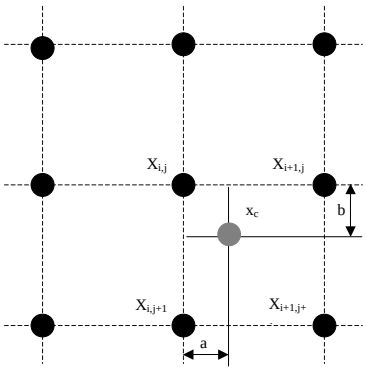


Figure 53 : principe de l'interpolation linéaire

La valeur du pixel x_c est une somme pondéré des quatre pixels les plus proches :

$$x_c=(1-b)\cdot(1-a)\cdot X_{i,j}+a\cdot(1-b)\cdot X_{i,+1j}+b\cdot(1-a)\cdot X_{i,j+1}+a\cdot b\cdot X_{i+1,j+1}$$

Figure 54 : Projection d'une image sans interpolation

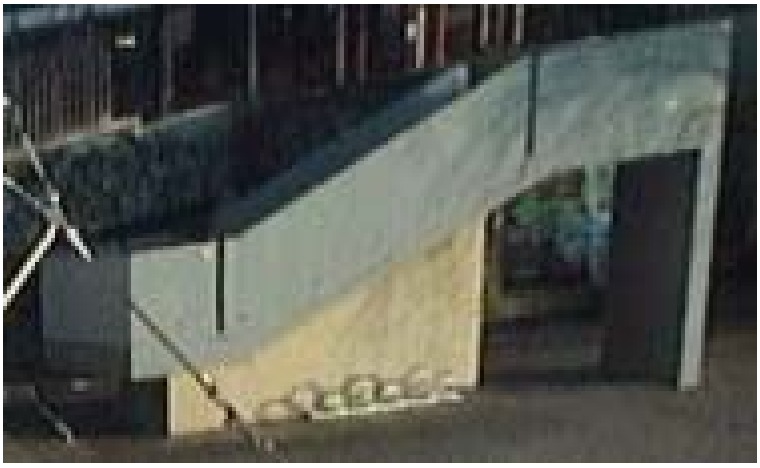


Figure 55 : Projection de la même image avec interpolation bilinéaire

La première image correspond à une projection sans interpolation. Nous pouvons remarquer la pixélisation qui apparaît sur l'ensemble de l'image mais dont l'effet est plus sensible au niveau des gradients forts. Par contre, sur la deuxième image réalisée avec une interpolation bilinéaire, les gradients forts sont légèrement lissés.

Il existe d'autre type d'interpolation comme l'interpolation bicubique ou l'interpolation au plus proche voisin. L'intérêt de l'interpolation bilinéaire est qu'elle est simple à calculer et donne de bons résultats dans l'ensemble, même si elle laisse apparaître des discontinuités au niveau des dérivées de l'image obtenue. Par exemple, le motif suivant a une résolution de 3 x 3:

Si nous augmentons cette résolution pour la passer à 21 x 21 avec une interpolation bilinéaire, nous obtenons l'image suivante (Figure 56) :

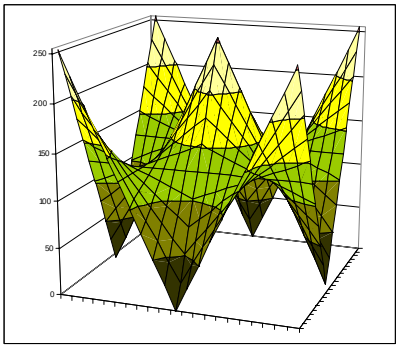


Figure 56 : Représentation 3D du motif après une interpolation bilinéaire

Sur cette représentation 3D de l'image, nous pouvons constater les discontinuités de la dérivée au niveau des sommets. Avec une interpolation bicubique d'ordre 1, nous obtenons l'image suivante (Figure 57):

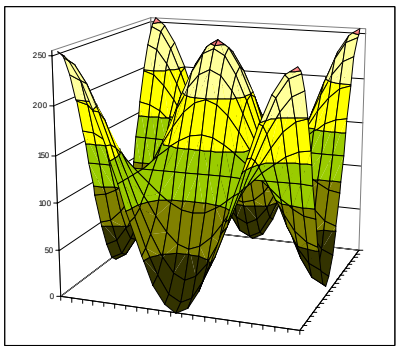


Figure 57 : Représentation 3D du motif après une interpolation bilinéaire d'ordre 1

L'interpolation bicubique permet un meilleur lissage mais le nombre d'opérations nécessaires est plus important, donc le temps de calcul est plus long. L'interpolation bilinéaire étant plus rapide et donnant des résultats satisfaisants, nous avons utilisé cette interpolation dans nos différentes applications.

3.3. Défaut d'alignement

Avant de réaliser notre premier panorama, nous avons encore à vérifier le domaine de validité du modèle mathématique que nous utilisons. Notre but ici n'est pas de remettre en cause le modèle sténopé. Le domaine de validité de ce modèle a été abondamment commenté dans la littérature, notamment dans l'ouvrage de Faugeras [FAU93] et dans celui de Hartley [HAR04]. Cependant, nous faisons souvent l'hypothèse que l'axe de projection coupe le plan focal au centre de l'image. Dans [SZE94] Szeliski montre que si le point d'intersection est peu éloigné du centre de l'image l'impact de ce décalage n'a pas d'effet significatif. D'autre part, pour justifier l'utilisation des matrices d'homographie 2D entre deux images, nous avons fait également l'hypothèse que le centre de projection est confondu avec le centre des rotations. C'est à dire que le centre de projection ne se déplace pas avec le mouvement de la caméra.

3.3.1. Etude théorique

Dans le cas idéal, le centre de projection Oc de la caméra est confondu avec le centre O des rotations, comme indiqué dans la figure ci-dessous. Pour illustrer notre propos nous étudions simplement le décalage suivant l'axe z et nous considérons que l'angle $\varphi = 0$. Le plan image Π est à une distance f correspondant à la distance focale.

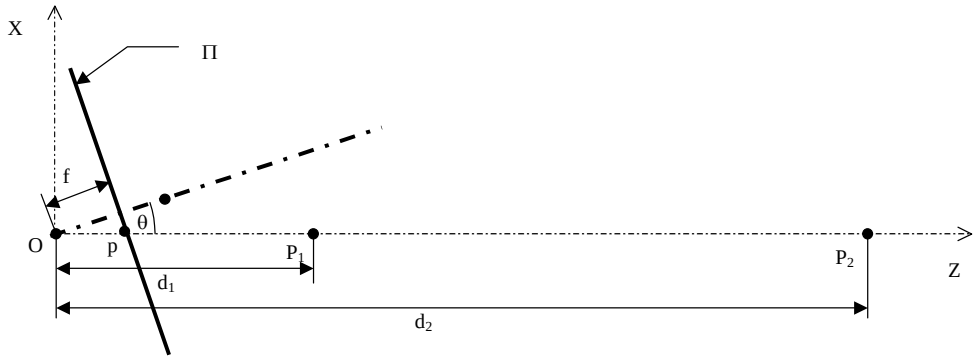


Figure 58 : Centre de projection de la caméra confondu avec le centre des rotations

Si deux points P_1, P_2 ainsi que le centre O sont alignés, quelle que soit la rotation θ (à l'exception des cas dégénérés $\theta = \pi/2$ et $\theta = -\pi/2$) les deux points se projettent en un point p unique sur le plan image.

Maintenant, si le centre de projection Oc est à une distance d du centre O des rotations, pour $\theta = 0$, les points P_1 et P_2 se projettent toujours en un point p du plan image. Par contre, pour une rotation $\theta \neq 0$, les points P_1 et P_2 se projettent respectivement en deux points p_1 et p_2 sur le plan image. L'angle θ_1 est l'angle que fait le rayon $[OcP_1]$ avec l'axe optique et l'angle θ_2 est l'angle que fait le rayon $[OcP_2]$ avec l'axe optique.

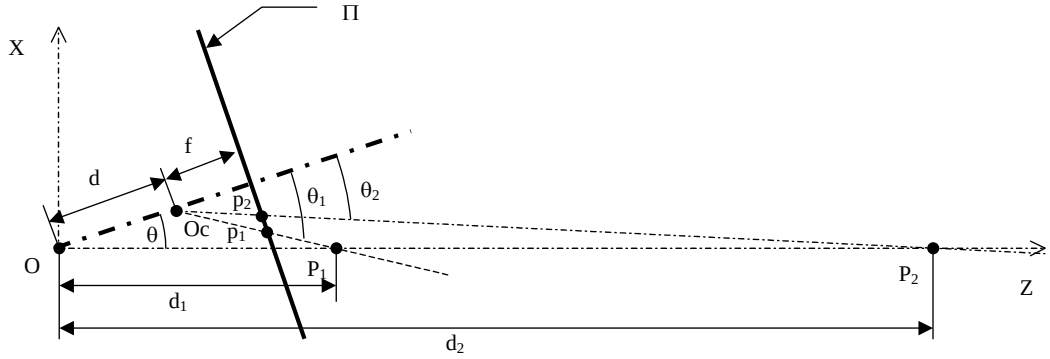


Figure 59 : Décalage positif du centre optique de la caméra par rapport au centre des rotations

Dans le schéma ci-dessus, nous avons arbitrairement placé le centre Oc entre le centre O et le plan focale. Le même schéma s'applique si Oc est placé en arrière du centre O.

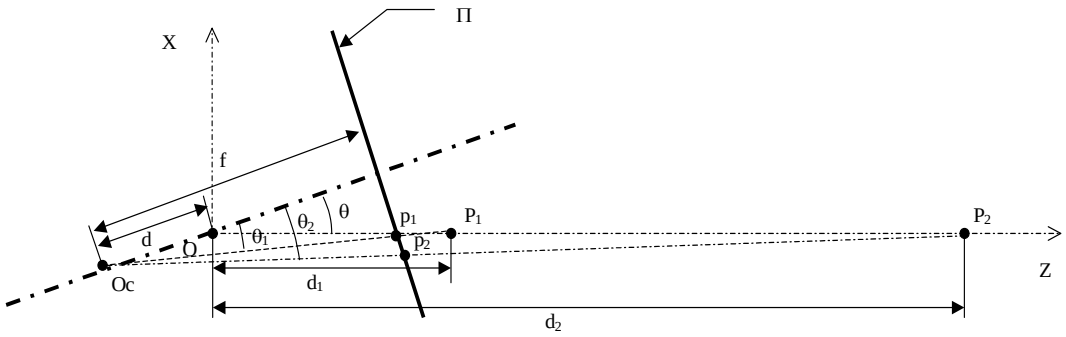


Figure 60 : Décalage négatif du centre optique de la caméra par rapport au centre des rotations

La différence est que si le centre de projection est entre le centre des rotations et le plan image, un point proche de la camera aura tendance à être décalé vers l'extérieur de l'image. A l'inverse, si le centre de projection est placé derrière le centre des rotations, un point proche de la camera aura tendance à être décalé vers le centre de l'image. Dans les deux cas, le décalage observé est un décalage relatif par rapport au cas où O et Oc sont confondus.

A partir des équations présentées au chapitre précédent, il est possible de déterminer l'angle θ_1 d'un point p_1 de coordonnée (u_1,v_1) dans le plan image. Pour rappel :

$$\tan\theta_1 = \frac{u_1}{f}$$

Nous cherchons à présent à déterminer la distance d correspondant à la distance entre le centre O des rotations et le centre Oc de projection. En fonction des données du problème, nous pouvons reconstituer le schéma simplifié suivant :

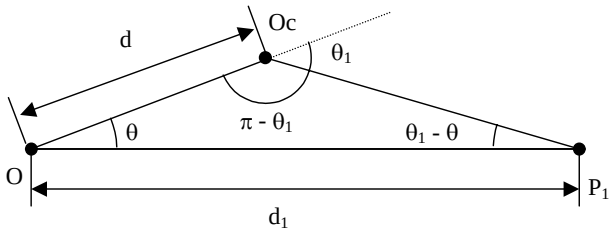


Figure 61 : Schéma simplifié du décalage positif entre le centre optique et le centre des rotations

De ce schéma, nous pouvons écrire :

$$d \cdot \sin\theta_1 = d_1 \cdot \sin(\theta_1 - \theta)$$

d'où :

$$d = d_1 \cdot \frac{\sin(\theta_1 - \theta)}{\sin\theta_1} \qquad (3.1)$$

3.3.2. Expérimentations

Afin de vérifier si le centre optique est bien confondu avec le centre de rotation de la caméra, nous avons réalisé une petite expérience à partir de deux fils à plomb suspendus au plafond. Nous avons placé la caméra de façon à ce que le plan formé par les vecteurs OY et OZ soit confondu avec les deux droites symbolisées par les fils à plomb. La position de la caméra est réglée de façon à ce que le fil1 apparaisse verticalement et passe au centre de l'écran. Dans cette position le fil2 n'est pas visible à l'écran puisqu'il est caché par le fil 1.

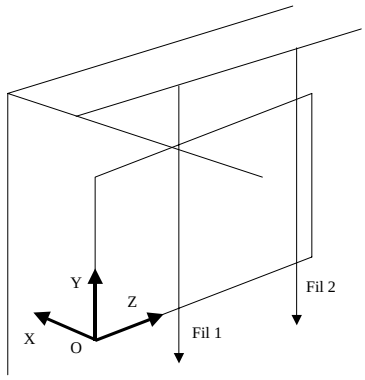


Figure 62 : Schéma de principe de l'expérience des fils à plomb

Si l'axe de rotation de panorama est confondu lui aussi avec l'axe Y alors les deux fils et l'axe Y sont toujours dans le même plan quelle que soit la valeur de l'angle de panorama. En revanche, si les deux axes ne sont pas confondus alors les deux fils et l'axe Y ne sont pas dans le même plan. Dans le premier cas, le fil 2 ne sera jamais visible, alors que dans le deuxième cas, le fil 2 apparaîtra à gauche ou à droite du fil1 pour une rotation panoramique.

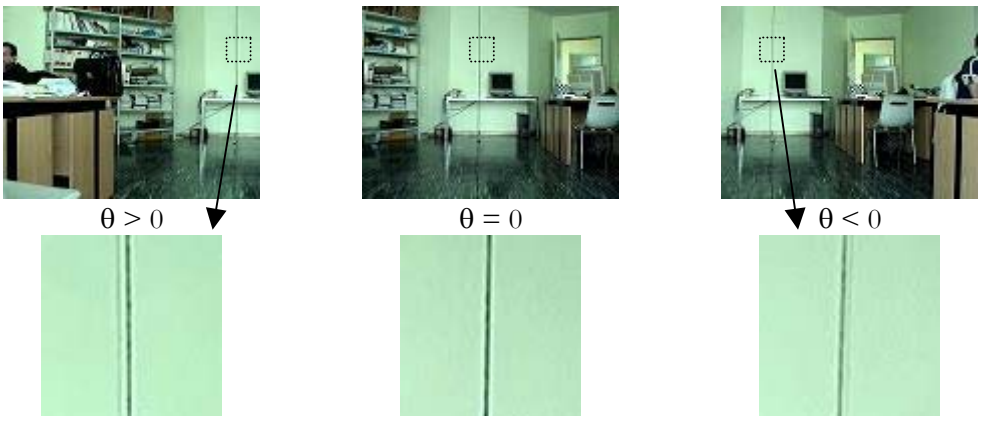


Figure 63 : 3 images prises avec des angles panoramique différent et pour un angle de tangage $\varphi = 0$ avec au-dessous le grossissement au niveaux des fils

La figure ci-dessous montre les résultats de l'expérience pour trois angles panoramiques différents et pour un angle de tangage $\varphi = 0$. Le fil épais sur les images correspond au fil 1 qui est le fil le plus près de la caméra. Le fil fin correspond au fil2, le plus éloigné. On remarque que sur l'image du milieu, correspondant à un angle $\theta = 0$, les deux fils sont

confondus. Dans l'image de gauche pour un angle $\theta > 0$ et pour l'image de droite correspondant à $\theta < 0$, les deux fils ne sont pas confondus. En conclusion, l'axe de rotation panoramique n'est donc pas confondu avec l'axe vertical passant par le centre optique de la caméra. Nous avons réalisé une série de mesures avec notre caméra afin de déterminer la valeur de ce décalage.

Distance focale (mm)	d (mm)
4.1	21 ± 2
8	0 ± 2
20	-56 ± 4
30	-105 ± 10
40	-154 ± 20

Tableau 4 : Mesure du décalage entre le centre des rotations et le centre optique pour plusieurs distances focales. Mesures réalisées sur une caméra Sony RZ25P

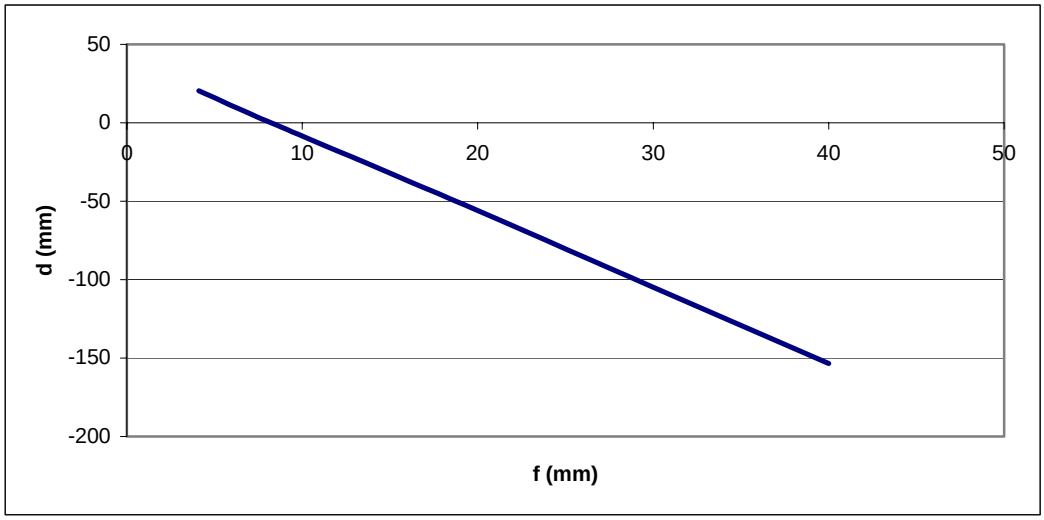


Figure 64 : Représentation graphique du décalage entre le centre des rotations et le centre optique pour plusieurs distance focale. Mesures réalisées sur une caméra Sony RZ25P

N'ayant pas à notre disposition d'outils de mesure et d'alignement précis, nos mesures manquent de précision. Cependant, nous pouvons en tirer quelques enseignements intéressants. Le plus important est que pour une distance focale de 8 mm, nous n'avons pratiquement pas de décalage observable. Avec notre caméra, cette distance sera donc privilégiée pour réaliser des panoramas robustes. Les autres enseignements sont que pour une distance focale inférieure à 8mm, le centre de projection se situe entre le centre de rotation et le plan focal alors que pour une distance focale supérieure, le centre de projection se situe à l'arrière du centre des rotations.

3.3.3. Influence du décalage

Ce décalage que nous avons observé va introduire une distorsion supplémentaire de l'image. Nous allons donc étudier l'influence de ce décalage sur la projection d'un point sur l'image.

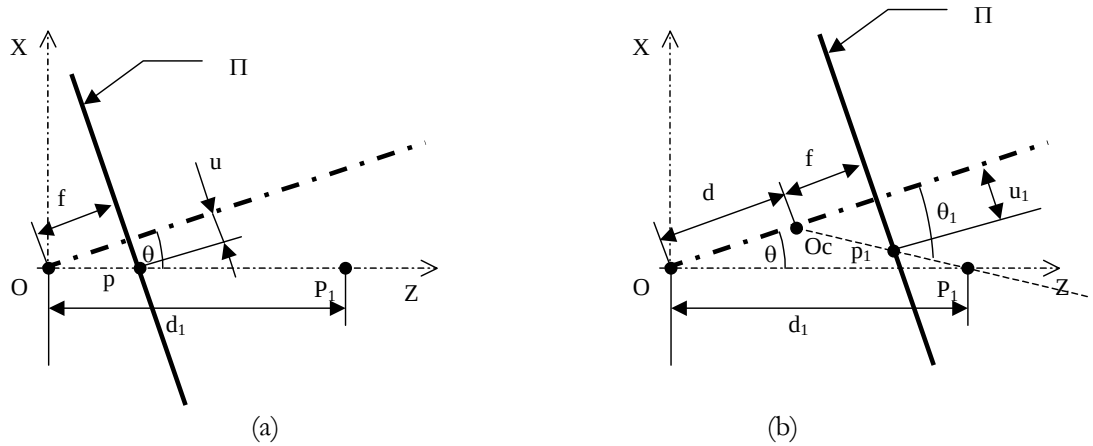


Figure 65 : Influence du décalage entre le centre optique et le centre des rotations sur un point de l'image.

Le schéma de gauche (Figure 65a) correspond au cas idéal où le centre de projection est confondu avec le centre des rotations. La coordonnée u de la projection du point P1 sur le plan image est donnée par :

$$u = f \cdot \tan\theta$$

Le schéma de droite (Figure 65b) correspond au cas réel où le centre de projection est à une distance d du centre des rotations. A partir des équations précédentes, nous pouvons exprimer l'expression de la coordonnée u_1 de la projection du point P1 sur le plan image :

$$u_1 = f \cdot \tan\left[\theta + \tan^{-1}\left(\frac{d \cdot \sin\theta}{d_1 - d \cdot \cos\theta}\right)\right] \qquad (3.2)$$

Le décalage $\Delta u = u_1 - u$ entre ces deux expressions correspond à l'erreur de projection liée au décalage du centre de projection et du centre des rotations. A partir de l'expression ci-dessous, nous pouvons déjà remarquer que si d_1 tend vers l'infini, l'erreur tend vers 0. L'erreur de projection diminue lorsque le point s'éloigne de la caméra. Pour être significatif, l'erreur de projection doit être supérieure à 1 pixel.

Le graphique suivant est une étude purement théorique. Elle illustre, pour différentes distances focales, l'erreur de positionnement en fonction l'éloignement d'un point pour un décalage d = 20mm et un angle de panorama θ =2°.

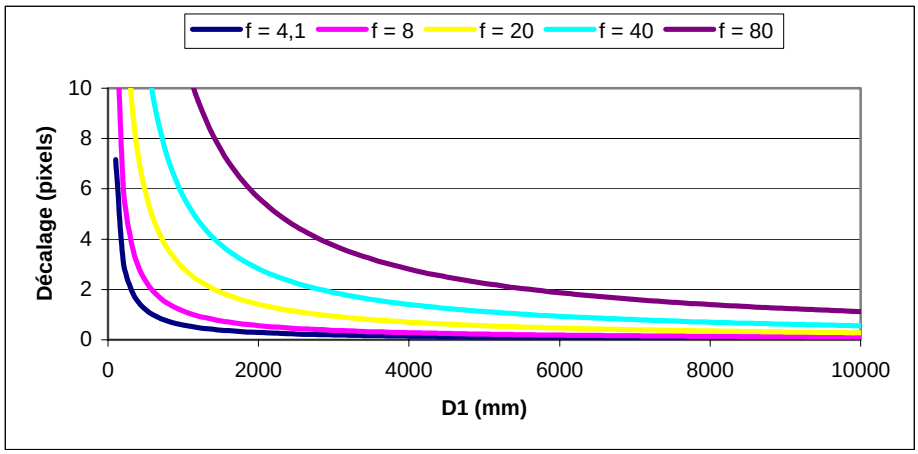


Figure 66 : Erreur de projection en fonction de l'éloignement d'un point pour différente distance focale et avec d = 20mm et θ =2°

Par l'intermédiaire de ce graphique nous cherchons à déterminer l'influence du décalage en fonction de la distance focale. Comme nous pouvions nous y attendre, pour un même décalage fixé arbitrairement à 20 mm, les effets sont plus marqués lorsque la distance focale est importante. Autre enseignement, l'effet du décalage s'estompe lorsque la distance entre l'objet observé et la caméra augmente.

Le graphique suivant est plus représentatif de notre caméra. Il représente également l'erreur de projection en fonction de l'éloignement d'un point. Par contre, pour chaque distance focale, nous avons appliqué le décalage entre les centres de projection et de rotation en fonction des relevés que nous avons effectués (Tableau 4). De plus, comme le décalage est aussi fonction de l'angle de rotation, pour chaque distance focale, nous avons appliqué un angle de rotation maximum. Cet angle est déterminé de telle sorte qu'un objet placé au centre de l'image avant rotation soit toujours visible dans l'image après rotation et qu'il se situe sur le bord droit ou gauche de l'image.

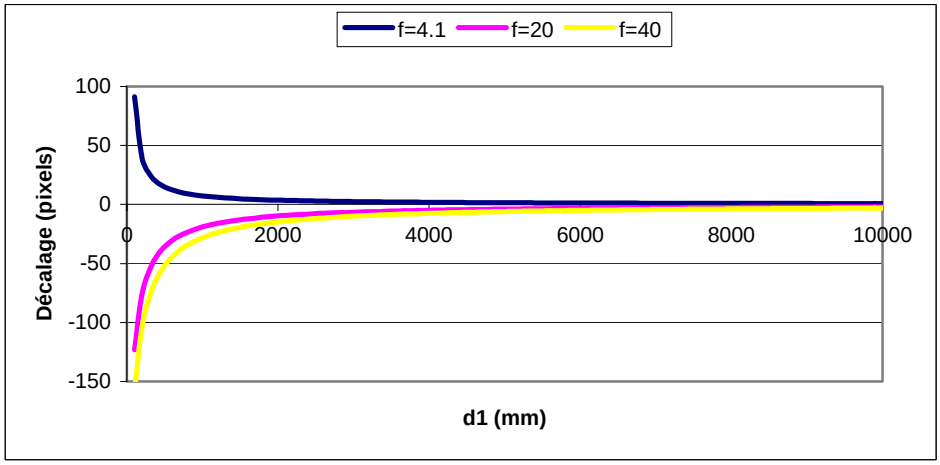


Figure 67 : Erreur de projection en fonction de l'éloignement d'un point sur une caméra Sony RZ25P

Le tableau suivant est une synthèse des différentes mesures que nous avons effectuées et des limites pour obtenir un panorama robuste. Dans ce tableau, d_1 min correspond à la distance minimum entre la caméra et un objet dans la scène de telle sorte que le décalage maximum observé pour cet objet, après rotation de la caméra, soit inférieur à 1 pixel dans les différentes images. Ces différentes mesures ont été relevées sur notre caméra SONY RZ25P avec une résolution d'image de 640 x 480 pixels.

f (mm)	d (mm)	θ max. (°)	d_1 min. (m)
4.1	21 ± 2	21.3	7.2
8	0 ± 2	11.3	0
20	-64 ± 4	4.5	20
30	-96 ± 10	3.1	31
40	-172 ± 50	2.3	55

Tableau 5 : Synthèse du défaut d'alignement de notre caméra Sony RZ25P

3.3.4. Conclusion

En conclusion, pour réaliser un panorama robuste, nous devons prendre un certain nombre de précautions. Il n'y a pas de solution simple pour corriger le défaut d'alignement. Si le centre de projection se déplace le long de l'axe optique en fonction du facteur de zoom, le mouvement de rotation de la caméra entraîne un décalage des pixels vers la droite ou vers la gauche en fonction de l'angle et de la profondeur (dans le repère du monde) du point visé. Pour Peer [PEE02] ce défaut est une qualité. Il utilise justement cette excentricité pour calculer la carte de profondeur d'un panorama. Pour réaliser un panorama robuste, nous devons nous mettre dans des conditions qui minimise la portée de cette excentricité. Dans la mesure du possible, afin d'éviter les distorsions dues au décalage entre le centre de projection et le centre des rotations, nous utiliserons une distance focale de 8mm. Un autre intérêt est que pour cette distance focale, les distorsions radiales sont également négligeables. Pour les distances focales inférieures, si les obstacles sont à une distance supérieure à 7m (pour la focale la plus petite) et que la résolution de l'image est de 640 x 480 pixels, les distorsions dues au décalage ne sont pas significatives. Par contre, nous devrons tenir compte des distorsions radiales. Pour des distances focales supérieures à 8 mm, la distance minimale devient rapidement importante. Elle est de 55m pour une distance focale de 40mm. C'est une contrainte importante lorsque l'on souhaite réaliser un panorama dans un espace relativement réduit. Par contre cela ne constitue plus un problème lorsque l'on réalise le panorama d'un paysage.

3.4. Défaut de positionnement

Le défaut d'alignement que nous avons abordé au paragraphe précédent peut être considère comme un défaut intrinsèque à la caméra puisqu'il résulte de la conception même de la caméra. Un autre défaut que nous pouvons qualifier également de défaut intrinsèque est le défaut de positionnement. En effet, si la commande de la caméra n'est pas suffisamment précise, le plaquage des images sur le panorama ne sera pas correct. En prenant la distance focale la plus faible (i.e. la moins sensible au défaut de positionnement), une erreur de 0,5° sur la position de l'angle panoramique engendre un décalage de près de 7 pixels sur l'axe horizontal de l'image. Pour réaliser un panorama robuste, nous avons donc à résoudre un problème de recalage d'image avec comme première conséquence, la nécessité de garantir un certain recouvrement entre les images et donc d'augmenter le nombre de prises de vue. Le

recalage d'image est un problème complexe qui motive énormément la communauté scientifique. Nous l'évoquons simplement ici et nous lui consacrons le chapitre suivant.

3.5. Alignement photométrique

3.5.1. Problématique

Le contrôle de la luminosité n'est pas un problème aussi simple qu'il y paraît. Si la luminosité est relativement constante dans toutes les directions alors il suffit de fixer manuellement le gain de la caméra pendant toute la durée de la prise de vue. Par contre, il ne faut pas que cette luminosité évolue pendant cette prise de vue (lors du passage d'un nuage par exemple). Si il y a de trop gros écarts de luminosité dans la scène alors des zones risquent de se trouver sous-exposées. Elles apparaîtront donc très sombres dans la mosaïque d'image. A l'inverse, d'autres zones beaucoup plus lumineuses risquent de se trouver sur-exposées. La solution est donc de laisser la caméra gérer automatiquement le gain en fonction de la luminosité de la portion de scène visée. Cette solution offre bien sûr l'avantage que chaque image est acquise en optimisant la dynamique du capteur. Le problème est qu'une même portion de la scène prise avec deux gains différents n'a plus la même valeur de luminosité. L'exemple suivant est un cas d'école. La caméra est placée à l'intérieur d'une pièce. Cette pièce est éclairée par une lumière artificielle et par une lumière naturelle à travers une fenêtre. La luminosité à l'intérieur est faible par rapport à celle de l'extérieur. Si le gain de la caméra est fixé manuellement pour obtenir une luminosité correcte à l'intérieur de la pièce, la fenêtre apparaîtra sur-exposée et il ne sera pas possible de visualiser l'extérieur. Dans le cas contraire, si le gain est piloté automatiquement par la caméra, alors l'extérieur devient visible, mais le phénomène de couture apparaît.

Figure 68 : Gain automatique, l'extérieur est visible mais présence de couture

3.5.2. Etat de l'art

Pour résoudre cet alignement photométrique et éviter ce problème de couture, plusieurs solutions sont proposées dans la littérature. Nous pouvons les classer en 2 catégories en fonction de la portée de la correction dans le panorama. Une première classe d'algorithme proposée dans la littérature s'attache à appliquer une correction du gain sur l'ensemble du panorama. Dans ce cas, deux solutions se détachent. La première consiste à normaliser le gain des différentes images constituant le panorama. Huang et al [HUA98] proposent une normalisation du gain de la camera à partir de la moyenne et de l'écart type des pixels présents dans la zone de recouvrement. La correction du gain est appliquée à l'ensemble de

l'image. Leur approche a finalement pour effet de fixer le gain de toute la construction panoramique à la valeur d'une image de référence. On se retrouve donc dans le cas du gain fixe à la différence qu'il est calculé automatiquement et non fixé manuellement par l'utilisateur. Dans [CAN03], Candocia utilise une technique similaire mais appliquée localement de façon à améliorer la mise en correspondance des images.

Figure 69 : Normalisation du gain à partir de la valeur moyenne et de l'écart type

Dans cette première approche, il y a une perte d'information lié à la contraction de la dynamique des pixels. Pour éviter ces désagréments, plusieurs auteurs dont Mann [MAN95], proposent d'étendre la dynamique des pixels. Les composantes RGB des pixels ne sont plus codées sur 8 bits mais sur 16 ou 32 bits. Cette solution offre l'avantage de tirer profit de la dynamique totale du capteur. Les zones sous exposées et sur exposées sont acquises en optimisant la définition. Le problème se pose lors de la restitution du panorama. En fonction de l'utilisation demandée, une contraction ou un glissement de la dynamique du panorama doit être opéré.

La deuxième catégorie d'algorithme a une portée plus locale et s'attache à améliorer la transition entre deux images. Une première solution consiste à réaliser une simple moyenne entre les pixels des deux images. Si l'écart de gain entre les deux images est faible, cette solution donne des résultats acceptables mais dès que l'écart est important, la présence de couture réapparaît.

Figure 70 : Correction du gain par calcul de la moyenne entre deux pixels commun

Uyttendaele and al [UYT02] proposent une solution pour éliminer ce qu'ils appellent les artefacts de luminosité. Ils découpent chaque image en blocs de 32x32 et calculent pour chaque bloc une fonction de transfert permettant de lisser la luminosité.

Dans [LEV04], les auteurs proposent une solution localisée uniquement sur la jonction entre deux images. Ils lissent la couture de façon à limiter le gradient entre les deux images. Le reste de l'image comme la zone de recouvrement, ne sont pas modifié. Utilisée seule, cette solution peut s'avérer très utile lorsque la zone de recouvrement entre les images est très faible. C'est notamment le cas si nous utilisons notre algorithme d'optimisation du pavage de la sphère. En revanche, le résultat est visuellement moins bon lorsque les recouvrements sont plus importants, notamment au regard des autres méthodes. Par contre, utilisée avec un autre algorithme de lissage, cette méthode permet d'atténuer efficacement les coutures résiduelles.

Une autre solution proposée par [GRA09, LEM07] consiste à utiliser un algorithme de type Graph-Cut. Le principe de cet algorithme est de trouver la jointure qui donne la meilleure délimitation dans la zone de recouvrement.

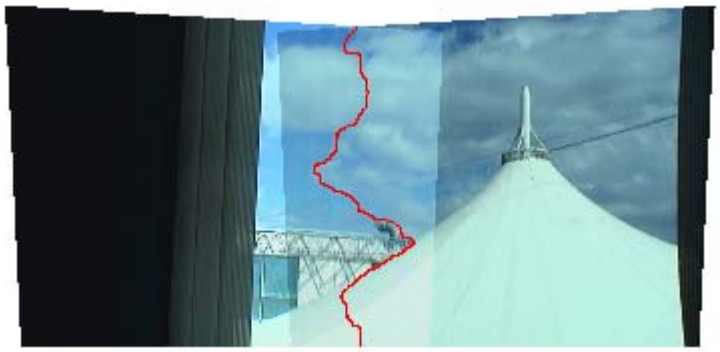


Figure 71 : Exemple de délimitation avec l'algorithme « Graph Cut »

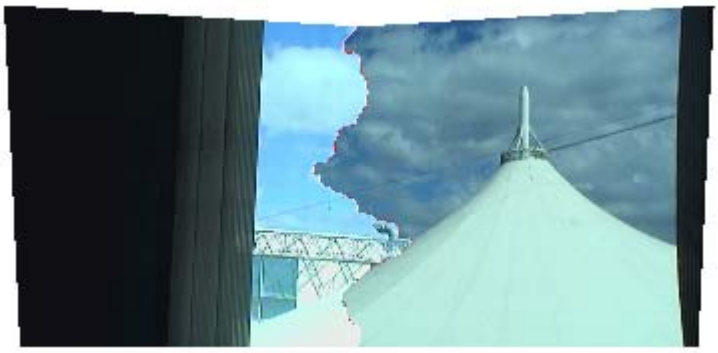


Figure 72 : Résultat du plaquage des deux images avec l'algorithme « Graph Cut »

Comme pour la méthode de diffusion par la moyenne, cette solution donne d'assez bons résultats lorsque la différence de gain est faible. Comme nous pouvons le voir sur l'image ci-dessus, le résultat est visuellement moins bon avec un écart de gain important.

3.5.3. Solution proposée

La solution que nous proposons permet de résoudre ce problème de couture en temps réel. Le résultat obtenu est visuellement un peu moins bon que les solutions proposées dans les logiciels de construction d'un panorama. Par contre ces solutions ne s'exécutent pas en temps réel. La méthode temps réel que nous proposons consiste à lisser par un dégradé les zones des images qui se recoupent. Traditionnellement la zone de recouvrement entre deux images est calculée en effectuant la moyenne des pixels issus des deux images. Dans l'exemple ci-dessous, nous avons plaqué deux images l'une rouge et l'autre jaune sur une face de cube avec une zone de recouvrement. La méthode simple basée sur la moyenne fait apparaître cette zone de recouvrement en orange avec des coupures franches.

(a) *(b)*

Figure 73 : Exemple de rendu du calcul de la zone de recouvrement entre deux images
(a) : calcul de la valeur moyenne dans la zone de recouvrement
(b) : pondération proportionnelle à la distance du point au contour

Nous avons introduit un facteur de pondération proportionnel à la distance du pixel par rapport à un point du contour le plus proche. Concrètement, nous commençons par déterminer la zone de recouvrement entre les deux images. Cette zone est délimitée par un contour. Pour chaque point de ce contour nous recherchons l'image à laquelle il appartient. En toute objectivité, il appartient aux deux, mais nous considérons qu'il appartient à l'image I si les 8 pixels qui l'entourent appartiennent aussi à l'image I. Puis, pour chaque pixel p de la zone de recouvrement, nous calculons la distance minimale d^1 qui sépare ce point d'un point du contour appartenant à l'image 1 et la distance d^2 le séparant d'un point du contour de l'image 2. La valeur d'un pixel p_k dans la zone de recouvrement est donnée par :

$$p_k = p_k^1 \cdot \frac{d_k^2}{d_k^1 + d_k^2} + p_k^2 \cdot \frac{d_k^1}{d_k^1 + d_k^2} \qquad (3.3)$$

Le résultat final est le suivant :

Figure 74 : Correction du gain par diffusion

La correction que nous apportons n'est pas parfaite, mais elle permet de lisser le changement de luminosité et ne laisse pas apparaître de discontinuité. L'image suivante correspond à un panorama complet d'une scène d'extérieur.

Figure 75 : Correction du gain par diffusion sur une scène en extérieur

Pour construire ce planisphère, nous avons appliqué notre algorithme sur chaque image acquise. Le défaut du à la correction est particulièrement visible sur le ciel. Paradoxalement, le ciel est représenté plus sombre là où se situe le soleil. Les nuages apparaissent plus sombres qu'ils ne sont en réalité. Ceci est du au fait que le gain de la caméra a été particulièrement diminué sur cette zone de l'image particulièrement lumineuse.

3.6. Suppression des « fantômes »

3.6.1. Problématique

Le dernier problème que l'on rencontre lors de la construction d'une mosaïque d'images est que pendant le processus d'acquisition, les objets contenus dans la scène ne sont pas forcément immobiles. Entre deux prises de vue, il arrive qu'un objet (ou plusieurs objets), situé(s) dans la zone de recouvrement de deux images, soi(en)t en mouvement. Lors de la projection des images dans le panorama, les objets en mouvement ne pourront pas être alignés correctement. Si nous appliquons les différents algorithmes d'alignement photométrique que nous avons présentés précédemment, les objets en mouvement vont apparaître flous ou plus ou moins fondus dans la scène, d'où l'apparition de « fantômes ».

Figure 76 : Exemple de séquence d'images avec un personnage en mouvement dans la scène

Figure 77 : Projection des images dans le même plan en utilisant la méthode de la moyenne.

Dans l'exemple ci-dessous, les trois images ont été acquises à deux secondes d'intervalle. Le décalage entre deux images successives est d'environ 2° selon l'angle de panorama. Dans cette scène un personnage est en mouvement. Après projection des trois images dans le plan de l'image t+1, où une simple moyenne des pixels est calculée dans la zone de recouvrement, les composantes statiques de la scène sont correctement alignées et apparaissent nettes. Par contre, les pixels correspondant au personnage en mouvement sont moyennés avec des pixels du fond de la scène. Le personnage semble donc se « dématérialiser ».

3.6.2. Cas général

Les solutions proposées dans la littérature afin de supprimer ces « fantômes » dépendent du nombre d'images mises à contribution pour reconstruire une portion de la scène. A partir de 3 images, la plupart des auteurs dont [IRA96] proposent l'utilisation d'un filtre médian. C'est une méthode simple à mettre en œuvre et qui donne des résultats relativement satisfaisants, comme nous pouvons le voir dans l'image suivante.

Figure 78 : projection des images dans le même plan en utilisant un filtre médian

Pour calculer cette image, nous avons repris les images de l'exemple précédent. Comme nous pouvons le constater, le personnage n'apparaît plus. Seules les composantes statiques sont conservées.

Dans [SHU00], les auteurs proposent une méthode basée sur le calcul du flot optique. Dans [UYT02], les auteurs identifient et isolent les régions en mouvement dans chacune des images. Chacune de ces régions correspond à une partie plus ou moins importante de l'objet en mouvement. A partir d'un graphe de mise en correspondance de ces régions, ils recherchent la région qui contribue le plus à la représentation de l'objet. Seule cette région est maintenue dans l'image correspondante et les autres sont supprimées.

Figure 79 : projection des images dans le même plan en utilisant la méthode décrite dans [UYT02]

Lorsque le panorama est réalisé à partir d'un flux vidéo, nous pouvons disposer d'un nombre beaucoup plus important d'images pour restituer une portion de la scène. Nous pouvons alors, après projection de chaque image dans un plan de référence, utiliser des méthodes de modélisation des composantes statiques du fond. Ces méthodes sont traditionnellement utilisées pour segmenter les objets en mouvement dans le cas des caméras fixes. Dans le cas d'une caméra PTZ en rotation, nous pouvons considérer qu'après projection de chaque image dans un plan de référence l'image correspondant à ce plan est une image issue d'une caméra fixe. De même, chaque prise de vue peut être réalisée à partir d'une collection

d'image pendant laquelle la caméra est fixe. Nous allons donc nous intéresser aux différents algorithmes permettant cette modélisation.

3.6.3. Modélisation des composantes statiques du fond

Plusieurs stratégies différentes sont proposées dans la littérature pour créer et mettre à jour un modèle de fond. La première approche consiste à réaliser une moyenne glissante à chaque nouvelle acquisition d'image. Cette solution simpliste peut donner de bons résultats lorsqu'il n'y a pas de changement de luminosité et que la scène ne subit pas de modification. Suivant le temps d'intégration de l'image de référence et la vitesse de déplacement des objets, cette technique permet d'extraire la forme complète contrairement à l'approche précédente où seuls les pixels ayant changés de luminosité entre deux images sont détectés. Il n'en reste pas moins que cette approche présente, comme la précédente, l'inconvénient d'avoir à déterminer, souvent empiriquement, le seuil de la différence à fixer pour étiqueter les pixels en mouvement. Ce seuil, qui est généralement le même pour chacun des pixels de l'image, est sensible aux contrastes de l'image. Il s'agit alors de trouver un compromis entre les zones ensoleillées et les zones de l'image à l'ombre, à moins d'analyser localement le contraste des différentes zones de l'image de façon à adapter le seuil.
Cette solution est bien trop restrictive pour s'appliquer à l'extérieur. Bertolino et al. [BER01] ont développé une méthode basée sur le calcul d'une moyenne glissante en considérant deux phases : initialisation et mise à jour. Dans la phase d'initialisation, l'image de référence est construite à partir du calcul d'une moyenne glissante où chaque pixel est pondéré par un terme α_p sur n images successives d'une séquence.

$$Iref(p,t+1)=\alpha_p \cdot I(p,t)+(1-\alpha_p)\cdot Iref(p,t) \tag{3.4}$$

La valeur du terme α fixe le temps de réponse du filtre. Pour ne prendre en compte dans l'initialisation de cette image de référence que les pixels fixes, ils créent une carte de stabilité en calculant la différence de trois images successives. Cette carte leur permet donc de distinguer les pixels fixes des pixels mobiles. Pour chaque pixel p, si p appartient au fond, alors $\alpha_p \in]0,1]$ sinon $\alpha_p=0$. Une fois l'image de référence construite, ils passent dans une deuxième phase où l'image de référence est mise à jour en utilisant le même principe de la moyenne glissante pondérée par α_p sans utiliser de carte de stabilité. Les changements brutaux de la luminosité globale de l'image sont détectés en calculant simplement la différence cumulée de chaque pixel entre deux images successives divisée par le nombre de pixels dans l'image. Si cette valeur atteint un certain seuil, cette valeur est ajoutée à tous les pixels de l'image de référence.
Dans un article très complet sur la surveillance en temps réel des personnes, Haritaoglu et al [HAR00] proposent également une solution permettant de modéliser le comportement de chaque pixel de l'image dans une fenêtre glissante. Après l'application d'un filtre médian sur l'image courante, ils déterminent pour chaque pixel, la valeur minimum d'illumination, la valeur maximum ainsi que le gradient maximum entre deux images successives. Ces trois informations pour chaque pixel leur permettent de déterminer les seuils pour l'étiquetage des pixels de l'image suivante. L'inconvénient majeur de cette approche, comme pour l'algorithme de Perner, est qu'il est nécessaire de garder en mémoire un buffer de plusieurs images.
Afin de palier ce défaut, plusieurs auteurs [STA99,PAV01,CHU04], modélisent la variation de chaque pixel de l'image au cours du temps par plusieurs distributions gaussiennes représentées par une moyenne et un écart type. Cette méthode est communément appelée « mélange de gaussienne ». Le nombre de distributions utilisées pour la modélisation, dépend

de la complexité des mouvements du fond. Si le fond reste fixe, deux distributions suffisent. Si le fond varie, ce qui est notamment le cas dans les scènes en extérieur avec du vent dans les arbres par exemple, il peut être nécessaire d'utiliser plusieurs distributions. Dans [DAR05], Dar-Shyang propose une solution basée sur l'algorithme de Stauffer et al. [STA99] permettant d'améliorer la stabilité et la rapidité de la convergence. Nous reviendrons un peu plus en détail sur cette approche dans le chapitre suivant.
Enfin, dans [KIM04] les auteurs proposent une méthode utilisant ce qu'ils appellent des « codebook ». Le principe consiste à isoler la variation de chaque pixel dans une ou plusieurs zones colorimétrique en mémorisant un certain nombre de caractéristiques comme les dates de début et de fin d'apparition du pixel dans la zone, le nombre d'occurrence, etc. Les auteurs proposent des stratégies pour fusionner ou supprimer des blocs. Cette méthode semble donner de bons résultats. Cependant, le temps d'intégration pour stabiliser le modèle est relativement long.

Figure 80 : Extrait du panorama réalisé à partir d'une séquence vidéo en utilisant les mélanges de gaussienne.

3.7. Conclusion

Notre première préoccupation était de déterminer l'influence de l'écart entre notre caméra et les modèles mathématiques que nous utilisons. Au regard des résultats, nous pouvons considérer que les modèles utilisés sont valables dans des conditions normales d'utilisation. Nous avons également présenté les problèmes liés à l'alignement photométrique et à la suppression des « fantômes ». Plusieurs solutions existent mais elles sont utilisées essentiellement lorsque la finalité de la construction du panorama est la visualisation de la scène. Lorsque la caméra est utilisée pour de la détection en temps réel ou pour piloter un robot, ces traitements ne sont pas appliqués. Le défaut de positionnement que nous allons aborder maintenant est plus significatif. C'est la raison pour laquelle nous lui consacrons le chapitre suivant.

Mosaïque d’images multi-résolution et Applications

Chapitre 4 : Recalage d’images appliqué aux caméras PTZ

4. Recalage d'images appliqué aux caméras PTZ

4.1. Problématique

Au cours du chapitre précédent nous avons abordé un certain nombre de problèmes et apporté des solutions permettant d'obtenir un panorama robuste aux conditions de prise de vue et aux objets en mouvement dans la scène. Ces différents points sont essentiellement gênants pour la visualisation *a posteriori* de la scène. Dans les applications de détection ou de pilotage d'un robot, ces défauts ne sont pas traités de la même façon dans la mesure où ce qui importe ce n'est pas l'ensemble de la scène mais principalement ce qui est vu à l'instant présent par la caméra. Pour la visualisation d'un panorama, nous cherchons à supprimer les objets en mouvement alors que pour la détection nous cherchons justement à les détecter. Cependant, dans les deux cas, il y a un défaut qu'il est absolument impératif de corriger. C'est le défaut de positionnement. Pour que la visualisation de la scène soit parfaite, il est nécessaire que les images acquises soient correctement plaquées sur le panorama. De même pour le suivi automatique des objets en mouvement et le pilotage d'un robot, il est indispensable de connaître les paramètres de prise de vue de façon à extraire de l'image les bonnes informations de pilotage. La solution la plus simple est d'instrumenter correctement la caméra. Si ce n'est pas le cas, nous devons donc extraire les paramètres de prises de vue à partir de l'image elle-même et des images acquises précédemment. Nous avons donc un problème de recalage d'images à résoudre.

Le recalage d'images est un problème particulièrement intéressant. Initialement, il ne devait pas faire l'objet de nos travaux de recherche. Cependant, au vu des problèmes rencontrés, notamment en ce qui concerne le temps de calcul, nous avons dû y consacrer une partie de notre travail de thèse. Bien que de nombreuses solutions aient été proposées pour la construction de panoramas, la réalisation de mosaïques de haute qualité en temps réel reste une tâche très difficile. Le recalage d'images n'est pas un problème spécifique au mosaïquage. Dans certaines applications, le recalage d'images est l'objectif final, alors que dans d'autres applications, c'est un lien exigé pour accomplir des tâches d'un niveau plus élevé. Le but du recalage d'images est d'aligner géométriquement deux images ou plus de sorte que des pixels respectifs ou leurs dérivés (bords, coin, etc..), représentant la même structure fondamentale, puissent être mis en correspondance. L'objectif de la plupart des algorithmes décrits dans la littérature est de mettre en correspondance les images selon leurs propriétés radiométriques ou géométriques en utilisant une fonction spécifique pour évaluer la qualité de la mise en correspondance. Quelles que soient les méthodes utilisées et comme précisé dans [BRO92], le recalage entre deux images I et J peut se formaliser de la façon suivante :

$$\hat{T} = \arg\min_{T \in E} C(I, J \circ T) \tag{4.1}$$

où T est la transformation qui appartient à un espace de recherche E de transformations, $J \circ T$ est l'application de la transformation T sur l'image J et C est une mesure de similarité que l'on va chercher à minimiser.

Pour rappel (cf. Chapitre 2), dans le cas de notre étude, T est la transformation homographique H qui relie deux images. Dans la suite de cette étude, nous continuerons à utiliser la lettre T pour exprimer la transformation de façon à généraliser notre propos, tout en gardant à l'esprit que la transformation que l'on recherche est une homographie 2D qui s'exprime avec 8 coefficients et qui est définie à un facteur d'échelle près :

$$x' \sim H \cdot x = \begin{bmatrix} m_0 & m_1 & m_2 \\ m_3 & m_4 & m_5 \\ m_6 & m_7 & 1 \end{bmatrix} \bullet \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

où x est un point de l'image I_1 défini par ses coordonnées homogènes (u, v, 1) dans l'image et x' sa projection homographique dans l'image I_0 et défini par ses coordonnées homogènes (u', v', 1). Le signe ~ indique la relation d'équivalence, à un facteur d'échelle près, entre le point x' et la transformation du point x.

Comme nous l'avons vu dans le chapitre 2, les paramètres de cette transformation peuvent être calculés directement à partir des informations des prises de vue. Cependant, nous pouvons également la calculer à partir des coordonnées des points dans les deux images. En effet, les coordonnées (u', v') du point x' s'expriment en fonction des coordonnées (u, v, 1) de la façon suivante :

$$u' = \frac{m_0 \cdot u + m_1 \cdot v + m_2}{m_6 \cdot u + m_7 \cdot v + 1}, \qquad v' = \frac{m_3 \cdot u + m_4 \cdot v + m_5}{m_6 \cdot u + m_7 \cdot v + 1}$$

Soit sous forme matricielle :

$$\begin{bmatrix} u & v & 1 & 0 & 0 & 0 & -u.u' & -v.u' \\ 0 & 0 & 0 & u & v & 1 & -u.v' & -v.v' \end{bmatrix} \cdot \begin{bmatrix} m_0 \\ m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_5 \\ m_6 \\ m_7 \end{bmatrix} = \begin{bmatrix} u' \\ v' \end{bmatrix} \tag{4.2}$$

L'appariement de n points entre deux images permet de disposer de 2n équations. Ce système peut alors se résoudre par la méthode des moindres carrés par exemple.

Comme nous l'avons évoqué, le recalage d'images n'est pas un problème spécifique à la construction d'un panorama. Nous n'allons pas présenter un état de l'art exhaustif des méthodes de recalage et nous limitons notre étude aux méthodes qui peuvent être appliquées aux caméras PTZ.

4.2. Etat de l'art du recalage appliqué aux caméras PTZ

Il y a plusieurs façons de classer les différentes approches. Elles peuvent tout d'abord être classées en fonction du type de transformation que nous allons faire subir à l'image ou de la complexité du modèle utilisé. Une autre méthode de classification consiste à considérer d'une part les approches locales qui visent à déterminer la transformation mathématique entre deux images seulement et les approches globales qui prennent en compte la totalité du panorama. Enfin, nous pouvons également les classer en fonction de l'utilisation d'une partie ou de la totalité des pixels des deux images.

La première classification que nous avons évoquée concerne le type de transformation que nous devons faire subir aux images. Dans ce cas, les transformations sont généralement classées en deux catégories : les transformations rigides et les transformations non rigides ou élastiques. Nous aurons une transformation rigide lorsqu'il n'y a pas de déformation de l'image et où nous n'utilisons que des rotations, des translations et des facteurs d'échelles. A l'inverse, s'il est nécessaire de faire subir une transformation inhomogène à l'image avec éventuellement changement de la topologie, nous aurons une transformation non rigide. Les transformations homographiques sont un cas particulier. Comme elles déforment l'image, nous avons une tendance naturelle à les classer dans les transformations non rigides. Cependant nous les classerons dans la catégorie des transformations rigides puisqu'en coordonnées homogènes la transformation est linéaire et bijective. Bhat et al [BHA00] utilisent un modèle basé uniquement sur un mouvement de translation de la caméra PTZ. Ce modèle simple est utilisé par les auteurs pour segmenter les objets en mouvement entre deux prises de vue. Ils considèrent donc que le mouvement de la caméra peut être approximé par une translation. Toutefois, cette hypothèse n'est vérifiée que pour les petits angles d'inclinaison. L'estimation de la translation est obtenue par la détection et le suivi de points d'intérêts, nous y reviendrons par la suite. Des modèles de transformations plus complexes sont généralement proposés, telles que les transformations rigides ou affines [SZE97, BRO03], ou les transformations projectives [BEV05, BEV06]. Toutefois, la plupart des caméras s'écartent du modèle sténopé, généralement en raison des distorsions radiales que l'on observe pour les distances focales faibles. Dans leur approche, Sinha et al. [SIN04] proposent une solution pour compenser ces distorsions. Leur modèle prend également en compte le rapport entre la hauteur et la largeur des pixels. En effet, comme nous l'avons évoqué dans le paragraphe de la définition de l'unité pixélique (2.2.2), les pixels ne sont généralement pas carrés mais rectangulaires. Cependant, cette information est un paramètre intrinsèque à la caméra qui n'évolue pas avec le temps. Il peut donc être déterminé à partir d'un calibrage de la caméra.

le deuxième mode de classification consiste à séparer les approches locales des approches globales. Les approches locales, qui sont les plus courantes, visent à déterminer les paramètres du modèle utilisé pour chaque couple d'images successives. Ces approches sont généralement efficaces et rapides, mais les petites erreurs d'alignement successives ont tendances à s'accumuler avec le temps. Ces erreurs sont d'autant plus visibles lorsque la vidéo renvoie à une zone du panorama déjà capturée (problème connu sous le nom de "looping path"). Les approches globales [SZE97, BRO03] formulent le problème du recalage d'images de façon à résoudre l'ensemble des paramètres et contraintes du système ; dans le cas des mosaïque d'images, une de ces contraintes est que les extrémités d'un panorama se rejoignent. Ces types d'approches d'optimisation exacte sont la plupart du temps incompatibles avec le temps réel.

De façon plus générale, les techniques de recalage sont classées en fonction de la contribution des pixels de l'image. Traditionnellement, deux types d'approches sont considérées : denses (dites aussi iconique) et éparses (géométriques). En simplifiant, nous pouvons dire que les méthodes denses mettent à contribution l'ensemble des pixels présents dans la zone de recouvrement des deux images alors que les méthodes éparses n'utilisent que certains pixels aux caractéristiques particulières. Les méthodes denses [BER92, IRA96, SZE97, SIN04] cherchent à estimer itérativement les paramètres de la caméra par la minimisation d'une fonction de coût basée la plupart du temps sur la différence d'intensité entre les zones de recouvrement. Nous verrons par la suite qu'il existe plusieurs méthodes de calcul pour estimer cette différence. La plus commune est la somme des différences au carré (Sun Square Difference SSD). Szeliski et Shum [SZE97] proposent une mise à jour itérative des paramètres de la matrice d'homographie basée sur l'utilisation de la mesure SSD. Les méthodes denses sont réputées plus précises mais ne sont pas robustes lorsqu'il y a beaucoup d'objets en mouvements dans les images. Elles sont donc particulièrement efficaces dans le cas de scènes statiques. Les méthodes éparses [BAR03, BEV05, BEV06, BRO03] utilisent la mise en correspondance de points d'intérêts, de lignes ou d'autres primitives géométriques pour déterminer les paramètres de la caméra. Plusieurs solutions proposées sont basées sur le détecteur de coins de Harris [HAR88] ou sur les SIFT décrit par Lowe [LOW04]. De façon plus marginale, d'autres solutions sont basées sur les appariements de régions [COH89, LEE93] ou sur des structures topologiques [FLE91]. Plus classiquement, Bevilacqua et. al [BEV05] proposent de mettre en correspondance des points caractéristiques de l'image. Ils utilisent un modèle de projection homographique et proposent une solution pour la fermeture du panorama. Ils améliorent leur approche dans [BEV06] en prenant en compte les changements d'illuminations. Brown et Lowe [BRO03] proposent de mettre en correspondance des points d'intérêts à partir de l'algorithme SIFT qui sera décrit par Lowe dans [LOW04]. Ils utilisent une transformation affine en justifiant que le descripteur SIFT est invariant aux changements affines, ainsi qu'un algorithme de type RANSAC. Cet algorithme probabiliste permet de minimiser la complexité de la mise en correspondance et permet de supprimer les correspondances aberrantes. Enfin, ils utilisent un ajustement décrit dans [TRI00] permettant un recalage global d'un ensemble d'images. Leur approche permet la génération du panorama particulièrement robuste. Cependant leur algorithme nécessite 83 secondes pour recaler 8 images avec un PC équipé d'un processeur 2GHZ.

Nous avons essentiellement axé notre travail de recherche sur l'optimisation du temps de calcul. Dans un premier temps, nous avons étudié deux méthodes décrites dans la littérature. Une méthode dense proposée par Szeliski [SZE94] et une méthode éparse, à partir des SIFT, décrit par Lowe [LOW04]. Ces deux méthodes se sont révélées particulièrement efficaces mais nécessitent des temps d'exécution beaucoup trop longs. Dans le cadre des applications visées, le recalage n'est qu'une étape intermédiaire, mais nécessaire, avant de réaliser des opérations de plus haut niveau. Nous avons donc étudié plusieurs approches différentes, denses ou éparses, que nous avons adaptées à notre cas particulier de façon à optimiser le temps de calcul. La suite de ce chapitre est structuré de la façon suivante. Nous allons tout d'abord définir notre espace de recherche puis nous allons étudier des mesures de similarités. Nous présentons ensuite succinctement les différentes méthodes de recalage que nous avons étudiées plus particulièrement. Nous ne présentons pas la vingtaine de méthodes que nous avons explorées mais seulement celles qui ont retenu notre attention et dont les résultats sont intéressants. Nous présentons ensuite notre solution puis le protocole de tests ainsi que les résultats que nous avons obtenus.

4.3. Limitation de l'espace de recherche

Pour aborder le problème du recalage, nous nous plaçons dans le cas d'une projection centrale pour laquelle le centre de projection est confondu avec le centre des rotations.

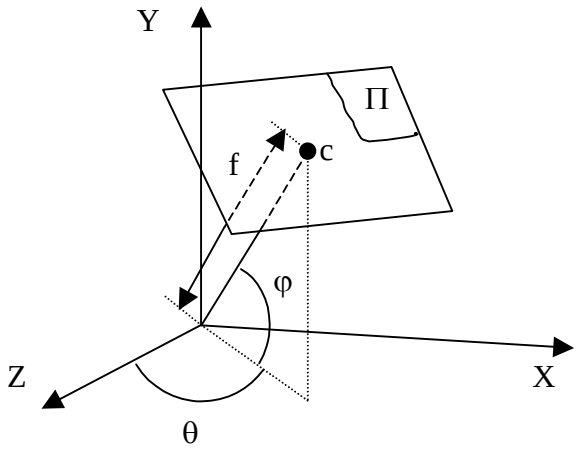


Figure 81 : Schéma de la projection centrale

Nous avons montré que dans ce cas, la transformation mathématique qui relie deux images est une homographie (i.e. Chap 2 : mosaïque d'image). Pour rappel, soit θ l'angle de panorama, φ l'angle de tangage et f la distance focale. c, représente le centre optique de l'image[6] et Π le plan image. Nous avons vu qu'une transformation homographique entre deux images s'exprime à partir d'une matrice H à 8 paramètres. Dans le cas d'une projection centrale on montre que cette matrice peut être calculée à partir des paramètres intrinsèque et extrinsèque du modèle sténopé [FAU93]. En utilisant un modèle simplifié et sans tenir compte des distorsions géométriques, chromatiques ou autre, la projection s'exprime de la façon suivante :

$$H = K_I \cdot R_{\varphi I} \cdot R_{\theta I} \cdot R_{\theta J}^{-1} \cdot R_{\varphi J}^{-1} \cdot K_J^{-1} \tag{4.3}$$

où K est la matrice simplifiée des paramètres intrinsèques du modèle, R_θ et R_φ les matrices de rotations suivant les angles de panorama et de tangage. La matrice K s'exprime de la façon suivante :

$$K = \begin{bmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.4}$$

où f_u et f_v correspondent à la distance focale exprimée en unité pixélique suivant l'axe u et l'axe v, et (u_0, v_0) correspond au centre de projection de l'image. Généralement, le modèle utilisé est encore plus simplifié. Dans ce cas, nous considérons que le repère $(c, \vec{u}, \vec{v})$ est orthonormé, c'est à dire que $f_u = f_v = f$. S'il est vrai que les axes u et v sont orthogonaux, les pixels ne sont pas forcément carrés mais plutôt rectangulaires. La distance focale exprimée en unité pixélique n'est donc pas forcément identique suivant les axes. Par exemple, nous

[6] Pour rappel, nous appelons centre optique de l'image, le point d'intersection entre l'axe optique et le plan image

avons mesuré un rapport fu/fv d'environ 1.02 sur la caméra sony RZ25P. Ce rapport est une donnée intrinsèque à la caméra et reste constant quel que soit le pilotage de la distance focale. De même nous considérons que le centre optique de l'image (u_0, v_0) reste fixe. Ce n'est pas tout à fait le cas sur notre caméra lorsque nous pilotons le facteur de zoom. Cependant, Szeliski a montré dans [SZE94] qu'un faible décalage du centre optique de l'image n'a que peu d'influence sur le recalage. Comme données variables au cours du temps, il reste à estimer les paramètres de prise de vue des deux images, c'est-à-dire la distance focale et les deux rotations. Soit 6 paramètres en tout. Cependant, la formulation précédente de la matrice H que nous avons volontairement décomposée laisse apparaître une simplification. En effet :

$$R_{\theta I} \cdot R_{\theta J}^{-1} = \begin{bmatrix} \cos(\theta_I - \theta_J) & 0 & -\sin(\theta_I - \theta_J) \\ 0 & 1 & 0 \\ \sin(\theta_I - \theta_J) & 0 & \cos(\theta_I - \theta_J) \end{bmatrix} \tag{4.5}$$

Ceci montre que les angles θ_I et θ_J n'interviennent pas directement dans le calcul de l'homographie, mais la différence entre les deux angles. En définitive, ce ne sont pas 6 paramètres qui sont nécessaires mais 5, à savoir : f_I, f_J, φ_I, φ_J et $\Delta\theta_{I,J}$. Notre espace de recherche est donc à cinq dimensions. Cependant, si les paramètres de l'image I sont connus ou estimés à l'instant t-1, seuls les 3 paramètres de l'image J sont à calculer. L'espace de recherche n'a donc plus que 3 dimensions : f_J, φ_J et θ_J ou $\Delta\theta_{I,J}$.

Pour chacune de ces dimensions, nous pouvons éventuellement poser des limites en fonction de connaissances *a priori*. Par exemple, en fonction de la vitesse d'acquisition, nous pouvons déterminer le déplacement maximum de la caméra. De même, si l'information de position de la caméra est disponible, il est possible que nous connaissions l'imprécision maximale de ces données. Il est utile de pouvoir clairement poser les limites de l'espace de recherche de façon à éviter la divergence des méthodes de recalage. Surtout en ce qui concerne les méthodes itératives.

Maintenant que nous avons clairement posé les limites de notre espace de recherche, nous allons définir des métriques pour mesurer la qualité de la mise en correspondance.

4.4. Les mesures de similarité

4.4.1. Introduction

L'intérêt principal de ces mesures est de donner une valeur numérique à une notion subjective qui est la ressemblance. Parmi une collection d'objets ou d'images, chaque individu pourra avoir un classement différent. Est-ce la couleur que l'on va privilégier, la forme ou l'utilisation ? Dans notre problématique de recalage d'images, nous recherchons une mesure de similarité qui, après transformation d'une des deux images, permet de vérifier la mise en correspondance. Nous ne sommes donc pas dans le cas de l'indexation où nous cherchons à retrouver des structures communes dans les deux images : paysage de montage, village, foule de personnes, etc. Notre objectif est de vérifier, indépendamment des structures contenues dans les images, si les zones de recouvrement des images sont identiques. Donc, dans le cas d'une simple similarité pixel à pixel, plusieurs mesures sont décrites dans la littérature. Nous allons en étudier quelques-unes de façon à sélectionner la ou les mesures en adéquation avec notre problématique. Pour plus d'informations sur les mesures de similarité, le lecteur pourra se référer aux travaux de Hill [HIL94], Bro-Nielsen [BRO97], Banks

[BAN98, BAN01], Sarrut [SAR00] et Chambon [CHA05]. Cette dernière présente une classification assez complète des différentes mesures ainsi qu'un protocole de test. Avant de présenter les mesures que nous avons étudiées, nous faisons une petite précision sur la terminologie. Par abus de langage, nous parlons toujours de mesure de similarité alors que certaines mesures sont plutôt des mesures de dissimilarité ou de distance. En effet le résultats de certaines de ces mesures augmente lorsque la similarité augmente. Dans ce cas, le terme « mesure de similarité » a tout son sens. A l'inverse, d'autres mesures retournent une valeur décroissante lorsque la similarité augmente. Nous devrions alors parler de « mesure de dissimilarité (ou de distance)». Comme nous cherchons à minimiser une fonction de coût, nous avons tendance à utiliser des mesures de dissimilarité même si nous continuons à parler de mesure de similarité. Dans le cas où nous emploierions réellement une mesure de similarité, nous pouvons toujours nous ramener à une mesure de dissimilarité.

4.4.2. Evaluation des mesures de similarité

Pour chaque mesure, nous avons utilisé le même protocole de test. Sur une image réelle, nous appliquons une transformation homographique T à partir de paramètres connus et nous calculons la mesure de similarité entre l'image de départ et l'image transformée. Nous multiplions les tests en augmentant les décalages dans les 2 directions θ et φ.

Figure 82 : Représentation du protocole de test : une même image (au centre) à laquelle nous appliquons différentes transformations

Cette première série de tests nous permet d'étudier le comportement de la mesure de similarité dans notre espace de recherche. Nous présentons les résultats sous la forme d'un graphique 3D représentant la mesure de similarité en fonction du déplacement en θ et en φ. Nous pouvons imaginer qu'une bonne mesure serait telle que le résultat est égal à zéro lorsque les images sont absolument identiques et est égal à 1 dès qu'il y a le moindre décalage. On conçoit alors aisément qu'une telle mesure ne soit pas adaptée lorsque nous utilisons un algorithme itératif de recherche d'une minimisation. Tout résultat qui n'est pas « idéal » renvoyant invariablement 1, nous ne pouvons pas progresser dans une direction particulière. Finalement la mesure « idéale » que nous recherchons doit avoir une pente relativement forte près du minimum à atteindre de façon à accélérer la progression. Mais il est aussi nécessaire que l'influence de ce minimum soit perceptible relativement loin de la position optimale de façon à augmenter la probabilité de le trouver rapidement. Ceci est donc notre premier critère. Notre deuxième critère est que la mesure de similarité doit être, autant que faire se peut, insensible aux changements de luminosité. Lorsque nous réalisons un panorama complet, les conditions de luminosité ne sont pas identiques sur l'ensemble de

la scène ou au cours du temps. Afin de s'adapter à ces différentes conditions, la caméra corrige le gain, l'ouverture de l'objectif ou le temps d'exposition. La conséquence est qu'entre deux prises de vue, l'intensité des pixels représentant une même zone de la scène ont une valeur différente alors que la structure de l'image est la même. La mesure que nous recherchons doit donc être invariante en fonction de la luminosité. Nous réalisons, pour cela, une deuxième série de mesures. A partir de l'acquisition d'une image de référence, nous faisons l'acquisition de plusieurs images dans les mêmes conditions de prise de vue en modifiant manuellement le gain de la caméra de façon à assombrir ou éclaircir l'image. Ce sont ces images auxquelles nous appliquons la transformation et que nous comparons à l'image de référence suivant le même protocole que précédemment.

Gain 50 | *Gain 75* | ***Gain 100 (réf.)*** | *Gain 120* | *Gain 150* | *Gain 215*

Figure 83 : Images du protocole de test 2

Nous présentons, sous forme graphique, la moyenne de la mesure de similarité en fonction de la distance de l'image transformée par rapport à la position de référence. La « distance » correspond à la distance Euclidienne séparant les centres des deux images sur la sphère de rayon 1.

Dans la suite de cette étude, nous présenterons toujours les mesures telles qu'elles sont exprimées dans la littérature sous le terme de « mesure de similarité » sans faire la distinction similarité/dissimilarité. Par contre, nous présentons toujours les résultats de telle sorte qu'ils minimisent la valeur de la similarité lorsque celle ci est importante entre les images de façon à les comparer plus facilement. Dit autrement, nous nous plaçons toujours dans le cas d'une mesure de dissimilarite. De plus, nous appliquons si nécessaire une pondération en fonction du nombre de points dans la zone de recouvrement ainsi qu'une normalisation pour obtenir une valeur comprise entre 0 et 1.

Enfin, pour présenter les différentes équations des mesures de similarité nous utilisons la notation suivante :

- I est l'image de référence,
- J est l'image sur laquelle nous allons appliquer la transformation T
- T est la transformation homographique que nous appliquons à l'image J,
- Ω est l'ensemble des pixels contenus dans la zone de recouvrement entre l'image I et l'image J transformée,
- x est un pixel appartenant à l'ensemble Ω.

Dans la première série de tests, J = I puisque nous cherchons à évaluer l'influence du décalage sans être perturbé par un changement de luminosité. Dans le seconde série de tests, I correspond à l'image acquise avec un gain 100 et J correspond aux autres images acquises avec un gain différent.

Nous allons explorer trois grandes familles de mesures de similarité : les mesures de distance, les mesures de corrélation et les mesures basées sur l'histogramme joint.

4.4.3. Mesures de distance

4.4.3.1 Valeur absolue de la différence (SAD)

Dans la première famille, la distance la plus simple à calculer est la somme des valeurs absolues de la différence (SAD : sum of absolute differences) :

$$SAD(I,J,T) = \sum_{x \in \Omega} \left| I(x) - J\left(T(x)\right) \right| \tag{4.6}$$

Le premier graphique (Figure 84) est une représentation 3D de l'évolution de la mesure de similarité en fonction du décalage θ et φ. Pour ce premier résultat, la luminosité est constante. Nous pouvons voir, à travers cette représentation, la forme de cuvette que nous recherchons avec un minimum bien franc lorsque les images sont parfaitement similaires.

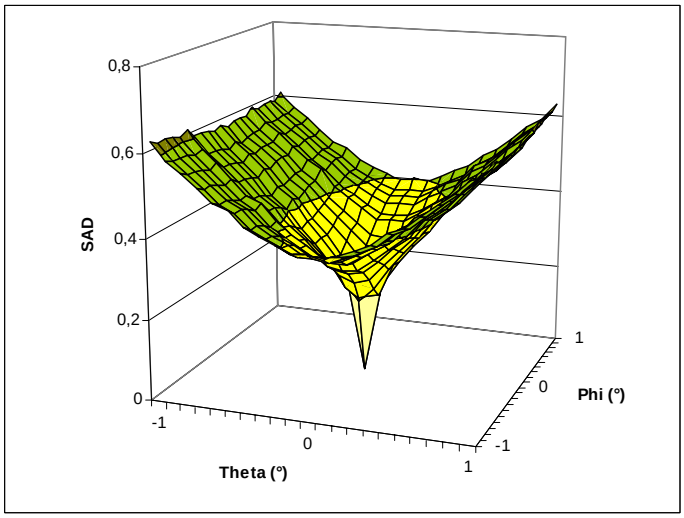


Figure 84 : Evolution de la mesure SAD en fonction du décalage θ et φ

Le graphe suivant (Figure 85) correspond à l'évolution de la mesure de similarité pour plusieurs changements de luminosité en fonction du décalage suivant l'angle de panorama et avec l'angle de tangage constant et égale à zéro. L'intérêt de cette représentation est de vérifier la robustesse de la mesure aux changements de luminosité. Dans le cas de la mesure SAD, nous pouvons voir que cette mesure est particulièrement sensible aux changements de luminosité.

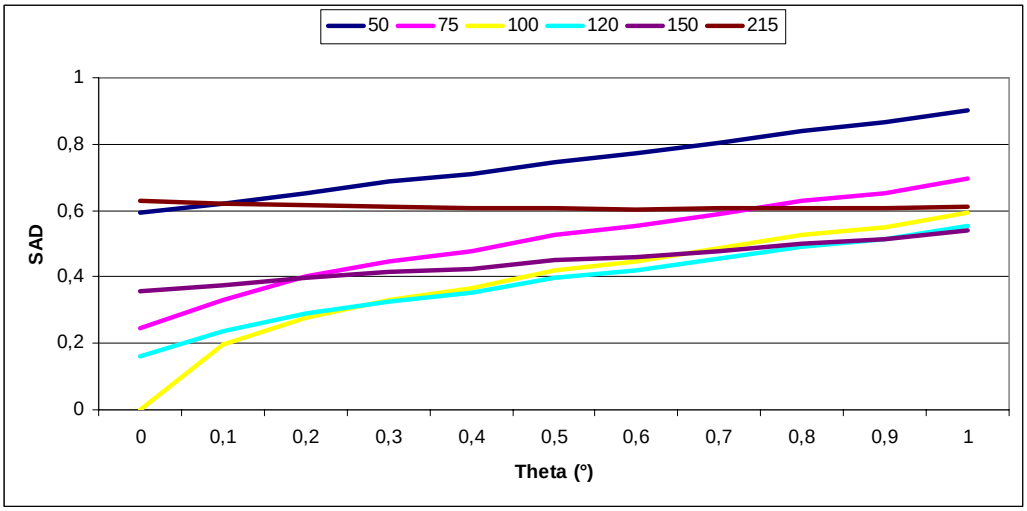


Figure 85 : Evolution de la mesure SAD en fonction du décalage θ et à φ=0 pour plusieurs changements de luminosité

Cette mesure est très simple à calculer et l'influence du minimum est sensible sur l'ensemble de l'espace de recherche. Ce dernier point est particulièrement intéressant pour les algorithmes itératifs. Cependant, cette mesure est particulièrement sensible aux changements de luminosité. Si le changement de luminosité est faible, la mesure reste valable, à un décalage près. Par contre pour les changements forts, la mesure dérive complètement à tel point que pour le gain 215, le minimum est atteint paradoxalement pour les grands déplacements. Nous avons là un effet de bord lié aux nombres de points de la zone de recouvrement des deux images. Le nombre de points est maximal lorsque le recouvrement est total et il diminue avec l'éloignement des deux images.

Nous pouvons également formuler une autre remarque valable pour les autres mesures que nous allons présenter. Sur la Figure 84, le décalage suivant l'axe φ montre une dissimilarité plus forte que suivant l'axe θ. Ceci est dû au contenu de notre image de référence. En effet, notre image peut se décomposer en deux parties. Le haut de l'image est occupé par le ciel et le bas de l'image par des arbres et des habitations. Lorsque nous faisons glisser l'image transformée suivant l'angle de panorama, les pixels du ciel de cette image sont comparés à d'autres pixels du ciel de l'image de référence. Même si les pixels ne sont pas appariés, ils restent globalement de la même couleur. Il en va de même, dans une moindre mesure, pour les pixels du bas de l'image. La dissimilarité est donc atténuée. Par contre, lorsque nous faisons glisser l'image transformée suivant l'angle de tangage, les pixels des arbres sont comparés avec ceux du ciel, d'où une dissimilarité rapidement plus importante.

Dans le cadre du recalage d'images, cette mesure est assez peu utilisée. On la trouve dans [ZHU99]. En règle générale, les auteurs lui préfèrent la mesure suivante.

4.4.3.2 Différence au carré (SSD)

La mesure la plus répandue est aussi une mesure de distance. Elle correspond à la somme des différences au carré (SSD : sum of squared differences) :

$$SSD(I,J,T)=\sum_{x\in\Omega}\left(I(x)-J(T(x))\right)^2 \qquad (4.7)$$

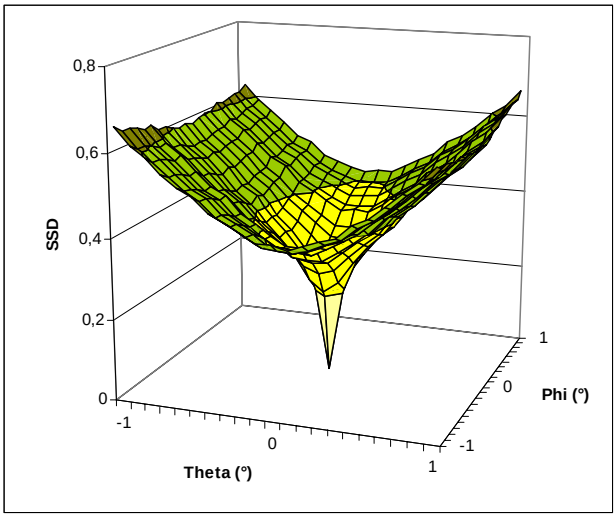


Figure 86 : Evolution de la mesure SSD en fonction du décalage θ et φ

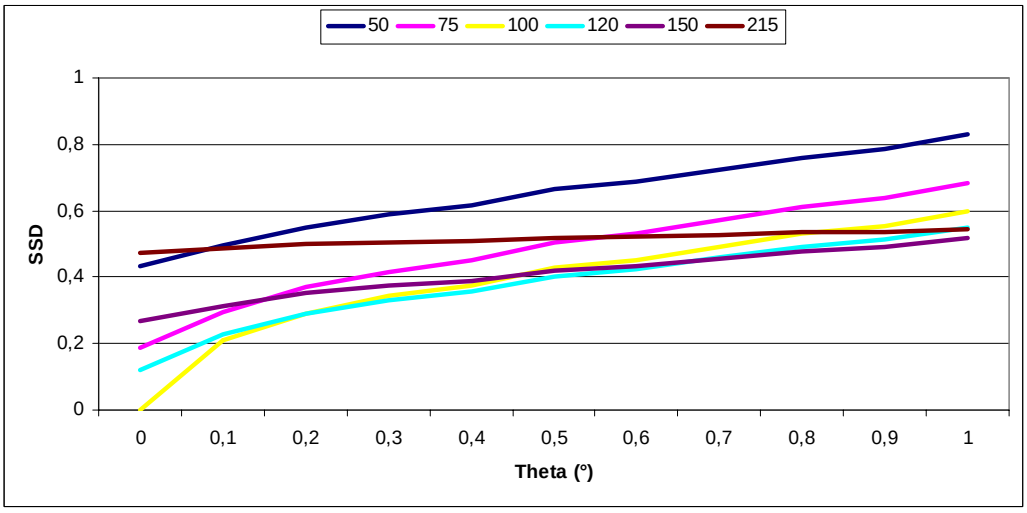


Figure 87 : Evolution de la mesure SSD en fonction du décalage θ et à φ=0 pour plusieurs changements de luminosité

Cette mesure est populaire à juste titre. Elle reste très simple à calculer. L'influence du minimum est sensible sur l'ensemble de notre zone de recherche et elle est moins sensible aux changements de luminosité que la mesure SAD. Nous avons toujours le même phénomène de décalage qui apparaît sur la valeur du minimum ainsi qu'une atténuation de la pente lorsque le changement augmente. Dans le cas des méthodes de recalage itératives, typiquement les descentes de gradient, cette atténuation de la pente peut éventuellement diminuer l'efficacité des algorithmes. Le décalage est cependant plus gênant. Nous constatons que la valeur de la mesure de deux images parfaitement recalées mais avec des gains différents de valeur 100 et 50 est comparable à celle de deux images très décalées mais prises avec le même gain. Le recalage des deux images prises avec des gains différents est bon mais la mesure est mauvaise. Dans le cas d'une recherche d'un minimum, le décalage est moins gênant puisque le minimum correspond bien à la position optimale. Par contre la

condition d'arrêt d'un algorithme itératif sera plus délicate à déterminer. Cette mesure est notamment utilisée dans [SZE94, ZHO04]

4.4.3.3 Somme des carrés des différences centrées (ZSSD)

Pour atténuer la sensibilité liée à la différence de luminosité, nous pouvons utiliser une mesure centrée ZSSD (Zero mean Sum of Squared Differences) :

$$ZSSD(I,J,T) = \sum_{x\in\Omega}\left[\left(I(x) - \bar{I}(x)\right) - \left(J(T(x)) - \bar{J}(T(x))\right)\right]^2 \qquad (4.8)$$

où $\bar{I}(x)$ et $\bar{J}(T(x))$ sont les moyennes de I et J au voisinage de x. Pour le test, nous avons utilisé un voisinage carré de taille 9x9.

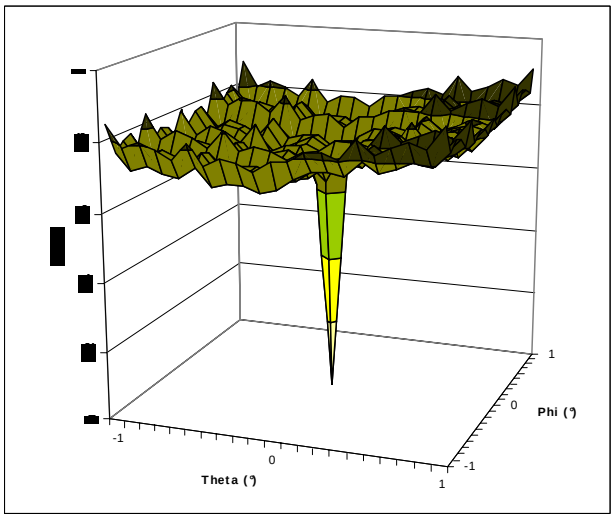


Figure 88 : Evolution de la mesure ZSSD en fonction du décalage θ et φ

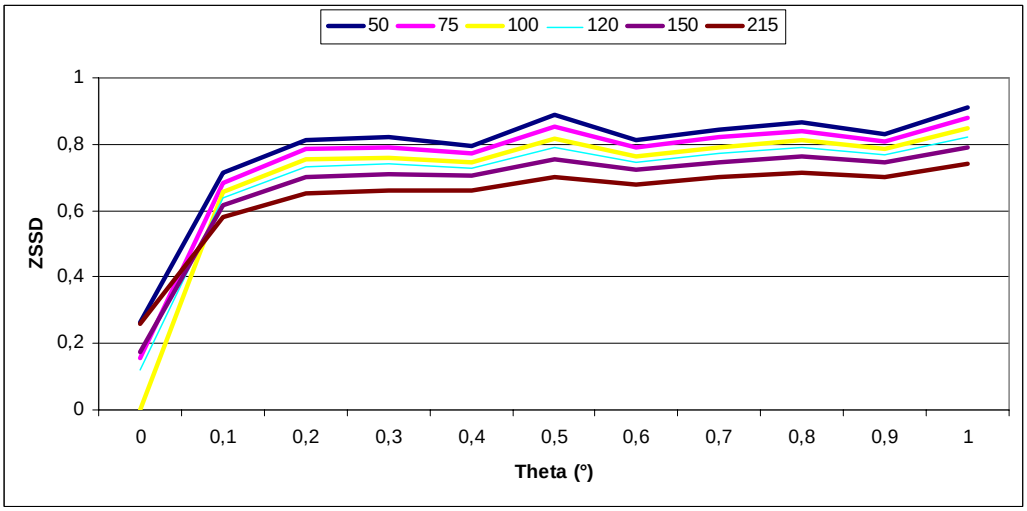


Figure 89 : Evolution de la mesure ZSSD en fonction du décalage θ et à φ=0 pour plusieurs changements de luminosité

Conformément à ce qu'affirme Jasmine Banks dans [BAN98], cette mesure est beaucoup moins sensible aux changements de luminosité. De plus la forme du pic est plus franche. Par contre, lorsque l'on s'éloigne du minimum global, cette mesure présente un relief très accidenté. Elle ne sera donc pas très adaptée pour des méthodes de recalage itératives basées sur la descente du gradient. Le risque de tomber dans un minimum local étant fort. Par contre, elle pourra être utilisée pour donner une mesure finale sur la qualité du recalage. Dans [HAN07], les auteurs utilisent cette mesure pour le recalage d'images dans une application de super-résolution.

4.4.3.4 Somme des carrés des différences centrées normalisées (ZNSSD)

Il existe également une version centrée normalisée de la mesure ZSSD. Elle est notée ZNSSD et son expression est la suivante :

$$ZNSSD(I,J,T) = \frac{\sum_{x\in\Omega}\left[\left(I(x)-\bar{I}(x)\right)-\left(J(T(x))-\bar{J}(T(x))\right)\right]^2}{\sum_{x\in\Omega}\left(I(x)-\bar{I}(x)\right)^2 \cdot \sum_{x\in\Omega}\left(J(T(x))-\bar{J}(T(x))\right)^2} \tag{4.9}$$

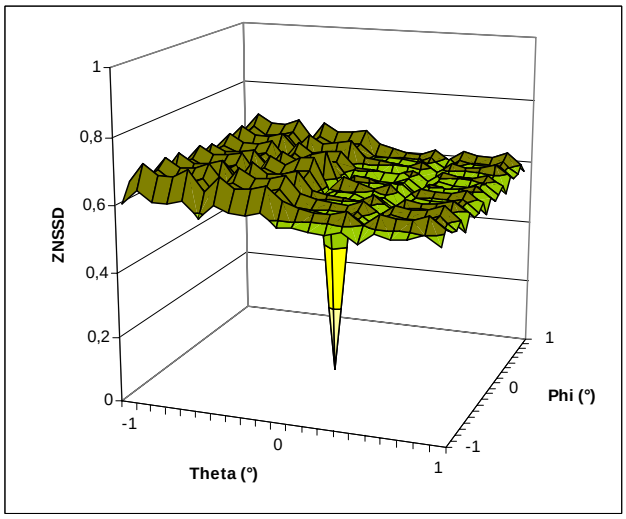


Figure 90 : Evolution de la mesure ZNSSD en fonction du décalage θ et φ

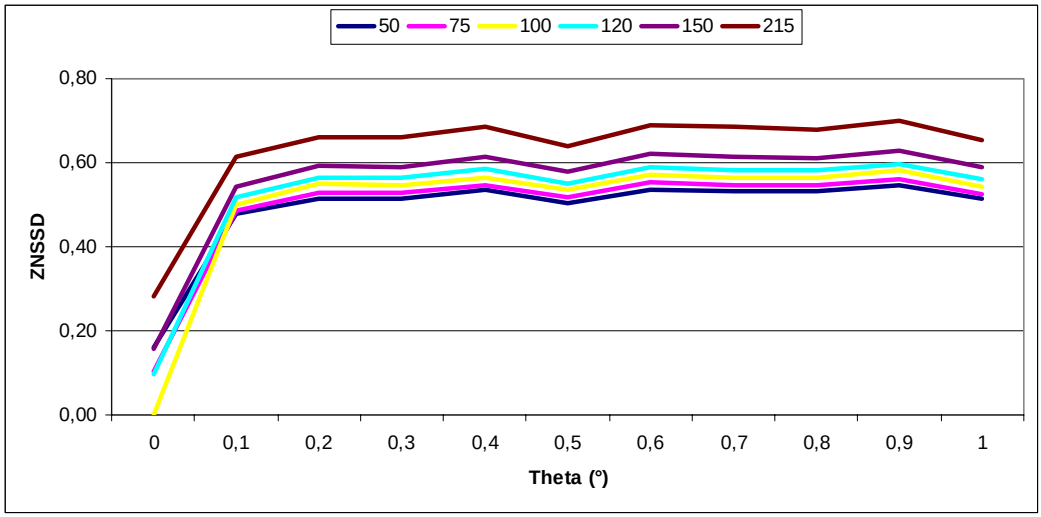


Figure 91 : Evolution de la mesure ZNSSD en fonction du décalage θ et à φ=0 pour plusieurs changements de luminosité

Le profil de l'espace de recherche est proche de la version centrée. Le pic est franc et lorsque l'on s'éloigne de la solution optimale, le profil est très accidenté. Nous aurons donc les mêmes difficultés que celles rencontrées avec la mesure ZSSD.

4.4.4. Mesures de corrélation

Nous allons maintenant présenter quelques mesures de corrélation. Les mesures de distance sont basées sur la différence entre les pixels alors que les mesures de corrélation sont basées sur la multiplication des pixels.

4.4.4.1 Corrélation croisée normalisée (NCC)

La première mesure qui consiste simplement à multiplier les pixels deux à deux n'est jamais utilisée directement puisque plus les pixels ont un niveau de gris important, plus le résultat de la mesure est important. Dans la pratique la mesure est utilisée sous sa forme normalisée (NCC : normalized cross correlation).

$$NCC(I,J,T)=\frac{\sum_{x\in\Omega} I(x)\cdot J(T(x))}{\sqrt{\sum_{x\in\Omega} I(x)^2}\cdot\sqrt{\sum_{x\in\Omega} J(T(x))^2}} \qquad (4.10)$$

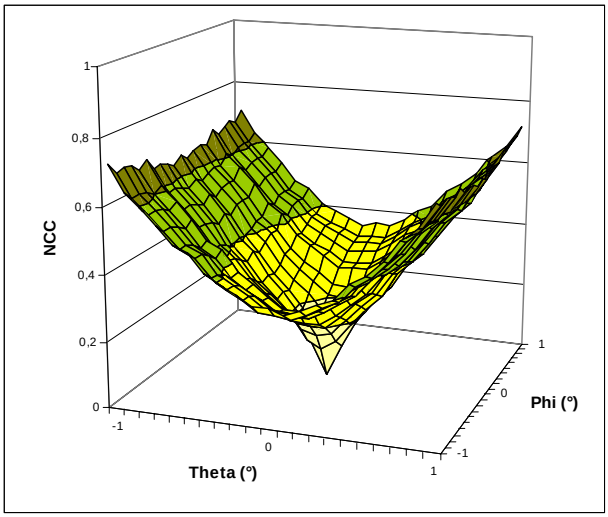


Figure 92 : Evolution de la mesure NCC en fonction du décalage θ et φ

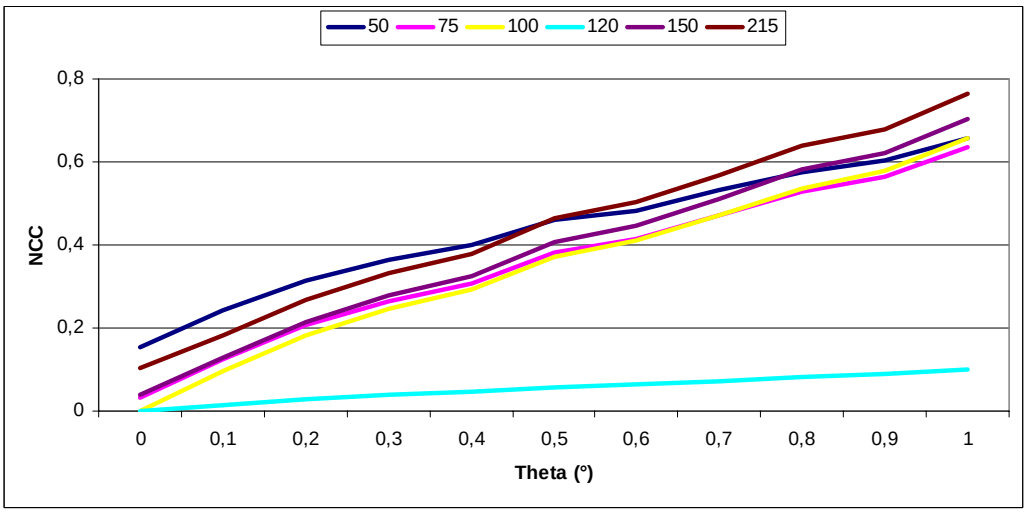


Figure 93 : Evolution de la mesure NCC en fonction du décalage θ et à φ=0 pour plusieurs changements de luminosité

Sous sa forme normalisée, la mesure de corrélation présente certaines caractéristiques intéressantes. Tout d'abord, la forme de l'influence de la solution optimale est sensible même pour les déplacements relativement importants. Pour les algorithmes basés sur la descente de gradient, cette caractéristique est importante puisque quelle que soit la (ou les) valeur(s) de départ, incluse(s) dans l'espace de recherche, l'algorithme a de bonnes chances d'atteindre le minimum global. Cette mesure est également peu sensible aux changements de luminosité. Il reste néanmoins un décalage mais il est moins important que pour les mesures SAD ou SSD.

4.4.4.2 Corrélation croisée centrée normalisée (ZNCC)

Comme pour la mesure SSD, une mesure centrée et normalisée est proposée dans la littérature. Elle est notée ZNCC (Zero-mean normalized cross correlation).

$$ZNCC(I,J,T)=\frac{\sum_{x\in\Omega}\left(I(x)-\bar{I}\right)\left(J(T(x))-\bar{J}\right)}{\sqrt{\sum_{x\in\Omega}\left(I(x)-\bar{I}\right)^2}\cdot\sqrt{\sum_{x\in\Omega}\left(J(T(x))-\bar{J}\right)^2}} \quad (4.11)$$

Pour le test, nous avons utilisé un voisinage de taille 9x9.

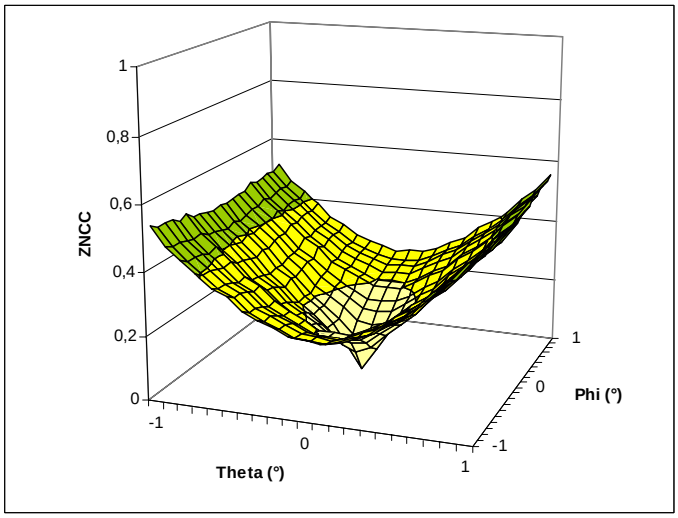


Figure 94 : Evolution de la mesure ZNCC en fonction du décalage θ et φ

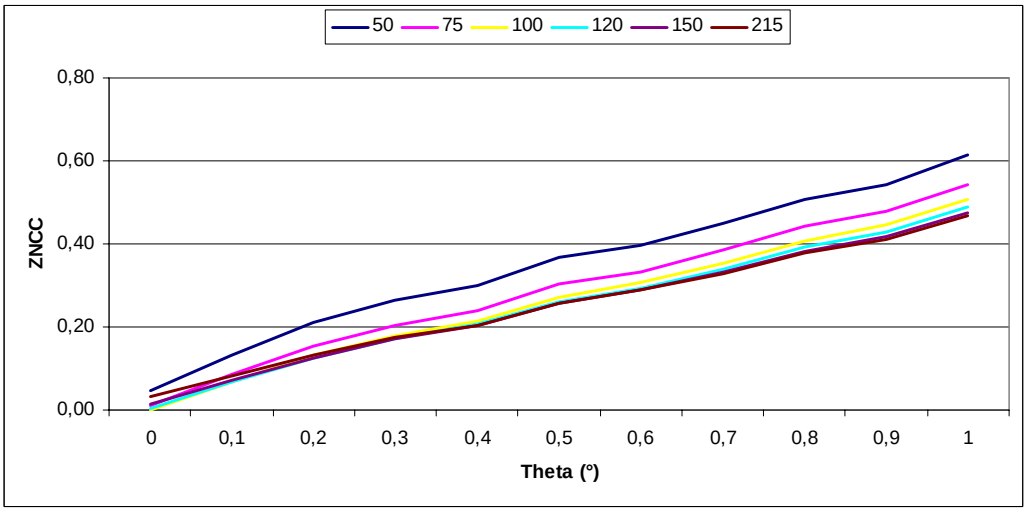


Figure 95 : Evolution de la mesure ZNCC en fonction du décalage θ et à φ=0 pour plusieurs changements de luminosité

Par rapport à la mesure NCC, la mesure ZNCC est plus robuste aux changements de luminosité. Par contre, contrairement à la mesure ZNSSD, le graphe du haut garde une forme évasée ce qui permet aux algorithmes de recalage itératifs basés sur la descente du gradient, de garder toute leur efficacité. Cette mesure est l'une des plus utilisée. Une version plus robuste est proposée dans [TRU04].

4.4.5. Histogramme joint

4.4.5.1 définition de l'histogramme joint

Nous venons de voir quelques exemples de mesures de similarités basées sur la notion de distance ou de corrélation. Il existe une autre classe de mesures basées sur la notion d'histogramme joint. Comme indiqué dans [SAR00], l'histogramme joint est une généralisation des matrices de cooccurrence utilisées en analyse de texture. La construction de l'histogramme joint consiste à comptabiliser les occurrences des couples de valeur de pixels (I(x), J(T(x))) dans la zone de recouvrement. Dans le cas d'images en niveau de gris 8 bits, l'histogramme joint est une matrice P de taille 256x256. Une cellule de cette matrice est notée pi,j avec $0 \leq i \leq 255$ et $0 \leq j \leq 255$. La somme

$$N = \sum_{i=0}^{255} \sum_{j=0}^{255} p_{i,j} \tag{4.12}$$

correspond au nombre de pixels dans la zone de recouvrement.

Théoriquement, si deux images sont parfaitement recalées, alors seules les cellules de la diagonale sont incrémentées. Les images suivantes (Figure 96) représentent l'histogramme joint obtenu lorsque :

(a) les images sont bien recalées et qu'il n'y a pas eu de changement notable de la luminosité,
(b) les images sont bien recalées mais il y a eu un changement significatif de la luminosité,
(c) les images ne sont pas recalées mais sans changement de luminosité.

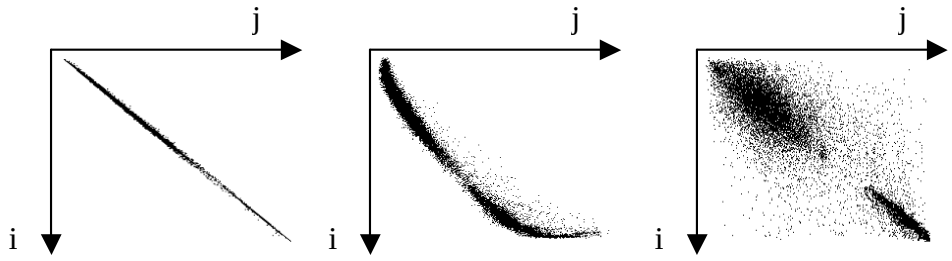


Figure 96 : Exemples d'histogramme joint (les pixels noirs correspondent aux cellules qui ont été incrémentées).
(a) Images recalées sans changement de luminosité,
(b) Images recalées mais changement significatif de luminosité
(c) Images non recalées sans changement de luminosité

Sur l'image (Figure 96b) nous pouvons remarquer le glissement de la diagonale lié au changement de luminosité. Nous pouvons même en déduire que l'image J est plus sombre que l'image I puisque pour une valeur d'intensité donnée dans l'image I, nous avons statistiquement une valeur d'intensité plus faible dans l'image J.

4.4.5.2 Information mutuelle (IM)

Le principe des mesures de similarité utilisant l'histogramme joint consiste à déterminer l'écart de l'histogramme par rapport à la diagonale. D. Sarrut [SAR00] montre que la plupart des mesures de similarités existantes, dont celles présentées ci dessus, peuvent être calculées à partir d'un histogramme joint.

L'information mutuelle est l'une des mesures classiques utilisant l'histogramme joint. La définition de cette mesure est donnée ci-dessous :

$$MI(I,J,T)=\sum_{i,j} p_{i,j}\cdot\log\left(\frac{p_{ij}}{p_i\cdot p_j}\right) \tag{4.13}$$

Les résultats des tests sont les suivants :

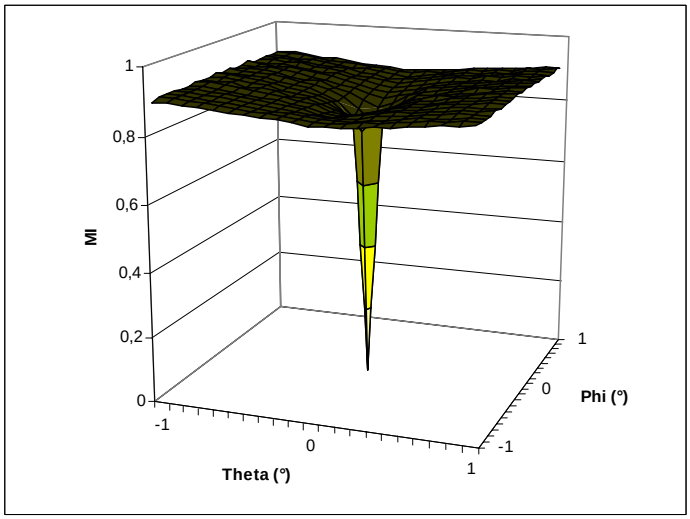


Figure 97 : Evolution de la mesure IM en fonction du décalage θ et φ

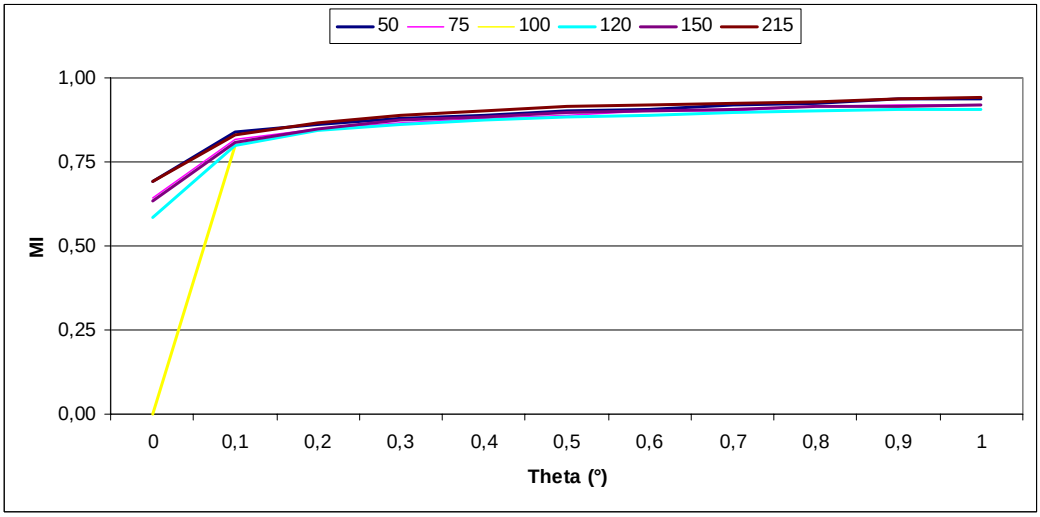


Figure 98 : Evolution de la mesure IM en fonction du décalage θ et à φ=0 pour plusieurs changements de luminosité

Le diagramme (Figure 97) montre, que dans le cas de la mesure d'information mutuelle, le minimum global a une valeur particulièrement significative. L'influence de ce minimum reste sensible sur l'ensemble du domaine de recherche même si il est très atténué. Le profil ne montre pas de minimum local important. La mesure est également peu sensible aux changements de luminosité par contre, la valeur du minimum est très différente lorsque la luminosité évolue.

4.4.6. Conclusion

Nous avons présenté les 6 mesures de similarité que nous rencontrons le plus couramment dans la littérature. Cette présentation est loin d'être exhaustive. Dans sa thèse, Sylvie Chambon [CHA03] en présente une quarantaine. Nous avons simplement cherché, parmi les plus classiques, la mesure la plus adaptée à notre besoin. Les mesures de distances SAD et SSD présentent un profil très intéressant pour les algorithmes de recherche d'une minimisation. L'influence du minimum local est significative sur l'ensemble de l'espace de recherche. Intuitivement nous pouvons prédire que si nous lâchons une bille n'importe où sur l'espace de recherche, la bille atteindra le minimum global. De plus, nous constatons une accélération à l'approche de ce minimum. Par contre ces mesures sont sensibles aux changements de luminosité. Les mesures ZSSD et ZNSSD sont beaucoup moins sensibles aux changements de luminosité, par contre leur profil est trop accidenté lorsqu'on s'éloigne du minimum global. Finalement, les mesures de corrélation sont les plus adaptées à notre besoin. Elles présentent un profil intéressant pour les algorithmes de recherche d'une minimisation et sont peu sensibles aux défauts de luminosité. Le tableau suivant est une synthèse des tests que nous avons effectués :

Nom	Formulation	IG[1]	DG[2]	TC[3]
SAD	$SAD(I,J,T)=\sum_{x\in\Omega}\lvert I(x)-J(T(x))\rvert$	--	++	0.32
SSD	$SSD(I,J,T)=\sum_{x\in\Omega}(I(x)-J(T(x)))^2$	-	++	0.36
ZSSD	$ZSSD(I,J,T)=\sum_{x\in\Omega}\left[\left(I(x)-\bar{I}(x)\right)-\left(J(T(x))-\bar{J}(T(x))\right)\right]^2$	+	--	0.76
ZNSSD	$ZNSSD(I,J,T)=\frac{\sum_{x\in\Omega}\left[\left(I(x)-\bar{I}(x)\right)-\left(J(T(x))-\bar{J}(T(x))\right)\right]^2}{\sum_{x\in\Omega}\left(I(x)-\bar{I}(x)\right)^2\cdot\sum_{x\in\Omega}\left(J(T(x))-\bar{J}(T(x))\right)^2}$	+	--	1.00
NCC	$NCC(I,J,T)=\frac{\sum_{x\in\Omega}I(x)\cdot J(T(x))}{\sqrt{\sum_{x\in\Omega}I(x)^2}\cdot\sqrt{\sum_{x\in\Omega}J(T(x))^2}}$	+	+	0.46
ZNCC	$ZNCC(I,J,T)=\frac{\sum_{x\in\Omega}\left(I(x)-\bar{I}\right)\left(J(T(x))-\bar{J}\right)}{\sqrt{\sum_{x\in\Omega}\left(I(x)-\bar{I}\right)^2}\cdot\sqrt{\sum_{x\in\Omega}\left(J(T(x))-\bar{J}\right)^2}}$	++	+	1.15
IM	$MI(I,J,T)=\sum_{i,j}p_{i,j}\cdot\log\left(\frac{p_{ij}}{p_i\cdot p_j}\right)$	+	+	2.03

Tableau 6 : Synthèse des mesures de similarité

(1) IG → Insensibilité au gain
(2) DG → Profil intéressant pour les algorithmes de recherche de minimisation

(3) TC → Temps de calcul en ms sur des images 320 x 240 (AMD 1,58GHz)

4.5. Algorithmes de recalage

Pour la présentation des méthodes de recalage, nous avons repris la classification traditionnelle qui consiste à définir deux catégories : dense et éparse. Nous abordons en premier le cas des méthodes denses à travers 4 méthodes que nous avons expérimentées. La présentation des méthodes éparses suit une progression un peu différente. Ces méthodes sont, pour la plupart, une succession d'étapes intermédiaires (recherche des points d'intérêt, appariement, optimisation …), pour lesquelles il peut y avoir différentes solutions.

4.5.1. Méthodes denses

Le principe général des méthodes denses consiste à explorer l'espace des transformations possibles de façon à minimiser la mesure de (dis)similarité. Nous avons alors à faire à un problème d'optimisation globale qui se décompose en deux grandes approches : déterministe et aléatoire. Dans la première approche, les algorithmes utilisent toujours le même cheminement en fonction des conditions initiales. Parmi ces méthodes nous pouvons citer : la descente de gradient, la minimisation de Gauss-Newton ou encore la minimisation de Levenberg-Maquardt. Si nous faisons abstraction de la phase d'initialisation, la méthode du simplexe, que nous allons voir en détail un peu plus loin, se classe également dans cette catégorie. Dans le cas des méthodes dites aléatoire, le cheminement est dicté, en partie, par des lois stochastiques. A partir des mêmes conditions initiales, le cheminement et surtout le résultat final n'est pas forcément le même. Par contre ces méthodes sont généralement réputées pour être moins sensibles aux minima locaux. Nous allons tout d'abord présenter la méthode déterministe de Szeliski [SZE94]. Cette méthode consiste à affiner itérativement les paramètres de la matrice d'homographie. Dans la classe des méthodes déterministes, nous verrons également la méthode du simplexe. Nous définissons la méthode du simplexe comme étant déterministe parce que pour une initialisation donnée du simplexe, qui peut être aléatoire, l'évolution du simplexe est déterministe. Nous présentons ensuite des méthodes stochastiques que nous avons adaptées à notre problématique. Nous verrons la méthode du recuit simulé ainsi qu'une adaptation des algorithmes génétiques.

4.5.1.1 Méthode de Szeliski

Une méthode de recalage classique utilisée pour la construction d'une mosaïque d'images est proposée pour la première fois par Szeliski dans [SZE94]. Cette méthode permet de calculer la matrice d'homographie H d'une image I_1 dans I_0 par itérations successives. L'algorithme fait intervenir une matrice de correction D permettant la mise à jour de la matrice H. Le calcul des paramètres de la matrice de correction s'effectue en recherchant la minimisation d'une fonction de coût. Szeliski propose d'utiliser la mesure SSD.

L'algorithme est finalement assez simple. La matrice H de départ est initialisée soit avec des paramètres approchés de l'homographie par une autre méthode ou par des informations issues de la caméra soit avec la matrice identité. L'image du gradient, la matrice jacobienne et la matrice de l'erreur d'intensité sont ensuite calculées à partir de l'image I_1 transformée par la matrice H de départ. A partir de ces données, la matrice de correction D est évaluée en utilisant la méthode des moindres carré. Cette matrice est ensuite appliquée à la matrice H. Par itérations successives, on affine les valeurs de la matrice H. L'algorithme s'arrête sur un critère d'arrêt qui peut être un nombre maximum d'itérations atteint, un calcul sur la somme

ou la somme au carré des valeurs de la matrice d'erreur d'intensité ou encore sur la somme ou la somme au carré des paramètres de la matrice D. Une présentation plus complète de cet algorithme est donnée en annexe.

La méthode de Szeliski donne de très bons résultats lorsque l'écart de position entre les images est très faible. Cependant, les résultats sont nettement moins bons lorsque les écarts sont importants. De plus, c'est une méthode particulièrement lente. Le temps de calcul est de l'ordre d'une seconde sur notre machine (AMD 1,58GHz). Il est cependant à noter que nous n'avons pas spécialement cherché à optimiser notre code.

4.5.1.2 Méthode du Simplexe

La méthode de Szeliski ayant un temps de calcul incompatible avec le temps réel, nous allons étudier la méthode du simplexe. Cette méthode est un grand classique la recherche d'une minimisation dans un espace à plusieurs dimensions. Elle a été introduite par Nelder et Mead en 1965 [NEL65]. Cette méthode repose sur l'utilisation d'un polyèdre de dimension n+1 qui est modifié à chaque itération pour se rapprocher du minimum d'une fonction de coût où n est le nombre de paramètres de la transformation que nous recherchons. Dans le paragraphe 4.3, nous avons montré que dans certaines conditions, l'espace de recherche n'a que 3 dimensions. En conséquence, le polyèdre du simplexe possède 4 sommets, c'est-à-dire un tétraèdre. A chaque sommet est associé une transformation H calculée à partir des paramètres θ, φ, f et le résultat d'une fonction de coût C_{Sk} (sk = s1 .. s4).

La première étape de la méthode du simplexe consiste à initialiser les sommets du polyèdre. Pour chaque sommet k, nous fixons arbitrairement les valeurs θ, φ, f de la transformation homographique, nous calculons les images transformées de J et nous calculons la fonction de coût C_{Sk}. Dans la pratique, les valeurs de θ, φ, f ne sont pas fixées totalement de façon arbitraire puisque nous connaissons une position approchée de la caméra. Nous choisissons donc des points au voisinage de celui donné par la caméra. A chaque itération, la progressions du simplexe est régie pas règles défini dans [NEL65] que nous avons résumées en annexe.

L'algorithme prend fin lorsque la taille du simplexe atteint un certain minimum fixé ou que la fonction de coût du sommet le plus faible atteint elle aussi un minimum ou encore lorsqu'un certain nombre d'itérations ont été effectuées.

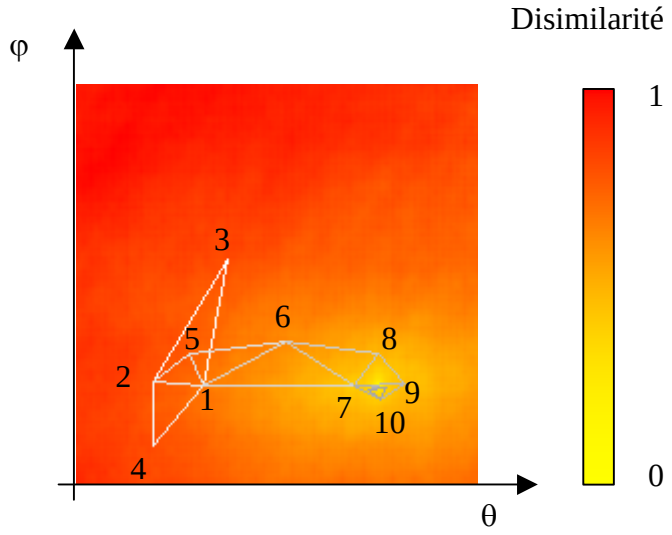


Figure 99 : Exemple de progression du simplexe dans un espace de recherche θ, φ

La méthode du simplexe que nous avons implémentée est beaucoup plus rapide que la méthode que Szeliski. Le temps de calcul est de l'ordre de 300 ms. Elle est surtout plus efficace lorsque les déplacements entre les deux images sont importants. L'utilisation de la mesure corrélation croisée normalisée permet d'éviter à la méthode de rester coincée dans un minimum local.

4.5.1.3 Recuit simulé

Le recuit simulé est une méthode de recherche stochastique qui s'inspire d'un procédé utilisé en métallurgie. Afin d'obtenir un arrangement régulier des atomes à l'état solide, on part de l'état liquide et on refroidit le métal très lentement. Si l'arrangement des atomes n'est pas satisfaisant, on réchauffe le métal sans aller jusqu'au point de fusion, mais suffisamment pour laisser une certaine liberté de mouvement aux atomes. Ce réchauffement est appelé recuit. L'algorithme du recuit simulé s'inspire de cette approche. Le principe consiste à explorer aléatoirement l'espace de recherche autour d'une position courante en fonction d'une « température » t. Plus la température est élevée, plus on s'autorise à explorer loin de la solution courante. Plus la température est basse, plus l'espace de recherche est restreint. A partir d'une température élevée, on diminue progressivement la température à chaque fois que la fonction de coût de la nouvelle solution est meilleure que la solution courante. Cependant, on s'autorise aléatoirement à accepter une solution dont la fonction de coût est supérieure à la solution courante. Le but est de permettre à l'algorithme de franchir un minimum local. Cet algorithme ne garantit pas d'atteindre le minimum local, mais dans la mesure où la température est élevée, il peut permettre de s'en approcher. Un organigramme de cet algorithme est donné en annexe.

L'implémentation de cet algorithme que nous avons testée n'est pas celle qui donne les meilleurs résultats. Cependant, elle reste robuste même pour les déplacements importants. La principale difficulté que nous avons rencontrée correspond à la fonction de décroissance de la température à utiliser. Le temps de calcul est équivalent à la méthode du simplexe.

4.5.1.4 Algorithme génétique

Les algorithmes génétiques sont également des algorithmes d'optimisation stochastiques. A l'instar de l'algorithme du recuit simulé, les algorithmes génétiques s'inspirent d'un phénomène naturel. Ici, ce sont les principes d'évolution et de sélection naturelle proposées par Darwin que l'on va chercher à reproduire. Dans la nature, les individus se croisent et se reproduisent en interagissant les uns avec les autres tout en s'adaptant à l'environnement. Les premiers travaux sur les algorithmes génétiques datent des années 1960-1970 avec John Holland. C'est en 1989 que D.E Golberg popularise ces algorithmes [GOL89] en décrivant leur utilisation dans le cadre de la résolution de problèmes concrets. Pour Lerman et Ngouenet [LER95], Les algorithmes génétiques diffèrent des algorithmes classiques d'optimisation et de recherche essentiellement en quatre points fondamentaux :

- Les algorithmes génétiques utilisent un codage des éléments de l'espace de recherche et non pas les éléments eux-mêmes.
- Les algorithmes génétiques recherchent une solution à partir d'une population de points et non pas à partir d'un seul point.
- Les algorithmes génétiques n'imposent aucune régularité sur la fonction étudiée (continuité, dérivabilité, convexité…). C'est un des gros atouts des algorithmes génétiques.
- Les algorithmes génétiques ne sont pas déterministes, ils utilisent des règles de transition probabilistes.

Le succès de ces algorithmes est justifié par leur simplicité de mise en œuvre et par leur performance. Ils ne fournissent pas nécessairement la solution globale mais permettent de généralement de s'approcher suffisamment prés de la solution optimale.

Dans notre cas particulier, le génotype de chaque individu est composé de trois gènes. Ces trois gènes correspondent aux trois inconnues (f_J, φ_J et $\Delta\theta_{I,J}$) de la transformation homographique entre deux images. Nous pourrions très bien généraliser l'algorithme aux 6 paramètres de la transformation. Pour le critère de performance, nous utilisons la mesure de similarité et pour le critère d'arrêt nous fixons un seuil sur la mesure de similarité du meilleur individu de chaque génération. Nous préservons systématiquement le meilleur individu à chaque génération de façon à assurer une décroissance monotone de l'algorithme.

Comme pour le recuit simulé, les tests que nous avons effectués avec cette méthode ne sont pas ceux qui donnent les meilleurs résultats, mais la méthode reste robuste pour les déplacements important. L'algorithme est simple à mettre en œuvre. La difficulté cependant est d'une part de fixer la taille de la population et d'autre part de définir la proportion de cette population qui sera sélectionnée, mutée et croisée. Une population importante assure une convergence plus rapide. Cependant, chaque itération est plus longue à calculer. En revanche, une population faible converge moins rapidement mais est plus rapide à calculer.

4.5.2. Méthodes éparses

Le principe des méthodes éparses consiste généralement à mettre en correspondance des points particuliers ou des structures géométriques de façon à disposer de suffisamment d'équations pour résoudre le système linéaire et déterminer la transformation mathématique entre les deux images.

Les méthodes éparses que nous allons présenter sont donc basées sur ce principe et reprennent, pour la plupart, le schéma suivant :

- Détection des points d'intérêts dans les deux images. Cette première étape consiste à sélectionner des points suffisamment caractéristiques pour être susceptibles d'être appariés sans ambiguïté. En général, les points à fort gradient sont privilégies. Dans notre étude nous nous sommes limités au cas des points d'intérêts. C'est la solution la plus courante, mais certains auteurs utilisent d'autres types de primitives géométriques (ligne, cercle).
- Mise en correspondance des points. Cette étape consiste à déterminer des couples de points entre les deux images et dont les caractéristiques sont proches.
- Première estimation des paramètres de la transformation et suppression de mauvais appariement (outliers). Cette étape est relativement classique lorsque l'on résout un système de plusieurs équations par la méthode des moindres carrés. Elle consiste à calculer une première solution avec l'ensemble des équations à disposition puis à appliquer la transformation sur les données. Les équations dont les résultats s'écartent trop de la mesure sont alors rejetées.
- Estimation de la solution finale après suppression des « outliers ».

Une variante de cette approche consiste non pas à détecter les points d'intérêts dans la deuxième image, mais de suivre ceux de la première image au cours du temps. Cette approche est essentiellement utilisée dans les séquences vidéo pour lesquelles le déplacement de la caméra entre deux images successive est relativement faible. Nous allons voir l'ensemble de ces différentes étapes de façon plus approfondie.

4.5.2.1 Détection des points d'intérêts

La première étape consiste donc à sélectionner des points caractéristiques dans les deux images. La détection de points d'intérêts n'est pas un problème récent et plusieurs solutions existent déjà. Nous nous sommes limités à étudier uniquement le détecteur de Harris. Dans [SCH00], le lecteur trouvera la comparaison de plusieurs détecteurs réalisée par Schimd et Mohr. Cependant, l'algorithme le plus cité dans la littérature est celui de Harris [HAR88] qui découle des travaux de Moravec [MOR77]. Ce détecteur s'attache à localiser des points où les variations différentielles sont importantes dans plus d'une direction de façon à ne pas inclure les contours.

Ce détecteur, bien qu'étant le plus utilisé, n'est pas toujours facile à mettre en œuvre, puisque cinq paramètres sont utilisés (cf annexe) : la taille du filtre gaussien, le filtre dérivatif, le paramètre k, le voisinage des maxima locaux et le seuil final. Une adaptation de ce détecteur est proposée par Zheng [ZHE99].

A l'issue de cette première étape, nous disposons de deux ensembles de points.

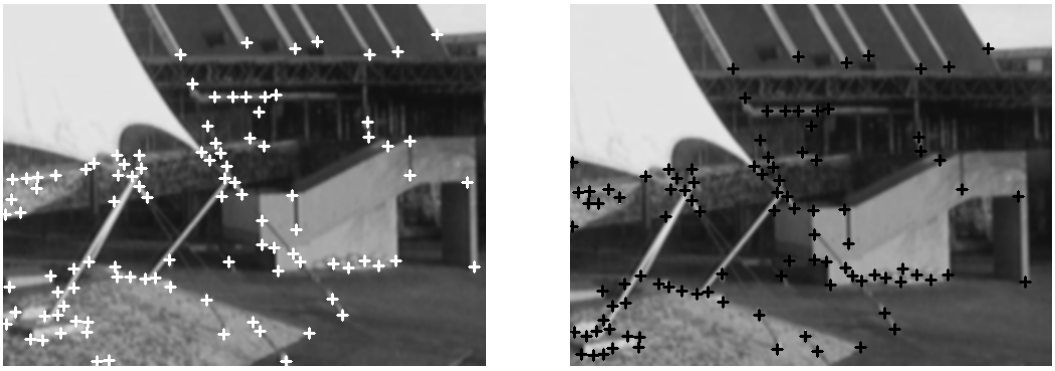


Figure 100 : Détection des points d'intérêts dans deux images avec le détecteur de Harris

4.5.2.2 Liste des candidats possibles

A partir des deux ensembles de points, une solution consiste à tester toutes les combinaisons possibles et de ne garder que la meilleure. Cependant, nous pouvons utiliser certaines connaissances *a priori* de façon à limiter les comparaisons. Pour chaque point de l'image I ($p_{i,I}$), nous calculons la zone de recherche dans l'image J correspondant au décalage maximal autorisé. Par exemple, si nous estimons que le décalage entre les informations de positions données par la caméra et sa position réelle est $\Delta\theta$ et $\Delta\varphi$, nous calculons la projection du point $p_{i,I}$ dans l'image J en tenant compte de ce décalage. Afin de limiter les calculs, nous ne calculons que les 4 points correspondant aux angles : $\theta_2'=\theta_2\pm\Delta\theta$ et $\varphi_2'=\varphi_2\pm\Delta\varphi$. Nous obtenons un polygone à 4 cotés délimitant l'espace de recherche. L'estimation de la distance focale est un cas particulier. Si la distance focale est plus faible, les polygones de recherche sont plus petits alors que si la distance focale augmente, la taille des polygones augmente également. Nous nous plaçons volontairement dans le cas le plus défavorable en calculant chaque polygone à partir de la plus grande distance focale de notre espace de recherche.

Les correspondances possibles sont donc effectuées entre tous les points de l'image J situés à l'intérieur de ce polygone.

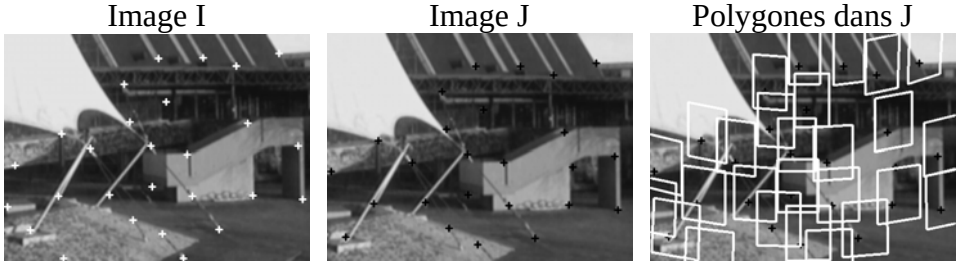


Figure 101 : Espace de recherche pour φ_I = 45°, $\Delta\theta$ = 3°, $\Delta\varphi$ = 3° et Δf = 100

L'exemple ci-dessus illustre la forme des polygones de recherche pour une valeur de φ_I = 45°. Dans l'image I, les points d'intérêt sont symbolisés par des croix blanches. Dans l'image J (décalée de $\Delta\theta_{réel}$ = 1° et $\Delta\varphi_{réel}$ = 1° par rapport à l'image I), les points d'intérêt sont symbolisés par des croix noires. Dans la figure de droite, nous représentons la projection dans l'image J des polygones de recherche de chaque point d'intérêt de l'image I. Afin de ne

pas surcharger cette figure, nous avons volontairement étendu l'espace minimum entre les points d'intérêts et donc limité leur nombre. Pour obtenir un recalage de qualité, nous devons augmenter ce nombre de points. Cependant, l'augmentation du nombre de points implique d'une part une augmentation du temps de calcul et d'autre part une augmentation du nombre de correspondances possibles pour chaque point.

4.5.2.3 Mise en correspondance

L'étape suivante est la mise en correspondance. Cette mise en correspondance consiste à déterminer les couples de points dont les caractéristiques sont communes. Les caractéristiques de chaque point peuvent être calculées en tenant compte de la couleur ou de la texture au voisinage des points. Certains auteurs utilisent également le résultat du détecteur de Harris. Les résultats de la comparaison entre tous les points sont donc placés dans un tableau de (n x m) éléments, où n est le nombre de points retenus dans l'image I et m le nombre de points retenus dans l'image J. Ce tableau est parcouru de façon à trouver la valeur maximum. Lorsque la valeur maximum est déterminée, les deux points sont appariés et la ligne et la colonne correspondant aux deux points sont effacées. Ce processus se poursuit jusqu'à ce que le min(n, m) points soit atteint où si le maximum trouvé n'atteint pas un seuil fixé à l'avance. En supposant que m est du même ordre de grandeur que n, cet algorithme de recherche est d'une complexité en $O(n^3)$ en l'état et pourrait aisément voir sa complexité ramenée à $O(n^2\log(n))$ si un tri préalable des n^2 valeurs était effectué. Ce problème de couplage bipartie pondéré pourrait être résolu plus efficacement en utilisant les tas de Fibonacci. Le nombre « n » de points à traiter étant relativement faible (inférieur à 100), ces optimisations algorithmiques auraient présenté un intérêt essentiellement théorique.
Une autre solution assez classique consiste à rechercher pour tous les points de la première image le point le plus semblable dans la deuxième image puis d'intervertir le processus. Les appariements retenus sont les paires de points qui se sont choisi mutuellement.

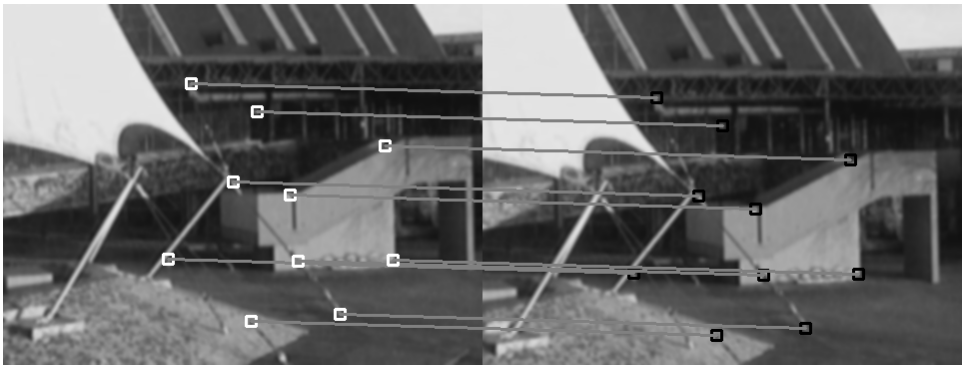


Figure 102 : Mise en correspondance des points d'intérêts dans deux images

Le fait d'utiliser la liste des candidats possibles que nous avons évoquée dans le paragraphe précédent permet de limiter le risque de faux appariement. De plus, comme tous les candidats possibles d'un même point sont tous dans un même voisinage cela évite de calculer des solutions complètements aberrantes. Le risque évidemment est que des mauvais appariements peuvent conduire à des solutions acceptables sans pour autant être optimales.

La mise en correspondance nécessite de disposer d'un descripteur local robuste aux transformations que l'on peut faire subir à l'image. Parmi les descripteurs locaux, les plus utilisés sont :

- Les invariants différentiels dont le *jet local* qui caractérise la surface locale en niveau de gris par un développement de Taylor [KOE87]. Dans [SCH97] Schmid utilise ce descripteur calculé jusqu'à l'ordre 3 pour l'indexation d'image. Les invariants différentiels ont été étendus aux images couleur par Gouet dans [GOU00]
- Le détecteur SIFT (*Scale Invariant Feature Transform*) basé sur le calcul un histogramme des orientations. Ce détecteur proposé par Lowe [LOW04] est considéré aujourd'hui comme l'un des plus performants. L'intérêt de ce détecteur est qu'il est invariant aux transformations affines (changement d'échelle, rotation et translation) et qu'il permet de calculer un descripteur robuste. Ceci le rend particulièrement efficace dans le cas de recalage rigide.
- Les invariants fréquentiels comme la transformée de Fourier-Mellin, la transformée de Gabor ou la transformée en ondelettes.

Avec la suppression des « outliers » que nous avons évoquée plus haut, ces trois étapes constituent l'ossature des méthodes de recalage éparse que nous avons étudiées. Afin de déterminer la qualité de ces méthodes, nous avons implémenté une première solution en utilisant les détecteurs SIFT décrit par Lowe [LOW04]. L'algorithme SIFT est très efficace cependant, le temps de calcul ainsi que l'espace mémoire nécessaire à son exécution sont assez important, en tout cas incompatible avec une application temps réel. Dans son article Lowe indique un temps de calcul de l'ordre de 300ms. L'implémentation que nous avons réalisée donne un temps de calcul de l'ordre la seconde. Cependant, notre but n'a pas été d'utiliser cet algorithme dans notre application mais simplement de disposer d'un algorithme de comparaison reconnu comme efficace et précis par la communauté.

Les résultats que nous obtenus avec la méthode « local jet » sont plus en adéquation avec notre problématique. Le recalage est stable sur l'ensemble de l'espace de recherche, mais surtout, le temps de calcul est de l'ordre de 15ms. Une description plus complète de ces deux descripteurs est donnée en annexe.

4.5.2.4 RANSAC

Une alternative à l'approche qui consiste à constituer des paires de points à partir de la similarité de leurs caractéristiques locales est de définir aléatoirement un sous ensemble de paires de points et de tester plusieurs solutions pour ne garder que la meilleure. Cette approche s'inspire de l'algorithme RANSAC (*RANdom SAMple Consensu*) exposé par Martin A. Fischler et Robert C. Bolles dans [FIS81]. A l'origine cet algorithme a été développé pour résoudre des problèmes liés à la cartographie.

Dans notre cas, la solution que nous recherchons correspond à l'homographie entre les deux images qui s'exprime à partir de 8 paramètres indépendants. Nous avons donc besoins de 4 paires de points pour résoudre le système linéaire. Le principe de l'algorithme est alors le suivant. A partir de deux ensembles de points, nous définissons aléatoirement un sous ensemble de 4 paires de points. Ces 4 paires de points nous permettent de calculer une matrice de transformation qui est appliquée à l'image J. Si cette solution minimise la fonction de coût, elle est préservée, sinon elle est rejetée. Dans les deux cas, une nouvelle solution est calculée. Lorsqu'un certain nombre d'itérations est atteint, la meilleure solution intermédiaire H', est utilisée pour déterminer précisément l'ensemble des paires de points. C'est-à-dire les points pour lesquels la distance entre $p_{i,I}$ et $H'.p_{j,J}$ est inférieure à un seuil.

$$\left\| p_{i,I} - H' \cdot p_{j,J} \right\| < \varepsilon \tag{4.14}$$

Cet ensemble de paire de points plus complet est utilisé pour calculer la matrice H définitive. Cependant, dans [LOW04] l'auteur indique que l'algorithme RANSAC ne donne pas de bons résultats lorsque la proportion de faux appariements est supérieure à 50%.

La méthode RANSAC donne de meilleurs résultats que la méthode « local jet » mais elle est un peu moins rapide. La difficulté de cette approche est qu'il faut réaliser suffisamment de tirage aléatoire de sous ensemble pour augmenter la probabilité de réaliser un bon appariement. Elle reste cependant compatible avec le temps réel que nous cherchons.

4.5.2.5 Méthode KLT

Les différentes méthodes éparses que nous avons présentées jusqu'à présent sont basées sur la mise en correspondance de deux nuages de points d'intérêts extraits des deux images à recaler. L'algorithme KLT repose sur un principe un peu différent. Une première étape consiste à extraire des points d'intérêt dans la première image I. La seconde étape consiste à suivre ces points dans la seconde image. Ce type d'approche n'est généralement possible que lorsque le déplacement de la caméra entre les deux images est faible. De plus, la plupart des solutions basées sur ce modèle font l'hypothèse que le mouvement de la caméra est continu et dérivable et qu'il peut être estimé par une transformation simple. La méthode KLT est un algorithme très répandu, surtout dans le domaine de la robotique. Il offre l'avantage d'être assez simple à mettre en œuvre, il est rapide et donne d'assez bons résultats. Les premiers travaux sur cet algorithme ont été initiés par Lucas et Kanade en 1981 dans [LUC81]. Ils proposent une solution qui consiste à suivre le déplacement en translation de région d'intérêt. Les auteurs utilisent pour cela une minimisation de premier ordre de type Gauss-Newton. Cette première version de l'algorithme soufre de plusieurs limitations. La première est que le mouvement de la caméra est approximé par une simple translation. L'algorithme n'est alors pas robuste aux rotations et aux changements d'échelles. Le deuxième défaut est que la luminosité est supposée constante entre les prises de vue. Pour palier la première limitation, Shi et Tomasi [SHI94] généralisent cette approche aux cas des transformations affines. Dans [SOA01], Soato et al. utilisent le même modèle de transformation affine décrit dans [SHI94] et proposent de prendre en compte le changement de contraste et de luminosité.

Une implémentation de cet algorithme est proposée dans la librairie OpenCv. C'est cette implémentation que nous avons testée. L'avantage est que le code est particulièrement bien optimisé. Les essais que nous avons effectués donnent de très bons résultats pour un temps de calcul de l'ordre de 8ms.

4.5.3. Conclusion

Les méthodes denses donnent de bon résultats, mais le principal inconvénient de ces méthodes est que si elles ne sont pas correctement initialisées, elles risquent de ne pas tendre vers la solution optimale. Même si les méthodes stochastiques permettent de contourner cette difficulté, le nombre d'itérations n'est pas constant ce qui rend aléatoire le temps de calcul. Parmi les méthodes denses testées, la méthode du simplexe et à la fois la plus rapide et celle qui donne globalement les meilleurs résultats.

Les méthodes éparses sont dans l'ensemble beaucoup plus rapides que les méthodes denses mais elles donnent des résultats légèrement moins bons. Dans [BAR03-2], les auteurs proposent une solution de recalage où ils utilisent une première méthode éparse pour s'approcher de la solution puis une méthode dense pour affiner le résultat.

4.6. Notre approche

4.6.1. Présentation

Nous venons de présenter plusieurs méthodes de recalage que nous avons évaluées. En anticipant sur les résultats de cette évaluation que nous allons présenter dans le paragraphe suivant, nous allons rejeter plusieurs de ces méthodes. Soit parce que la précision du recalage n'est pas suffisante, soit parce que le temps de calcul est beaucoup trop long. Nous verrons que deux approches vont se détacher du lot : la méthode du simplexe et la méthode KLT. La méthode du simplexe est globalement la plus robuste. Cependant, son temps de calcul est un peu trop long (de l'ordre de 300ms). La méthode KLT est beaucoup plus rapide puisqu'elle s'exécute en 8ms environ et elle est robuste sur la plupart des images de notre corpus de test. Par contre, les performances atteintes sont moins bonnes lorsqu'il y a des objets en mouvement dans la scène. Nous cherchons donc à développer une méthode qui allie la robustesse de la méthode du simplexe à la rapidité de la méthode KLT. Dans la méthode du simplexe présentée en annexe, les étapes les plus longues en termes de temps de calcul sont le calcul de la projection de l'image J en fonction des paramètres du sommet du simplexe et le calcul de la mesure de similarité sur l'ensemble des points de la zone de recouvrement. Pour gagner en temps de calcul nous devons utiliser une mesure de similarité qui porte uniquement sur quelques points de l'image.

4.6.2. Somme des distances Euclidiennes avec le plus proche voisin

Nous proposons donc une mesure de similarité, inspirée de la distance de Hausdorff, permettant d'estimer la qualité de l'homographie sans l'évaluation préalable des paires de point. Prenons deux nuages de points P_I et P_J dans la zone de recouvrement de deux images I et J. Les points sont déterminés en fonction des données images à partir du détecteur de Harris [HAR88]. Nous faisons alors l'hypothèse que dans chacun de ces deux nuages, les points sont statistiquement indépendants. La mesure de similarité que nous proposons est la somme des distances entre la projection suivant H de chaque point de l'image J avec le point le plus proche, au sens de la distance Euclidienne, dans la première image. Cette mesure est donnée par :

$$S = \frac{1}{N_I} \sum_i \min\left(d\left(p_{i,I} - H'_2 \cdot p_{j,J} \right)^2 \right) \tag{4.15}$$

où N_I est le nombre de points du nuage P_I.

(a) Image I *(b) Image J* *(c) Appariement des points*

Figure 103 : Exemple de l'application de la mesure de similarité éparse sur deux nuages de points

L'exemple présenté ci-dessus (Figure 103) est l'application de la mesure sur un cas réel. Sur l'image de la Figure 103c, les points gris correspondent aux points de l'image I, les points

noirs correspondent à la projection des points de J dans I à partir de la transformation H entre les deux images. Les traits gris représentent les liens réalisés entre un point de I et son plus proche voisin dans H(J) au sens de la distance Euclidienne.

Implicitement, nous réalisons quand même l'appariement des points de J avec des points de I mais uniquement sur la base de leur position dans les images. Si les deux nuages de points sont constitués de façon purement aléatoire, cette mesure de distance n'a aucune justification. Par contre, si l'extraction de ces points est basée sur un certain nombre de critères commun aux deux images, cette mesure à un sens. Nous pouvons également noté que nous ne limitons pas l'appariement d'un point de J avec un seul point de I. Sur la Figure 103, nous avons entouré en bleu un exemple d'appariement de plusieurs points de I avec un point de J. Nous pourrions améliorer la mesure en supprimant ces liaisons multiples. Les résultats expérimentaux que nous obtenons sont suffisamment bons sans avoir recourt à cette limitation.

Nous avons appliqué à cette mesure le protocole de test des mesures de similarité que nous avons présenté au début de ce chapitre. Le résultat du protocole de test 1 est donnée ci dessous.

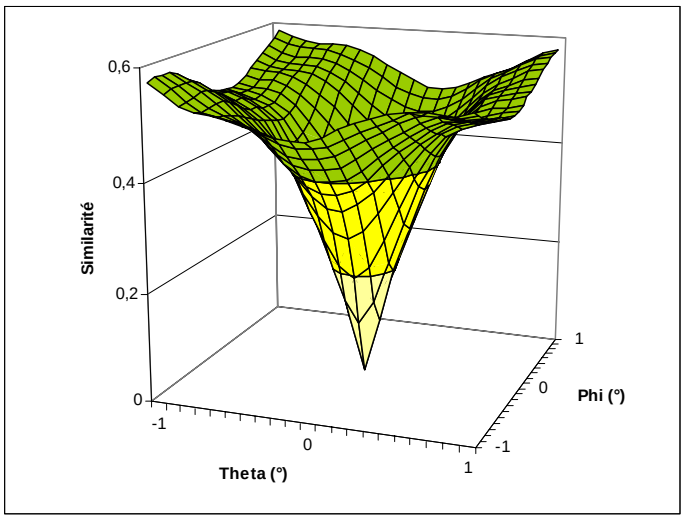


Figure 104 : Evolution de la mesure en fonction du décalage θ et φ.

Sur la Figure 104, nous pouvons remarquer que le profil de l'évolution de la mesure en fonction du décalage θ et φ répond pleinement à nos attentes. Le minimum est franc et son influence est sensible sur une grande partie de l'espace de recherche. L'autre intérêt de cette méthode est qu'elle ne dépend pas directement du changement de luminosité. C'est la méthode d'extraction des points qui doit être robuste aux changements. En revanche, la qualité de la mesure dépend du nombre de points détectés dans I qui sont également détectés dans J. En effet, si un point de I n'est pas détecté dans J, il y aura quand même un point de J qui minimise la distance Euclidienne entre les deux points. Afin de d'évaluer la robustesse de notre mesure au rapport entre le nombre de points correctement appariés et le nombre total de point, nous avons adapté le protocole de test numéro 2. Le graphique suivant (Figure 105) représente donc l'évolution de la mesure en fonction du décalage θ en fixant $\varphi=0$ pour plusieurs valeurs du ratio entre le nombre de points correctement appariés et le nombre total de points.

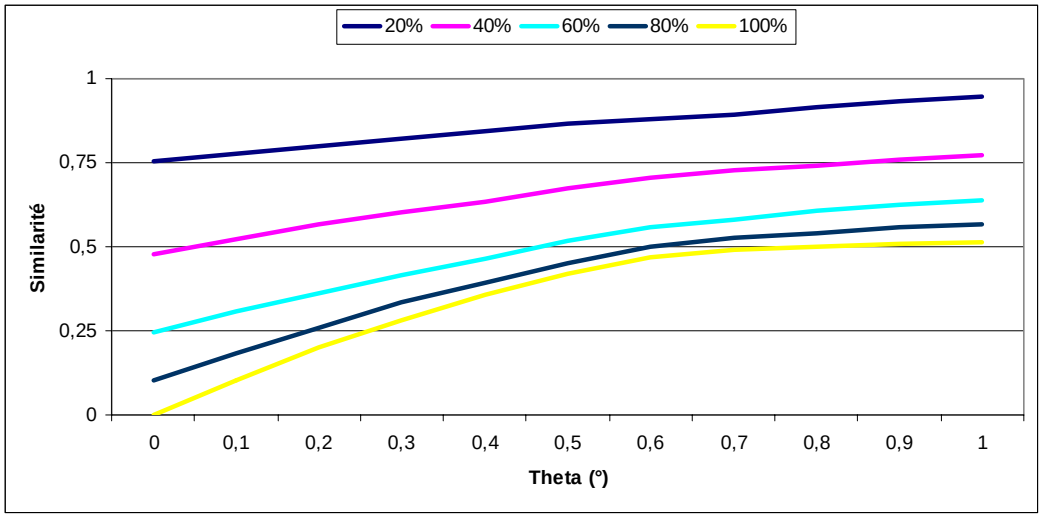


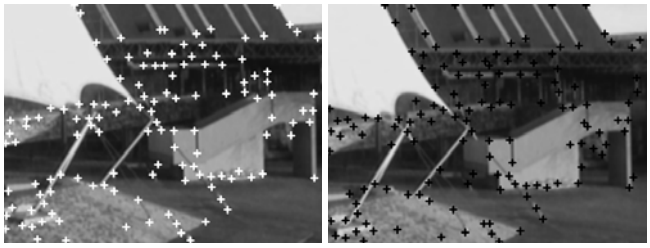
Figure 105 : Evolution de la mesure en fonction du décalage θ et à φ=0 pour plusieurs ratio entre le nombre de points correctement appariés et le nombre total de points

Nous pouvons remarquer que de 100% à 60% de bons appariements, la mesure donne de bons résultats. Pour 20% et 40% de bons appariements, la mesure évolue toujours dans le bon sens, par contre la pente est moins forte ce qui risque de ralentir la progression du simplexe. De même, si nous devons définir un critère d'arrêt sur un seuil de la valeur de similarité, le minimum atteint est très dépendant du nombre de bon appariement. Avec 40% de bons appariements, le minimum atteint est équivalent au maximum atteint, dans l'espace de recherche, par les courbes 60%, 80% et 100%. Dans le cas de l'algorithme du simplexe, ceci n'est pas réellement un handicap puisque nous pouvons définir un critère d'arrêt sur la taille du simplexe.

4.6.3. Algorithme de recalage

Notre algorithme de recalage est un algorithme itératif basé sur la recherche de la minimisation d'une mesure de similarité. La progression dans l'espace de recherche est basée sur la méthode du simplexe telle que nous la décrivons en annexe à la différence que la valeur associée à chaque sommet n'est pas calculée à partir de la mesure NCC mais à partir de la mesure que nous venons de présenter.

Avant d'initialiser les 4 sommets du simplexe, nous devons extraire les points d'intérêts dans les deux images de façon à constituer nos deux nuages de points. Le but de notre approche est de déterminer les paramètres f_J, φ_J et θ_J ou $\Delta\theta_{I,J}$ de la matrice de transformation mathématique permettant de faire coïncider ces deux nuages de points. Les paramètres f_J, φ_J et θ_J ou $\Delta\theta_{I,J}$ sont tels que décrit dans le paragraphe 4.3 « l'imitation de l'espace de recherche ».



(a) *(b)*

Figure 106 : Détection des points d'intérêts dans l'image I (a) et l'image J (b)

Une fois les deux nuages de points constitués, nous déterminons les paires de points possibles comme décrit dans le paragraphe 4.5.2.2 « liste des candidats possibles ». L'importance de cette phase est double. Elle permet, d'une part, de limiter la dérive de la mesure de similarité lorsqu'un point d'intérêt de I n'a pas été détecté dans J et qu'il n'y a pas d'autre point dans le voisinage. Elle permet, d'autre part, de limiter la complexité de la mesure de similarité. En effet, telle que nous l'avons décrite, la complexité de la mesure est en O(n.m) où n est le nombre de points d'intérêts du nuage de I et m le nombre de points d'intérêt du nuage de J. Avec cette phase, la complexité reste en O(n.m) avec la même constante de complexité O sauf que dans le premier cas n et m sont de l'ordre de 100 alors que dans le deuxième cas, n est toujours de l'ordre de 100 mais m est de l'ordre de 5, ce qui divise par 20 le nombre de calculs de distance entre les points.

Les paramètres de la transformation recherchée correspondant aux sommets du simplexe sont initialisés de façon aléatoire à l'intérieur de l'espace de recherche. Pour chaque sommet, nous calculons la matrice de transformation que nous appliquons aux coordonnées des points d'intérêts du nuage de J de façon à évaluer la mesure de similarité. Par rapport à la méthode classique, le gain de temps est ici considérable. Dans notre cas, la transformation s'applique sur un nombre restreint de points (de l'ordre de 100) alors que dans la méthode classique elle s'applique sur tous les points. De plus, nous n'avons pas à calculer d'interpolation bilinéaire et nous pouvons garder le résultat de la transformation au format réel.

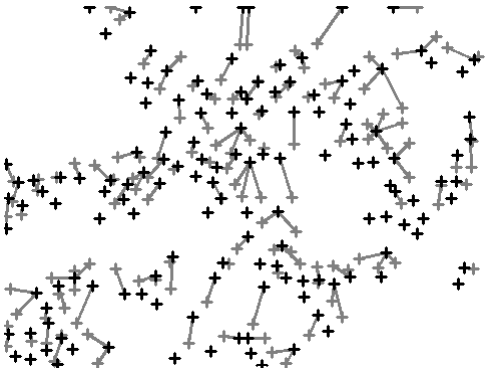


Figure 107 : Résultats du calcul de la similarité sur un sommet initialisé aléatoirement

L'image ci-dessus (Figure 107) est une illustration du résultat de la mesure de similarité sur des sommets après initialisation. Les points gris correspondent aux points de l'image I, les points noirs correspondent à la projection des points de J dans I à partir de la transformation

H entre les deux images. Les traits gris représentent les liens réalisés entre un point de I et son plus proche voisin dans H(J) au sens de la distance Euclidienne.

Une fois l'initialisation réalisée nous appliquons l'algorithme classique du simplexe permettant de faire évoluer les paramètres de la transformation dans l'espace de recherche. Le critère d'arrêt de l'algorithme est fixé sur la taille minimum du simplexe et sur un nombre maximum d'itérations. La figure suivante (Figure 108) représente les paires de points réalisés à l'arrêt de l'algorithme.

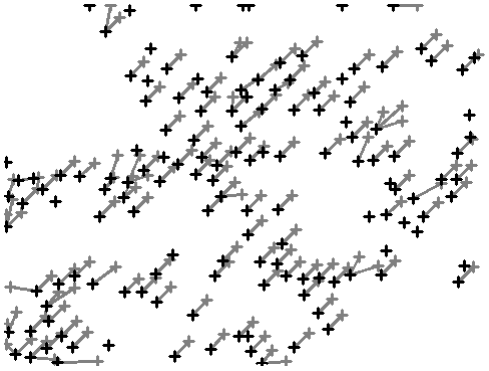


Figure 108 : Résultats du calcul de la similarité à l'arrêt de l'algorithme du simplexe

Sur cette figure, nous pouvons constater qu'une majorité des points ont été correctement appariés. Par contre, il subsiste quelques erreurs. Nous terminons notre algorithme par un filtrage des « outliers » et par la mise à jour de la matrice de transformation.

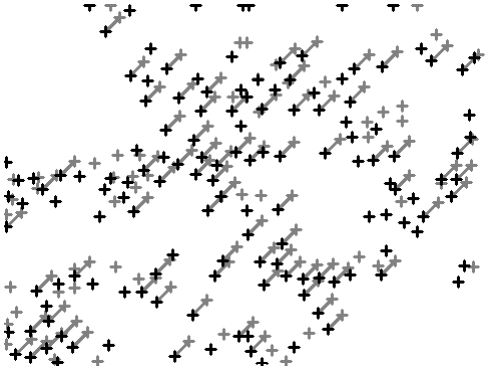


Figure 109 : Résultats du calcul de la similarité après filtrage des « outliers »

Cette approche s'est révélée très rapide. En effet, une fois les points d'intérêt extraits, il n'y a pas de calcul de caractéristiques sur les points et donc plus de traitement sur l'image elle-même. Il ne reste qu'à déterminer les matrices de transformations à partir des estimations f_J, θ_J, φ_J et à les appliquer à quelques points (< 100) de l'image J afin de calculer leur projection dans l'image I. Nous avons ainsi obtenu un temps de calcul moyen inférieur à 5ms ce qui place notre approche en tête des méthodes les plus rapides. Le résultat du recalage est excellent mais reste légèrement inférieur à la méthode du simplexe dense, par contre dans la plupart des cas, il est supérieur à la méthode KLT.

4.7. Evaluation des méthodes de recalage

4.7.1. Protocole de test

Afin de comparer les différentes méthodes de recalage que nous venons de présenter, nous avons mis en place un protocole de test comparable à celui utilisé pour tester les mesures de similarité. Sur une image réelle, c'est à dire acquise par notre caméra, nous appliquons différentes transformations homographiques autour des paramètres de prise de vue de l'image et nous recherchons les paramètres de cette transformation en appliquant différentes méthodes de recalage. Pour mesurer la qualité de la mise en correspondance, nous utilisons une adaptation de la mesure EDHE (Euclidean Distance for Homography Estimation) proposée dans [AGA05] que nous avons normalisée et que nous appelons NEDHE. La mesure NEDHE est une fonction d'erreur calculée pour l'image J après projection des points dans l'image I à partir de la matrice de la transformation homographique H de J dans I. C'est une mesure de distance Euclidienne entre les points $p_{i,I}$ et les points correspondants $p_{j,J}$ après transformation c'est à dire les points d'expression $H.p_{j,J}$.

$$NEDHE = \frac{1}{N}\sum_{i=1}^{N} d(p_{i,I}, H \cdot p_{j,J}) \tag{4.16}$$

Ainsi modifiée, cette mesure normalisée correspond à l'écart moyen exprimé en pixel.

Dans les différents tableaux de résultats que nous présentons, la distance d correspond à la mesure NEDHE entre l'image de départ et l'image transformée. Nous avons vu précédemment que l'homographie entre deux images, dans le cadre de notre application, se calcule à partir de 5 paramètres : f_0, φ_0, f_1, φ_1, $\Delta\theta_{0,1} = (\theta_1 - \theta_0)$. Si les paramètres de prise de vue de l'une des deux images sont connus alors, il reste 3 paramètres à estimer. Cette mesure permet de modéliser « l'éloignement » entre les deux images à partir d'une seule mesure. Ensuite, l'erreur e, correspond à la mesure NEDHE entre les deux images après recalage. Pour calculer la mesure NEDHE, nous utilisons N = 25 points répartis uniformément dans l'image. Les coordonnées de chaque point sont comprises, sur l'axe horizontal comme sur l'axe vertical, entre –200 et 200 par pas de 100.

Nous avons effectué 3 séries de tests de façon à estimer la robustesse des méthodes de recalage aux différents cas réels que nous rencontrons dans nos applications. Dans la première série de test, nous fixons f_0 = 800up, $\varphi_0 = 0°$ et $\theta_0 = 0°$. La distance focale f_0 est fixée à une valeur de 800up correspondant, sur notre modèle de caméra, à une distance focale d'environ 4mm. Nous appliquons ensuite notre protocole de test sur 4 images différentes.

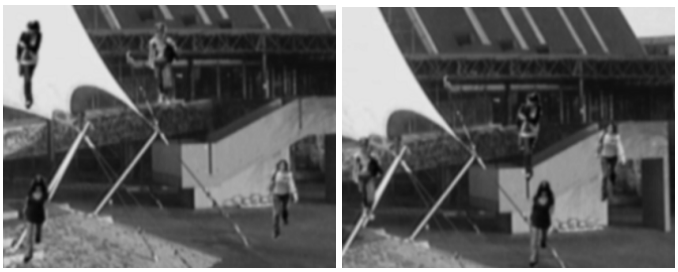
(a) Forum *(b) Entrée* *(c) Arbres* *(d)Pavés*

Figure 110 : Images de notre premier protocole de test

Ces 4 images ne représentent pas un panel exhaustif des situations rencontrées. Cependant, elles permettent de tester la robustesse dans 4 situations bien différentes. L'image « Forum » est un cas assez simple que l'on retrouve régulièrement en zone urbaine : peu de végétation et des structures géométriques simple. L'image « Entrée » est un cas plus compliqué : beaucoup de gradient dans certaines zones et peu dans d'autres ainsi que la présence d'ombre. Les deux dernières images « Arbres » et « Pavés » sont des cas d'école. Dans la première, il y a beaucoup de gradient mais pas de structure géométrique simple. Dans la seconde, il y a peu de gradient mais beaucoup de motifs répétitifs. Une synthèse des résultats obtenus est donnée dans le tableau 5.1.

La deuxième série de test nous permet de vérifier la robustesse des méthodes en fonction de l'angle φ_0 de l'image de départ. En effet, si φ_0 est proche de l'équateur, un décalage $\Delta\theta_{0,1}$ correspond approximativement à une translation entre les deux images alors que si φ_0 est proche des pôles, un décalage $\Delta\theta_{0,1}$ correspond approximativement à une rotation. Dans ce deuxième test nous avons réalisé une série de mesures sur l'image « Forum » uniquement et pour 4 valeurs de φ_0 : 0°, 45°, 66° et 80°.

Dans la troisième série de test, nous vérifions la robustesse des méthodes à la présence d'objets en mouvement dans la scène. En effet, si dans la scène il y a des objets en mouvement, entre deux prises de vue, ces objets vont se déplacer. Ils vont donc entraîner des modifications dans la scène. Comme nous évoluons essentiellement dans le monde urbain, nous allons placer aléatoirement des personnages dans l'image de départ et nous allons leur calculer un déplacement également aléatoire dans l'image transformée. Nous faisons le test pour 4 niveaux de personnages : 0, 2, 4 et 8 sur l'image « Forum ». Chaque personnage occupe environ 1% de l'image.



(a) *(b)*

Figure 111 :Exemple de placement des avatars dans les images
(a) Image de départ (f_0 = 800up, θ_0= 0°, φ_0= 0°) avec 4 personnages
(b) Image transformée (f_1 = 800up, θ_1= 2°, φ_1= 2°) avec déplacement des 4 personnages

Pour chaque série de test, nous fixons f_0, φ_0 et θ_0 puis nous faisons varier φ_1 et θ_1 de $\pm$ 2° par rapport respectivement à φ_0 et θ_0 par pas de 0.4° pour $f_1 = 0.8 \times f_0$, $f_1 = f_0$ et $f_1 = 1.2 \times f_0$. Notre champ d'investigation n'est pas choisi arbitrairement mais correspond à une réalité physique liée à notre matériel de test ainsi qu'aux applications visées. Ces applications peuvent se classer en deux catégories. Dans la première catégorie, les images sont acquises lorsque la caméra est à l'arrêt et après stabilisation. Dans ce cas, l'erreur de positionnement donnée par notre matériel est inférieure à 1°. Pour la distance focale, la précision est un peu meilleure mais elle nécessite une phase d'étalonnage qui elle peut manquer de précision. Dans la deuxième catégorie d'application, les images sont acquises pendant le mouvement de la caméra. L'image capturée étant composée de deux trames entrelacées acquises à deux instants différents, un mouvement trop rapide de la caméra entraînera un effet de « peigne »

que l'on rencontre habituellement lorsque les caméras sont en mouvement ou lorsque les objets en mouvement dans la scène ont un déplacement trop rapide. Pour éviter cet effet de « peigne », le déplacement en rotation de notre caméra ne doit pas excéder 1° entre deux acquisitions à 25 images par seconde. Dans les deux cas, la limite est de l'ordre de 1 degré. Afin de ne pas entraîner d'effet de bord aux voisinages de cette limite, nous fixons l'espace de recherche de chaque méthode à ± 2° par rapport à la position de l'image d'origine sans tenir compte de connaissances *a priori*.

4.7.2. Résultats

Nous présentons les résultats sous forme de 3 tableaux, c'est à dire 1 tableau pour chaque test. Le nombre d'essais que nous avons réalisé étant relativement important, nous avons défini trois domaines de « déplacement » de la caméra : $0 \leq d1 < 5$, $5 \leq d2 < 20$ et $20 \leq d3 < 40$. Ces déplacement correspondent à la mesure NEDHE appliquée sur la matrice de transformation avant recalage et s'exprime en pixel.

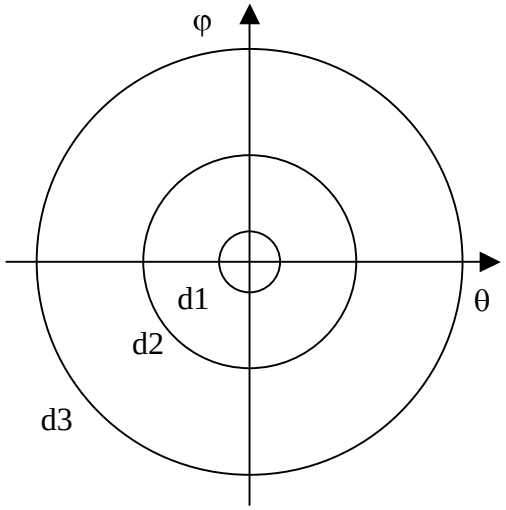


Figure 112 : Représentation des domaines de déplacement de la caméra dans l'espace (θ, φ).

Le premier correspond à des déplacements de la caméra très faibles. Le deuxième correspond à des déplacements que nous rencontrons généralement dans nos applications. Le troisième correspond à des déplacements supérieurs à ce que nous envisageons dans nos applications. Pour chaque domaine nous indiquons la moyenne des erreurs constatées après recalage.

4.7.2.1 Test image

Le premier tableau de résultats que nous présentons correspond au test « image ».

Test image	Forum			Entrée			Arbre			Pavé		
	d1	d2	d3	d1	d2	d3	d1	d2	d3	d1	d2	d3
Szeliski	0.07	6.11	11.2	2.62	13.0	24.9	1.82	12.4	27.0	2.04	9.88	24.2
simplexe dense	0.02	0.02	0.62	0.06	0.20	0.20	0.15	0.42	0.03	0.63	0.04	0.55
génétique	0.10	0.24	0.32	0.08	0.23	0.29	0.06	0.16	0.24	0.11	0.36	0.47
Recuit simulé	0.60	0.69	0.92	0.52	0.67	0.83	0.30	0.57	0.64	0.56	0.75	0.99
SIFT	0.38	0.67	0.61	0.11	0.66	1.07	0.02	0.03	0.02	1.02	1.08	22.9
Ransac	0.70	1.03	17.3	1.26	87.5	175.	0.45	4.64	42.0	0.50	0.93	86.4
Local jet	1.92	3.98	74.1	1.34	9.10	17.9	0.52	2.83	26.7	4.49	28.0	19.7
KLT	0.33	1.17	8.35	0.03	0.20	2.18	0.02	0.53	2.45	2.54	16.5	36.5
Notre Approche (classement)	0.12 (4)	0.58 (3)	1.05 (5)	0.11 (4)	0.42 (4)	2.71 (6)	0.08 (4)	0.41 (3)	1.8 (5)	0.09 (1)	0.09 (2)	3.20 (4)

Tableau 7 : Résultats des tests de la robustesse des méthodes de recalage sur différentes images

Dans ce tableau, nous pouvons remarquer que la méthode du simplexe dense donne des résultats stables sur l'ensemble des images de notre corpus de test. Il en est de même pour la méthode du recuit simulé bien que les résultats soient un peu moins bons. En règle générale les méthodes denses que nous avons testées donnes des résultats équivalents quel que soit le type d'image. Les méthodes éparses par contre sont beaucoup plus sensibles. La méthode RANSAC fonctionne très bien pour l'image Forum mais donne des résultats complètement aberrant sur l'image Entrée. Comme il y a beaucoup de fort gradient sur le coté gauche de cette image, le détecteur de Harris extrait beaucoup de point dans cette zone. Le risque de faire de mauvais appariements est donc important. Nous arrivons à améliorer les performances de cet algorithme en augmentant le nombre de tirages. Mais ceci au détriment du temps de calcul. Le classement de notre approche par rapport aux autres méthodes présentées ne semble pas très flatteur. Cependant, nous pouvons remarquer que les résultats obtenus sont toujours relativement proches des meilleures solutions et en aucun cas aberrants.

4.7.2.2 Test φ_0

Le tableau suivant correspond au test φ_0.

Test φ_0	$\varphi_0 = 0°$			$\varphi_0 = 45°$			$\varphi_0 = 66°$			$\varphi_0 = 80°$		
	d1	d2	d3	d1	d2	d3	d1	d2	d3	d1	d2	d3
Szeliski	0.07	5.62	13.9	0.07	6.11	11.2	0.08	4.42	11.8	0.08	1.51	12.0
simplexe dense	0.03	0.16	0.13	0.02	0.39	0.52	0.01	0.88	0.02	0.07	0.25	0.28
génétique	0.08	0.17	0.26	0.11	0.22	0.26	0.15	0.20	0.37	0.15	0.21	0.21
Recuit simulé	0.53	0.68	1.07	0.56	0.65	0.89	0.42	0.60	0.71	0.47	0.76	0.92
SIFT	0.38	0.70	1.00	0.38	0.66	0.61	0.61	0.82	0.49	0.46	0.66	0.58
Ransac	0.80	1.03	18.3	0.69	1.07	59.3	0.73	4.46	58.0	0.72	0.91	1.09
Local jet	2.86	7.08	29.4	1.92	3.98	74.1	2.15	4.66	8.82	1.07	1.01	4.16
KLT	0.03	0.90	6.51	0.33	1.17	8.35	0.42	0.70	5.45	0.23	1.70	6.53
Notre Approche (classement)	0.20 (5)	0.17 (3)	1.07 (5)	0.12 (4)	0.09 (1)	0.55 (3)	0.11 (3)	0.09 (1)	0.19 (2)	0.13 (3)	0.12 (1)	0.20 (1)

Tableau 8 : Résultats des tests de l'influence de φ_0 sur les méthode de recalages

Nous pouvons remarquer que toutes les approches présentées sont robustes à l'angle de tangage.

4.7.2.3 Test objets dans la scène

Le dernier test correspond à la robustesse des méthodes aux déplacements d'objets dans la scène.

Test avatar	0			2			4			8		
	d1	d2	d3	d1	d2	d3	d1	d2	d3	d1	d2	d3
Szeliski	0.07	6.11	11.2	0.75	9.11	16.1	3.20	8.89	17.5	4.39	16.8	26.2
simplexe dense	0.02	0.04	0.97	0.08	0.06	0.18	0.12	0.46	0.47	0.19	0.19	0.39
génétique	0.05	0.15	0.27	0.08	0.21	0.28	0.13	0.20	0.34	0.20	0.28	0.41
Recuit simulé	0.42	0.65	0.98	0.44	0.69	1.16	0.61	0.55	1.04	0.73	0.68	0.89
SIFT	0.38	0.67	0.61	14.1	111	1357	51.2	41.9	73	1902	3127	8562
Ransac	0.90	1.81	23.5	0.82	6.99	58.1	1.21	4.67	440	3.27	205	420
Local jet	1.92	3.98	74.1	1.30	11.8	53.8	3.33	8.32	36.3	3.85	34.4	32.6
KLT	0.33	1.17	8.35	0.36	2.98	7.99	0.37	1.56	9.36	3.58	0.91	15.1
Notre Approche (classement)	0.12 (4)	0.09 (2)	0.85 (3)	0.17 (3)	0.11 (2)	1.61 (4)	0.15 (3)	0.13 (1)	0.60 (3)	0.10 (1)	0.25 (2)	4.03 (4)

Tableau 9 : Résultats des tests de robustesse des méthodes de recalage à la présence d'objets en mouvement dans la scène.

Comme pour le type d'image, les méthodes denses sont moins sensibles que les méthodes éparses. Ceci va à l'encontre de ce que nous avions lu dans la littérature. Cependant, notre test est un peu particulier dans la mesure où la surface maximale occupée par les objets en mouvement n'excède pas 10% de la surface de l'image dans le cas le plus défavorable. Les méthodes éparses résistent bien lorsque le nombre d'avatars est inférieur ou égal à 4. Par contre, la méthode SIFT n'est absolument pas robuste aux objets en mouvement dans la scène. Avec seulement deux avatars, la méthode SIFT n'est pas exploitable. Pour le test de 8 avatars, les résultats des méthodes éparses sont moins convaincants, sauf pour notre approche qui donne des résultats comparables quel que soit le nombre d'avatar. Par rapport aux autres approches éparses, la méthode que nous proposons se focalise sur les éléments fixes de la scène. En effet la mesure de similarité que nous utilisons cherche systématiquement à minimiser la distance entre les points d'intérêt et leur plus proche voisin dans l'image transformée. Lorsqu'un point d'intérêt est détecté sur un personnage dans l'image I, la distance avec le point sur le même personnage en mouvement dans l'image transformé est différente de celle généralement observée sur les points fixes de la scène. Le point est alors considéré comme un « outlier » et il est rejeté. Au final, il y simplement un peu moins de points et donc moins d'équations pour la résolution aux moindres carrés .

4.7.2.4 Temps de calcul

Pour terminer, nous présentons le tableau correspondant au temps de calcul moyen des différentes méthodes :

Méthodes	temps (ms)
Szeliski	1405
Simplexe	275
Génétique	764

Recuit simulé	305
Ransac	22
Local jet	14
KLT	5,7
Notre approche	3,8

Tableau 10 : Temps de présence moyen des méthodes de recalage

Comme nous l'avons annoncé lors de la présentation des différentes mesures, le temps de calcul des méthodes éparses est bien inférieur à celui des méthodes denses. Notre approche est la plus rapide.

4.7.2.5 Test des mesures de similarité

Dans le chapitre sur les mesures de similarité, nous avons fait un certain nombre d'hypothèses sur le comportement de ces mesures vis à vis de notre problématique. Afin de vérifier que nos hypothèses sont fondées, nous avons comparé les résultats du recalage en fonction de différentes mesures de similarité. Nous utilisons pour cela la méthode du simplexe qui est, parmi les méthodes denses, la méthode qui nous a donnée les meilleurs résultats. Dans ce cas, les différentes mesures de similarités sont utilisées pour évaluer la qualité du simplexe et permettre à celui ci de progresser dans l'espace de recherche. Nous appliquons le protocole de test des méthodes de recalage pour $\varphi_0 = 0$ sur les images du test de luminosité (Figure 83). Nous utilisons l'image « Gain 100 » comme image de référence et nous réalisons le recalage avec les images « Gain 50 », « Gain 100 » et « Gain 150 ». Le test sur l'image « Gain 100 » nous permet dans un premier temps de tester les mesures de similarité en l'absence de changement de gain. Les deux autres tests, nous permettent de tester l'influence du changement de gain sur le comportement des mesures de similarité. Le tableau suivant est une synthèse des mesures effectuées :

	Gain 50			Gain 100			Gain 150		
	d1	d2	d3	d1	d2	d3	d1	d2	d3
SAD	5,96	8,00	11,93	0,04	0,03	1,35	2,91	4,84	10,43
SSD	1,85	1,99	5,75	0,21	0,14	0,13	1,28	1,10	1,36
ZSSD	10,71	16,05	30,37	11,30	17,43	29,72	9,58	14,77	28,68
ZNSSD	3,45	13,60	29,27	5,41	13,64	28,21	3,67	13,26	29,33
NCC	0,17	0,23	1,34	0,14	0,07	0,15	0,15	0,15	1,01
ZNCC	0,14	0,16	0,82	0,02	0,07	0,10	0,15	0,12	0,88
IM	41,21	41,37	48,19	49,35	44,74	49,65	42,92	47,17	50,35

Tableau 11 : Résultats des tests de robustesse des mesures de similarité dans le cas de la méthode du simplexe

Sur le test « gain 100 », nous pouvons remarquer que les méthodes de recalage se comportent comme nous pouvions nous y attendre. Les profils des mesures SAD, SSD, NCC et ZNCC permettent un recalage correct sur l'ensemble de l'espace de recherche. Les mesures ZSSD, ZNSSD et IM ne sont clairement pas adaptées à notre problématique. En fonction du changement de luminosité, les deux mesures de corrélations sont nettement plus robustes, avec un avantage pour la mesure ZNCC.

4.8. Conclusion

Il existe encore bien d'autres méthodes que nous n'avons pas évoquées dans ce chapitre dont les méthodes basées sur le « Block Matching ». Le principe de ces méthodes consiste à découper la première image en blocs d'une certaine taille et de chercher les blocs correspondants dans la seconde image. C'est le principe de base de plusieurs algorithmes de compression. Plutôt que de découper l'image en blocs de façon arbitraire, une autre solution consiste à détecter des points d'intérêt et à extraire des blocs centrés sur les points d'intérêt. L'idée étant de limiter le nombre de blocs sur les régions de l'image contenant le plus d'information. La mise en correspondance des blocs se fait traditionnellement en calculant une mesure de similarité de type SAD ou ZNSSD. Afin d'accélérer le temps de calcul, [LAM03] propose une recherche en diamant. Certains auteurs [FUM05] utilisent des techniques de type « block matching » pour effectuer le recalage d'images. Le principe de mise en correspondance de bloc n'est pas très robuste lorsqu'il s'agit d'estimer des rotations.

Une autre approche que nous n'avons pas étudiée est la relaxation : Le principe de cette approche éparse est de réaliser un premier appariement entre les points d'intérêts des deux images. Le lien entre deux points d'intérêts à un coût qui peut être fonction de certains critères (distance relative, similarité …). A chaque itération, les liens les plus forts sont préservés et d'autre combinaison sont réalisées.

Nous n'avons pas non plus évoqué les possibilités d'optimisations de façon à accélérer les calculs. L'optimisation du code est une étape important lorsque l'on travail en temps réel. D'une manière générale, nous cherchons toujours à gagner du temps de calcul, mais dans le cas du temps réel, c'est une donnée essentielle qui nécessite bien souvent de faire des choix au détriment de la précision. Nous n'allons pas parler ici des détails d'implémentations permettant de gagner quelques millisecondes mais évoquer rapidement l'aspect multi-résolution. L'approche pyramidale est un grand classique en traitement d'image. Il permet de d'approcher la solution à partir d'une résolution très basse et d'affiner les résultats en augmentant la résolution. Cette approche est particulièrement efficace dans le cas des méthodes denses. Elle est également utilisée dans [BOU99] afin d'améliorer les performances de l'algorithme KLT.

Mosaïque d’images multi-résolution et applications

Chapitre 5 :Applications

5. Applications

5.1. Avant propos

Dans ce rapport de thèse, nous nous avons tout d'abord présenté un aspect purement théorique du mosaïquage. Nous avons ensuite progressé dans « l'applicatif » en prenant en compte un certain nombre de défauts liés à l'acquisition. Nous avons à présent à notre disposition, une collection d'outils nous permettant de corriger l'image et de déterminer avec suffisamment de précision les paramètres de la prise de vue. Nous allons utiliser ces outils pour répondre à des besoins exprimés par la société Foxstream. La société Foxstream est spécialisée dans l'analyse de la vidéo et propose entre autres, des solutions de vidéo protection. Dans le cadre de cette activité, ces besoins sont doubles. Le premier est de proposer une solution permettant de repérer tout objet en mouvement dans une vaste étendue tout en limitant le nombre de caméras. Le deuxième est de fournir une solution de détection et de suivi d'objet en mouvement avec une caméra PTZ.

Après un état de l'art des solutions décrites dans la littérature, nous proposons une architecture complète permettant la détection des objets en mouvement. La solution que nous proposons intègre la détection ainsi que la calibration automatique de l'équipement. Nous proposons ensuite un algorithme permettant la détection des objets en mouvement sur une caméra PTZ elle-même en mouvement. L'originalité de notre approche est qu'elle permet une segmentation fine des objets en mouvement permettant leur classification.

5.2. Etat de l'art

Dans une activité de vidéo protection, l'usage des caméras PTZ est une solution intéressante puisqu'elle permet une vision très élargie du périmètre à surveiller avec une seule caméra. Toutefois, son usage conventionnel requiert l'intervention d'un opérateur pour contrôler le mouvement de la caméra. Depuis quelques années, des solutions automatiques sont proposées [KAN03, MIT04, MIG05, BEV05, HU05, BIS06]. La plupart des approches consistent à créer un panorama complet ou partiel de la scène. Par exemple, dans [BIS06], les auteurs créent un modèle de fond à base de mélanges de gaussiennes [STA99] qu'ils projettent sur un panorama cylindrique. L'article proposé par Hu [HU05] se démarque des solutions généralement proposées. Comme dans les autres cas, les auteurs utilisent la caméra PTZ pour détecter les changements intervenus dans la scène. Mais plutôt que de créer un panorama complet qui serait coûteux en place mémoire, ils réalisent une description concise de la scène à partir d'un modèle probabiliste basé sur les mélanges de gaussiennes. Ils obtiennent ainsi un gain sur l'espace mémoire de 1 pour 10 000. Leur méthode permet de détecter des intrusions dans la scène sans pouvoir cependant préciser la position de l'objet dans l'image.

La segmentation à partir d'un panorama représentant les composantes statiques de la scène n'est pas la seule alternative pour détecter des objets. Certains algorithmes, comme celui de Viola et Jones [VIO01, BUR06, NEG07] basé sur les descripteurs de Haar et sur un apprentissage de type adaboost, permettent de repérer des objets dans une image. A l'origine, l'algorithme de Viola et Jones a été développé pour détecter des piétons ou des visages dans une image, mais leur approche peut être appliquée à d'autres types d'objets. Dans [BUR06], cet algorithme est utilisé pour détecter des têtes d'animaux. Cependant la constitution de la base d'apprentissage est relativement fastidieuse. Pour que le niveau de détection soit acceptable, il faut plusieurs centaines d'imagettes « découpées » manuellement et une phase d'apprentissage dont le calcul peut s'étendre sur plusieurs heures. De plus, certains paramètres de cet algorithme sont délicats à paramétrer. Dans [MIA08], l'auteur utilise cet algorithme pour détecter et suivre les visages avec une camera PTZ.

Dans le cadre de caméras fixes, plusieurs auteurs [HEE88, NAG96, SPI98] utilisent le calcul du flot optique. En général, les auteurs calculent le flot optique puis segmentent les régions dont l'écoulement est uniforme. L'approche de Nagai est particulièrement intéressante. Pour améliorer le suivi des objets en mouvement, il utilise un accumulateur de Hough. Dans l'hypothèse où les objets à détecter ont un mouvement uniforme, cette technique permet de filtrer le mouvement de la végétation crée par le vent. La plupart des estimations du flot optique sont réalisée à partir des travaux de Lucas-Kanade [LUC81] ou de Horn-Schunck [HOR81]. De façon plus marginale Torralba [TOR99] a développé une méthode basée sur le calcul de la FFT. Demonceaux et Kachi-Akkouche [DEM03] utilisent la décomposition en ondelettes. Ces techniques d'estimation du flot optique sont intéressantes mais s'avèrent délicates à manipuler pour extraire de l'image une personne qui marche par exemple. En effet le balancement des bras ne suit pas forcement le sens de la marche. Pour un usage avec des caméras PTZ en rotation, la difficulté supplémentaire est qu'il faut estimer le flot optique engendré par le déplacement de la caméra.

Toutefois, le principal inconvénient de l'usage d'une caméra PTZ est que lorsque la caméra est orientée dans une direction, elle ne permet pas de visualiser ce qui ce passe dans le reste de la scène. Dans le cas d'une application de détection d'intrusion, il peut être important de surveiller l'ensemble de la zone en permanence de façon à détecter de façon certaine tout événement et de ne piloter la caméra que lorsqu'un événement survient. Pour contourner cette difficulté, plusieurs auteurs [ABI03, CHE08] utilisent deux caméras. Une caméra fixe avec un objectif grand angle ou de type fish-eye ou encore une caméra associée à un miroir sphérique ou parabolique et une caméra PTZ asservie par la première. Dans [CHE08], les auteurs utilisent une caméra PTZ et une caméra omnidirectionnelle. Ils proposent deux approches pour la calibration : géométrique et homographique. De l'aveu même des auteurs, l'inconvénient du calibrage géométrique est qu'il est nécessaire de connaître la position relative des deux caméras. C'est la raison pour laquelle, ils lui préfèrent la calibration homographique. Ils utilisent une transformation polynomiale pour déterminer les angles de pilotage de la caméra PTZ en fonction des coordonnées (x, y) de l'objet en mouvement dans l'image omnidirectionnelle.

Liu et all [LIU02] ont développé un système appelé FLYSPEC à base d'une caméra dôme et de 4 caméras fixes. Leur application permet le contrôle collaboratif de la caméra PTZ par plusieurs opérateurs avec une gestion intelligente des conflits et des flux vidéos transmis. Cependant, leur système repose sur une expertise humaine de la vidéo et non sur une détection automatique.

Les applications de poursuite, bien que plus récentes, sont également très présentes dans la littérature [SHA04, LAL07, MIA08]. Nous faisons une distinction entre le suivi des objets et la poursuite. Dans le premier cas, il s'agit de retrouver dans l'image courante les objets que nous avions précédemment détectés. Dans le deuxième cas, il s'agit, en plus, de piloter la caméra de telle sorte que l'objet en mouvement reste dans le champ de la caméra. Dans [SHA04], les auteurs proposent une solution de poursuite automatique de cible basée sur un filtrage particulaire. Dans [MIA08], l'auteur propose une solution de détection et de poursuite des visages. Il utilise pour cela l'algorithme de Viola et Jones [VIO01] à travers les fonctions de la librairie *OpenCv*[7].

Dans [CLA01], les auteurs utilisent la caméra PTZ dans le cadre de l'aide à la conduite. Par rapport aux applications plus classiques de poursuite, ils augmentent la difficulté dans la mesure où la caméra est embarquée dans le véhicule. En plus des rotations de la caméra, ils doivent prendre en compte le mouvement de translation du véhicule.

5.3. Détection d'objets en mouvement dans une scène étendue

5.3.1. Principe

Comme nous l'avons vu dans l'état de l'art précédent, l'une des architectures proposées dans la littérature consiste à utiliser une caméra PTZ asservie par une caméra omnidirectionnelle. L'intérêt de cette architecture est que la caméra omnidirectionnelle visualise en une seule prise de vue l'ensemble de la scène. La faible résolution de l'image obtenue ne permet pas de réaliser l'identification des objets. C'est la raison pour laquelle nous lui associons une caméra PTZ dont le pilotage est contrôlé par la caméra omnidirectionnelle. Tant qu'il n'y a pas de mouvement détecté par la caméra omnidirectionnelle, la caméra PTZ n'est pas utilisée et est en position de repos. Dans la plupart des cas de détection automatique d'intrusion la nuit ou le week-end, il n'y a absolument aucune activité dans le périmètre à surveiller. Par contre, dès qu'il y a une intrusion dans la zone à protéger, le système doit réagir immédiatement afin de contrôler la nature de cet événement et déclencher une alerte si nécessaire. Dans ce cas, pourquoi ne pas se passer de la caméra omnidirectionnelle et n'utiliser qu'un miroir vu par la caméra PTZ en position de repos ? C'est la solution que nous avons retenue. Dans notre cas, nous avons utilisé un miroir sphérique.

Le dispositif expérimental mis en place (Figure 113) se compose d'une caméra PTZ RZ25P et d'un miroir sphérique placé à l'aplomb de la caméra. L'origine du repère est confondue avec le centre des rotations de la caméra. Le centre du miroir est placé le long de l'axe vertical.

[7] OpenCv est une bibliothèque libre de droit et spécialisée dans le traitement d'image temps réel. http://opencvlibrary.sourceforge.net/

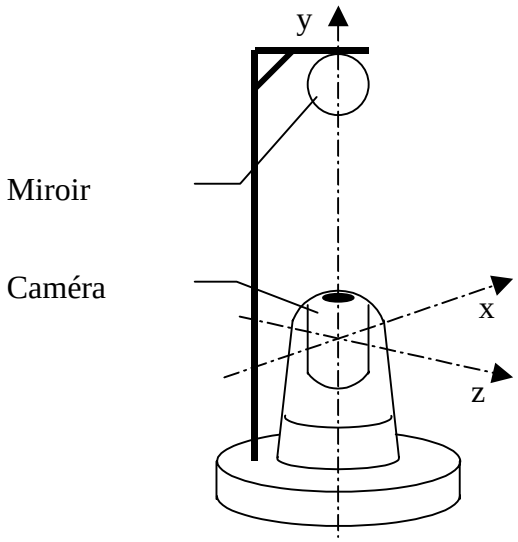


Figure 113 : Schéma de principe du dispositif Caméra/Miroir

La position de repos de la caméra correspond à $\theta = 0$ et $\varphi = 90°$. Dans cette position, la caméra vise donc le miroir sphérique. Le facteur de zoom est tel que la projection du miroir occupe la plus grande partie de l'image. Dans la phase de détection du mouvement, une résolution élevée de l'image n'est pas spécialement nécessaire.

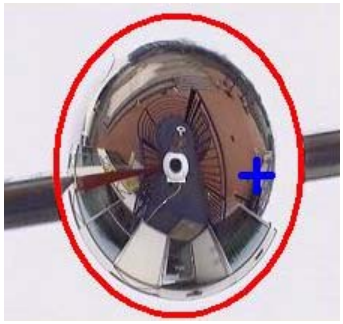


Figure 114 : Visualisation du miroir en position repos

De cette image il est possible d'extraire un panorama complet I_{p1} de la scène.

Figure 115 : Panorama de la scène extrait de l'image omnidirectionnelle

La vision totale et permanente de l'espace permet d'assurer qu'un événement isolé sera automatiquement détecté, dans la limite ou cet objet a une taille minimale sur l'image omnidirectionnelle. Lors de la phase d'installation, nous pouvons également construire un autre panorama I_{p2} de la scène vue cette fois du centre optique de la caméra PTZ en utilisant les techniques décrites aux chapitres précédents.

Figure 116 : Panorama de la scène généré par le pilotage de la caméra PTZ.

Dans la phase d'exploitation, il n'est pas nécessaire de générer ces deux panoramas. Les calculs de position peuvent être réalisés directement à partir de l'image omnidirectionnelle. Le scénario est donc simple. Dans un premier temps, la caméra est au repos et visualise l'ensemble de la scène en pointant sur le miroir sphérique. Dès la détection d'un mouvement dans l'image omnidirectionnelle, la caméra est pilotée en fonction des données récupérées sur l'image. Après identification et poursuite de l'objet en mouvement, la caméra se replace en position de repos. Pendant la phase de poursuite, la difficulté est que nous n'avons plus la vue omnidirectionnelle pour asservir la caméra sur le déplacement de l'objet. Ce problème sera abordé en détail avec la deuxième application que nous présentons. Nous allons nous intéresser ici à l'étude théorique de notre architecture et au problème lié à la calibration de l'ensemble caméra / miroir.

5.3.2. Etude théorique

Le but de ce paragraphe est de déterminer l'ensemble des équations qui nous permet de calculer les coordonnées d'un point P de la scène dans le repère du monde à partir des informations contenus dans l'image et des conditions de prise de vue.

Dans la limite où ce point de l'espace est visible dans l'image omnidirectionnelle et pour une position (θ, φ) de la caméra. Le schéma ci-dessous illustre le modèle présenté :

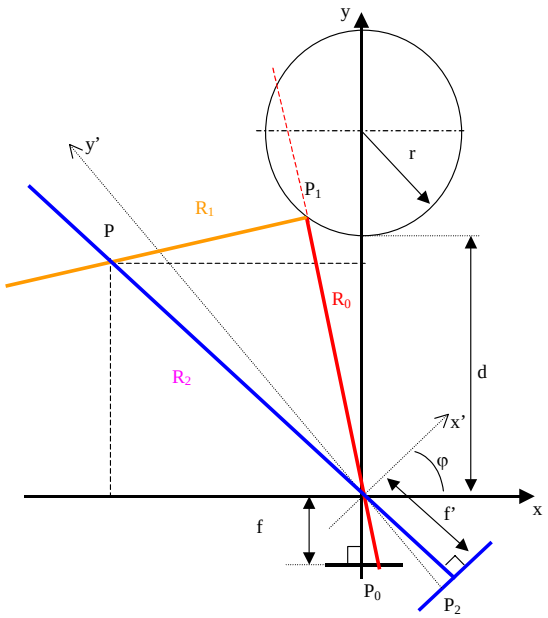


Figure 117 : Schéma de principe de l'architecture caméra PTZ / miroir sphérique

L'origine du repère correspond au centre optique de la caméra. La caméra étant au repos, l'axe $\vec{x}$ est parallèle au plan image et l'axe $\vec{y}$ est perpendiculaire au plan image. Le plan image est à une distance y = -f du centre optique et le centre de la sphère se situe sur l'axe $\vec{y}$ à une distance y = d + r.

La surface de la sphère de rayon r est décrite par l'équation :

$$x^2+y^2+z^2=r^2$$

Le centre de la sphère étant à une distance (d + r), l'équation de la surface de la sphère est donnée par :

$$x^2+\left(y-(d+r)\right)^2+z^2=r^2$$

De façon à simplifier le problème, nous allons passer en 2 dimensions. La sphère est alors un cercle dont les points du périmètre vérifient l'équation :

$$x^2+\left(y-(d+r)\right)^2=r^2 \qquad (5.1)$$

5.3.2.1 Equation du rayon R0 et point d'intersection P1

Ceci posé, nous pouvons déterminer l'équation du rayon R_0 qui passe par le centre optique et qui intersecte le plan image (ou plus exactement de la ligne image puisque nous travaillons en 2D) à la position $P_0 = (x_0, -f)$. L'équation de ce rayon est triviale et est donnée par :

$$y=\frac{-f}{x_0}x \quad \text{avec} \quad x_0 \neq 0$$

Ce rayon intersecte la surface de la sphère au point $P_1 = (x_1, \frac{-f}{x_0} x_1)$. Ce point satisfait également à l'équation :

$$x_1^2 + \left(\frac{-f}{x_0} x_1 - (d+r) \right)^2 = r^2$$

Le développement de cette équation donne :

$$\left(1 + \frac{f^2}{x_0^2} \right) x_1^2 + \frac{2f(d+r)}{x_0} x_1 + \left(d^2 + 2dr \right) = 0 \qquad (5.2)$$

Cette équation du second degré admet deux racines :

$$x_1 = \frac{-x_0 \left(fd + fr + \sqrt{f^2 r^2 - x_0^2 d^2 - 2dr x_0^2} \right)}{x_0^2 + f^2}$$

et

$$x_1' = \frac{-x_0 \left(fd + fr - \sqrt{f^2 r^2 - x_0^2 d^2 - 2dr x_0^2} \right)}{x_0^2 + f^2}$$

Ces deux racines correspondent à un point d'intersection du rayon passant par le centre optique et la sphère. Parmi ces racines la solution est donc la première intersection du rayon avec la sphère, puisque le rayon ne peut évidemment pas la traverser. Ceci correspond donc la racine dont la valeur absolue est la plus faible, c'est à dire $x_1 = x_1'$

Pour que cette racine ait une valeur réelle, il faut que $f^2 r^2 - x_0^2 d^2 - 2dr x_0^2 \geq 0$, ceci implique que :

$$|x_0| \leq \frac{fr}{d^2 + 2dr} \qquad (5.3)$$

si $|x_0| = \frac{fr}{d^2 + 2dr}$, l'équation admet une racine double. Le rayon qui passe par ce point est donc tangent au cercle et n'est pas dévié. Ce point permet également de déterminer l'angle φ_{max} couvert par la vue omnidirectionnelle. En théorie, l'angle $\varphi_{max} = 90°$ si le miroir est réduit à un point ou si il est placé à une distance infinie de la caméra. Pour $x_0 = 0$, l'angle $\varphi_{min} = -90°$ et correspond à un rayon confondu avec le centre optique de la caméra.

5.3.2.2 Equation du rayon R1

Le rayon R_0 se réfléchit au point P_1 par rapport à la normale de la courbe $x^2 + (y - (d+r))^2 = r^2$ en un rayon R_1. Le premier point P_2 qui intersectera ce rayon apparaîtra au point P_0. Notre problème est donc de retrouver l'équation d'une droite dans le cadre d'une symétrie axiale, c'est à dire une réflexion. Afin de simplifier les équations mises en jeux, nous allons procéder à un changement de repère et placer l'origine au point P_1. Une fois que nous aurons déterminé le coefficient directeur du rayon R_1, nous replacerons l'origine sur son point de départ.

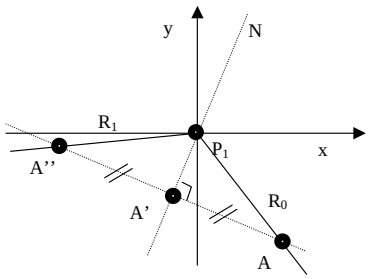


Figure 118 : Changement de repère au point P_1

On note a_{r0} le coefficient directeur du rayon R_0, a_{r1} le coefficient directeur du rayon R_1 et a_n le coefficient directeur de la droite passant par le centre de la sphère et par le point P_1. Cette droite correspondant à la normale de la surface de la sphère au point P_1.
Afin de déterminer le coefficient directeur du rayon R_1, nous cherchons à exprimer les coordonnées d'un point A" de coordonnée ($x_{A"}$, $y_{A"}$) placé arbitrairement le long du rayon R_1 et dont le symétrique par rapport à la normale est un point A de coordonnée (x_A, y_A) appartenant au rayon R_0. A partir des coordonnées ($x_{A"}$, $y_{A"}$), l'équation du rayon R1 est simplement :

$$y = \frac{y_{A"}}{x_{A"}} x = a_{r0} x$$

Nous rappelons que les droites perpendiculaires à la droite de coefficient directeur a_n, ont pour coefficient directeur $-\frac{1}{a_n}$.
La droite perpendiculaire à N et qui coupe le rayon R_0 en A, a pour équation :

$$y - y_A = -\frac{1}{a_n}(x - x_A) \qquad \Rightarrow \qquad y = -\frac{1}{a_n} x + \left(\frac{x_A}{a_n} + y_A \right)$$

Cette droite intersecte la normale N au point A'. Nous pouvons donc écrire :

$$-\frac{1}{a_n} x_{A'} + \left(\frac{x_A}{a_n} + y_A \right) = a_n x_{A'}$$

On en déduit rapidement que :

$$x_{A'} = \frac{x_A + a_n y_A}{a_n^2 + 1} \quad \text{et} \quad y_{A'} = a_n x_{A'} \tag{5.4}$$

Prenons le point A avec pour ordonnée $x_A = 1$ et par conséquent $y_A = a_{r0}$. Les coordonnées du point A' sont donc :

$$x_{A'} = \frac{1 + a_n a_{r0}}{a_n^2 + 1} \qquad \text{et} \qquad y_{A'} = \frac{1 + a_n a_{r0}}{a_n^2 + 1} a_n$$

Le point A", symétrique au point A par rapport à la normale N, vérifie les propriétés suivantes :

$$\begin{cases} x_A - x_{A'} = x_{A'} - x_{A''} \\ y_A - y_{A'} = y_{A'} - y_{A''} \end{cases} \text{ soit } \begin{cases} x_{A''} = 2x_{A'} - x_A \\ y_{A''} = 2y_{A'} - y_A \end{cases}$$

En remplaçant les coordonnées des points A et A' par leur valeur décrite plus haut, l'expression des coordonnées du point A'' deviennent :

$$x_{A''} = \frac{2(1+a_n a_{r0}) - (a_n^2+1)}{a_n^2+1} \quad \text{et} \quad y_{A''} = \frac{2(1+a_n a_{r0})a_n - (a_n^2+1)a_{r0}}{a_n^2+1}$$

L'équation du rayon R1 avec pour origine le point P1 est donc :

$$y = \frac{2(1 + a_n a_{r0})a_n - (a_n^2 + 1)a_{r0}}{2(1 + a_n a_{r0}) - (a_n^2 + 1)} x \tag{5.5}$$

Pour obtenir l'équation de ce rayon R1 dans le repère de la caméra, il suffit simplement d'effectuer un changement de repère, soit :

$$(y - y_1) = \frac{2(1 + a_n a_{r0})a_n - (a_n^2 + 1)a_{r0}}{2(1 + a_n a_{r0}) - (a_n^2 + 1)}(x - x_1)$$

avec :

$$a_n = \frac{y_1 - (d+r)}{x_1} \quad \text{et} \quad a_{r0} = -\frac{f}{x_0}$$

Afin de simplifier les expressions, on pose :

$$y = a_{r1}(x - x_1) + y_1 \quad \text{avec} \quad a_{r1} = \frac{2(1 + a_n a_{r0})a_n - (a_n^2 + 1)a_{r0}}{2(1 + a_n a_{r0}) - (a_n^2 + 1)} \tag{5.6}$$

5.3.2.3 Equation du rayon R2

Une fois que les rayons R_0 et R_1 sont déterminés, nous pouvons changer la position de la prise de vue en modifiant les valeurs θ, φ ainsi que le facteur de zoom, c'est à dire la distance focale. Dans notre optique de simplification, l'angle θ n'intervient pas, puisque nous sommes en 2 dimensions. Cependant, il faudra quand même appliquer une rotation panoramique de cet angle pour aligner les rayons R_0, R_1 et R_2. Cette première rotation étant effectuée, nous appliquons une rotation d'angle φ. Dans les équations présentées ci-dessous, le centre optique de la caméra est confondu avec le centre de rotation. Suivant le type de caméra utilisée, nous avons vu que cette condition n'est pas toujours respectée. Si tel n'est pas le cas, il faudra donc en tenir compte. Avec cette rotation d'angle φ, un changement de la distance focale peut également être envisagé voire nécessaire. Suivant le diamètre de la sphère et sa distance d par rapport au centre optique de la camera, il est fort possible qu'il soit nécessaire d'ajuster la distance focale de façon à ce que la sphère occupe la plus grande surface possible dans l'image.

L'image I' a donc subi une rotation d'angle φ autour de l'axe z. Dans le repère I', la droite R_2 qui coupe le plan image en x_2 et qui passe par le centre optique a pour équation :

$$y' = -\frac{f}{x_2} x'$$

Pour obtenir l'équation de ce rayon dans le repère de monde, il suffit simplement d'appliquer un changement de repère, consistant en une rotation d'angle φ autour de l'axe z. Ce changement de repère s'exprime de la façon suivante :

$$\begin{cases} x' = x \cdot \cos\varphi + y \cdot \sin\varphi \\ y' = -x \cdot \sin\varphi + y \cdot \cos\varphi \end{cases}$$

ce qui implique que :

$$-x \cdot \sin\varphi + y \cdot \cos\varphi = -\frac{f}{x_2}\left(x \cdot \cos\varphi + y \cdot \sin\varphi\right)$$

L'équation du rayon R_2 est donc :

$$y = \frac{x_2 \cdot \sin\varphi - f \cdot \cos\varphi}{x_2 \cdot \cos\varphi + f \cdot \sin\varphi} x$$

que nous pouvons écrire :

$$y = a_{r2} x \quad \text{avec} \quad a_{r2} = \frac{x_2 \cdot \sin\varphi - f \cdot \cos\varphi}{x_2 \cdot \cos\varphi + f \cdot \sin\varphi} \tag{5.7}$$

5.3.2.4 Coordonnée du point P

La finalité des calculs précédents est de déterminer les coordonnées d'un point P situé à l'intersection des rayons R_1 et R_2. On montre simplement que :

$$x = \frac{a_{r1} x_1 - y_1}{a_{r1} - a_{r2}} \quad \text{et} \quad y = \frac{a_{r1} x_1 - y_1}{a_{r1} - a_{r2}} a_{r2} \tag{5.8}$$

L'ensemble des équations de notre architecture étant à présent connu, nous allons aborder la calibration automatique de l'ensemble. Cette étape est particulièrement importante puisqu'elle conditionne les résultats futurs de notre application dans sa phase d'exploitation. L'intérêt d'un calibrage automatique est qu'il peut être réalisé directement par l'installateur sans formation particulière.

5.3.3. Calibrage automatique de l'ensemble

Dans le cadre de notre application, le but est de positionner correctement la caméra suite à la détection de mouvement dans l'image omnidirectionnelle de la position de repos. C'est à dire que nous devons déduire de l'image omnidirectionnelle les informations de pilotage de la caméra. En toute rigueur, pour déterminer correctement ces paramètres de pilotage, nous devons à la fois détecter l'objet en mouvement dans la vue omnidirectionnelle mais aussi connaître les coordonnées de l'objet dans le repère du monde. Il est bien évident que nous ne pouvons pas disposer de cette dernière information. C'est justement l'un des objectifs de l'application. Nous devons déterminer les paramètres de pilotage avec les seules informations disponibles sur l'image omnidirectionnelle. Nous ne détaillerons pas ici l'algorithme permettant d'extraire les objets en mouvement dans la vidéo de la vue omnidirectionnelle. Lorsque la caméra est au repos, nous pouvons considérer que le flux vidéo est issu d'une

caméra fixe. Nous utilisons donc une méthode classique basée sur la modélisation des composantes statiques de la scène à partir de mélange de gaussienne [STA99]. Par simple différence entre l'image courante et l'image de fond, nous réalisons la segmentation des objets en mouvement. Nous avons ainsi les coordonnées (u_1, v_1) de l'objet en mouvement dans l'image omnidirectionnelle. A partir de cette information, nous devons déterminer les paramètres θ et φ de pilotage de la caméra. Pour cela, nous avons mis en place une méthode de calibration automatique de l'ensemble PTZ-Omnidirectionnelle en fonction de la scène à observer. L'organigramme de l'étape de calibration est le suivant :

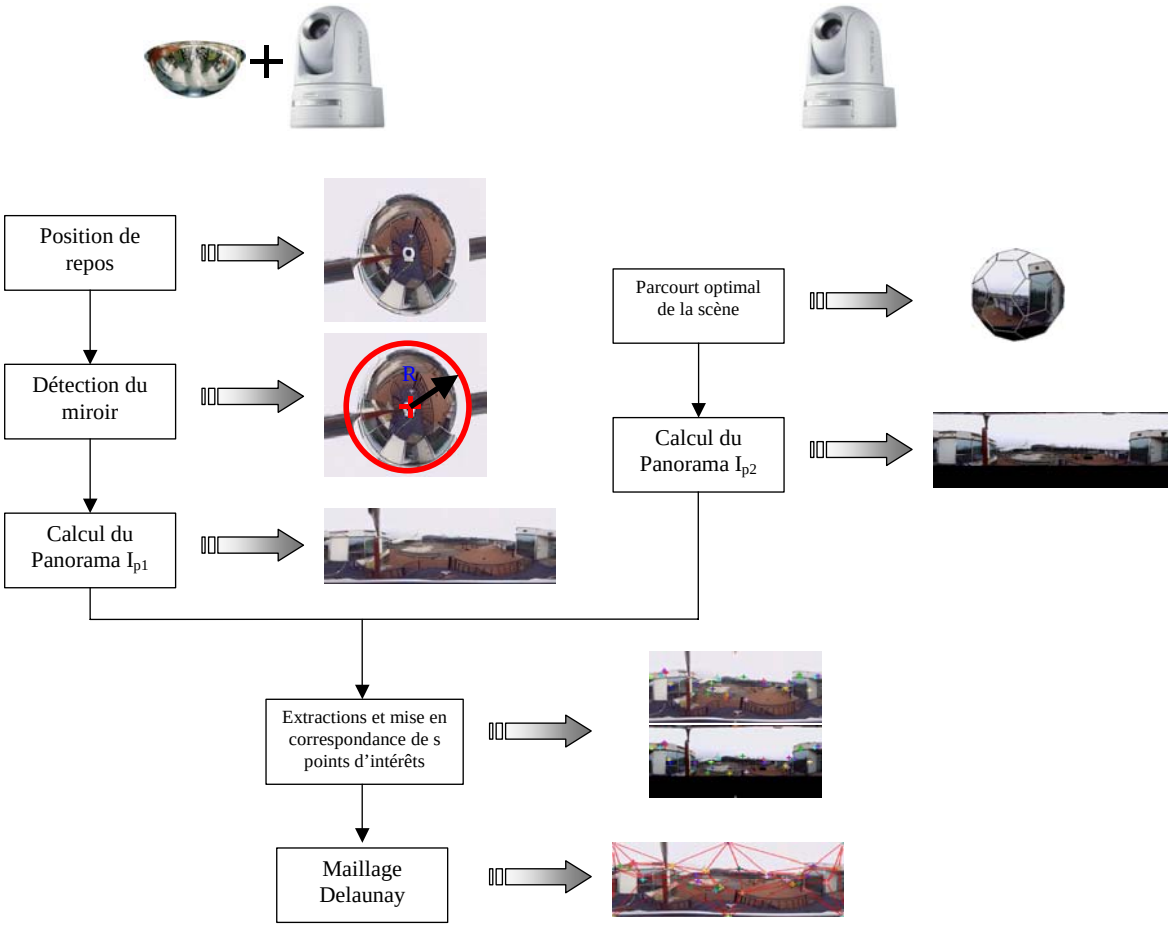


Figure 119 : Organigramme de la phase de calibrage

Cette calibration est à réaliser lorsque le système est mis dans sa position définitive d'exploitation puisque nous allons utiliser des points caractéristiques de la scène. La première étape est manuelle et consiste à définir la position de repos de la caméra. C'est à dire la position (angles et focale) dans laquelle, le miroir sphérique occupe le maximum de l'image. Le processus est ensuite entièrement automatique. Dans un premier temps, l'algorithme détecte le contour du miroir et en détermine le centre. A partir de ces informations il est possible de générer l'image panoramique Ip1. Le processus se poursuit par le pilotage de la caméra de façon à assurer un balayage complet de la scène. Afin d'optimiser le parcours complet de la scène, nous utilisons l'algorithme décrit dans le chapitre 2. A partir des différentes images acquises, nous calculons l'image panoramique Ip2. Nous avons à présent deux images panoramiques correspondant à deux points de vue différents. Nous utilisons

alors l'algorithme SIFT [LOW04] pour mettre en correspondances des points d'intérêt issus des deux images. Pour chaque point, nous connaissons exactement leurs positions dans les deux images, c'est à dire que nous pouvons en déduire les paramètres θ et φ de pilotage de la caméra. A ce stade, nous ne pouvons piloter la caméra avec précision que sur les points d'intérêt. Avec les points de l'image Ip1, nous réalisons une triangulation de Delaunay. A chaque sommet de ce maillage, nous associons les paramètres θ_{Ip1} et φ_{Ip1} de chaque point de I_{p1} aux paramètres θ_{Ip2} et φ_{Ip2} des points correspondant de I_{p2}.

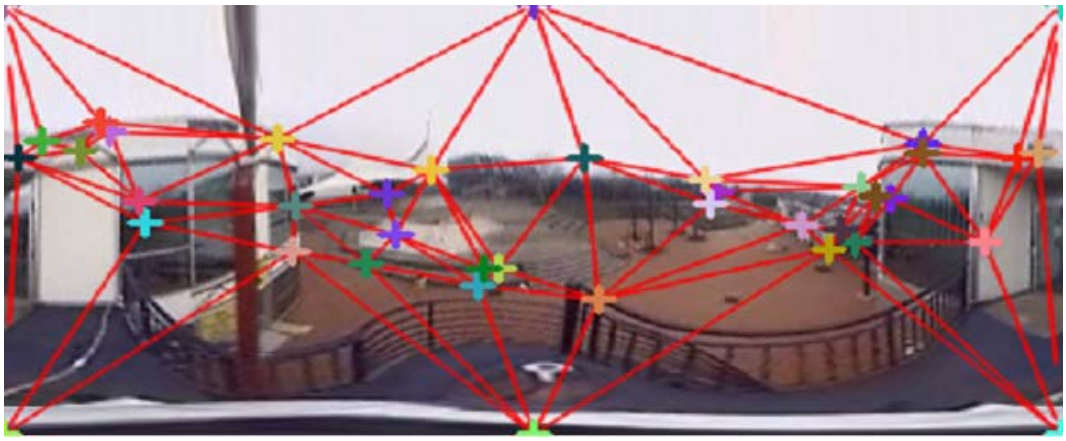


Figure 120 : Exemple d'un maillage de Delaunay réalisé sur le développement cylindrique de la vue omnidirectionnelle.

5.3.4. Résultats

Afin de tester la robustesse de notre algorithme de calibration, nous avons relevé manuellement les coordonnées θ_{Ip1}, φ_{Ip1}, θ_{Ip2} et φ_{Ip2} de plusieurs pixels dans les deux images panoramiques Ip1 et Ip2 et correspondant à un même point de la scène. Ces points étant bien sûr différents des points d'intérêt utilisés pour le maillage. Nous avons classé ces points en trois catégories : avant plan ①, plan d'évolution ② et arrière plan ③.

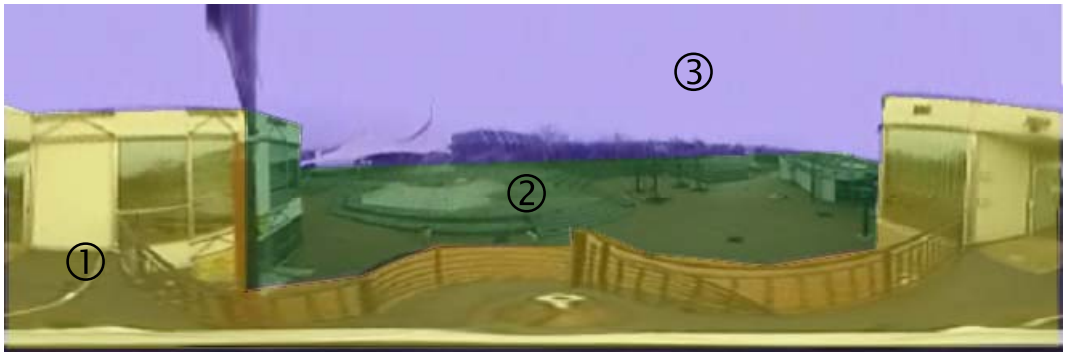


Figure 121 : Visualisation des différents plans de la scène de test

L'avant plan correspond aux points de la scène situés à moins de 5m de la caméra. Sur l'image Ip1, ces points correspondent à la passerelle et au bâtiment situé de part et d'autre de l'image. Le plan d'évolution correspond aux points de la scène situés entre 5m et 40m de la caméra. C'est essentiellement dans cette zone que les objets seront par la suite détectés. Enfin, l'arrière plan correspond à tous les points situés à plus de 40m de la caméra. Nous

avons testé trois approches différentes que nous appelons : sans calibration, linéaire et Delaunay. La premiere consiste à considérer que le centre optique de la vue omnidirectionnelle est suffisamment proche de celui de la caméra PTZ pour être considéré comme identique. C'est à dire que $\varphi_{Ip1} = \varphi_{Ip2}$. Le deuxième algorithme consiste à approximer la relation entre les angles de tangage par une transformation linéaire. Les coefficients de cette transformation étant calculés à partir d'une régression linaire des angles φ_{Ip1} et φ_{Ip2} des points d'intérêts. Dans [JOH98] l'auteur montre que si la mesure de corrélation entre plusieurs couples de valeur est supérieure à 0.6 alors la relation entre ces deux couples de valeur peut être approximé par une transformation linéaire. A partir des mesures que nous avons effectuées, nous obtenons un score de 0,95. Nous pouvons donc raisonnablement utiliser cette approximation. Dans le troisième algorithme nous utilisons notre méthode basée sur le maillage de Delaunay des points d'intérêts. Pour chaque plan et pour chaque méthode, nous donnons la moyenne de la valeur absolue des erreurs constatées entre la transformation de φ_{Ip1} vers φ_{Ip2c} et le relevé manuel φ_{Ip2m} des points de notre corpus de test. Ces mesures sont synthétisées dans le tableau suivant :

Méthodes	Avant plan	Plan d'évolution	Arrière plan
Sans calibration	5.4°	11.3°	15.0°
Linéaire	8.4°	5.4°	13.6°
Delaunay	2.8°	4.1°	4.8°

Tableau 12 : Erreurs de pilotage de φ_{Ip2c} en fonction des différentes calibrations

Nous pouvons remarquer que la première calibration est une approximation correcte pour les objets en avant plan. Par contre elle ne donne pas de bons résultats pour les deux autres plans. La méthode linéaire par contre donne de bons résultats pour les objets du plan d'évolution. Etant donné que nous allons essentiellement détecter des objets dans ce plan, nous pourrions considérer que cette approximation est suffisante. Si nous devons également détecter des objets dans les deux autres plans, les résultats sont nettement moins bons. Dans tous les cas, notre méthode donne de bons résultats et une meilleure approximation quel que soit le plan considéré.

Afin de valider l'étude théorique que nous avons présentée, nous avons réalisé une autre série de mesures. Pour cela, nous avons placé la caméra dans une pièce dont nous connaissons les mesures exactes. Nous avons relevé manuellement les mesures de différents points caractéristiques de façon à obtenir une vérité terrain. Sur l'image omnidirectionnelle de la caméra, nous avons également relevé les coordonnées des points caractéristiques. Enfin, pour chaque point, nous avons piloté la caméra de façon à ce que le point se situe au centre de l'image et nous avons relevé les valeurs θ et φ correspondantes. Nous avons donc les mesures de la vérité terrain ainsi que les différentes mesures nécessaires au calcul de la profondeur. Nous présentons les résultats obtenus sous la forme du tableau suivant. La première colonne correspond à la distance mesurée entre le point caractéristique et le centre optique de la caméra. La deuxième colonne correspond au calcul de cette distance à partir des données images. Enfin la troisième colonne correspond à l'erreur entre la mesure et le calcul exprimé en pourcentage par rapport à la mesure.

Mesure (mm)	Calcul (mm)	Err %
1879	2407	22
1325	1585	16
1741	2617	33
3744	3974	6
3135	4036	22
3289	3860	15
2048	1634	-25
2285	1858	-23
3568	3067	-16
2922	2421	-21
3087	2614	-18

Tableau 13 : Mesures de la cartes de profondeurs

Nous pouvons constater que sur ces quelques données, l'erreur de la mesure est de l'ordre de 20%. La mesure n'est donc pas très précise mais elle permet néanmoins d'obtenir une approximation de la distance de l'objet vis à vis de la caméra. Il y a essentiellement deux facteurs qui viennent perturber la mesure. Le premier est que le miroir que nous utilisons n'est pas réellement sphérique. Pour atténuer les défauts de sphéricité nous pourrions étalonner notre miroir comme proposé dans [RAM05]. Le deuxième est que notre caméra ne correspond pas exactement au modèle théorique que nous avons utilisé. Notamment, comme nous l'avons précisé dans le chapitre 3, le centre des rotations n'est pas confondu avec le centre optique.

Pour terminer, nous avons testé l'ensemble de notre algorithme sur un cas concret. L'ensemble caméra/miroir visualise une scène en extérieure dans laquelle des personnes se déplacent. Pour ce test, la caméra visualise uniquement le miroir sphérique. De cette vue, nous utilisons un algorithme afin de segmenter les objets en mouvement et déterminer leur trajectoire. Nous comparons les coordonnées de pilotage issues des trois approches par rapport à une vérité terrain. L'exemple ci dessous (Figure 122a) représente la trajectoire d'un objet en mouvement dans la vue omnidirectionnelle.

(a) *(b)*

Figure 122 : Trajectoire d'un objet en mouvement calculé à partir de la vue omnidirectionnelle de la caméra en position de repos (a) et sa projection dans le développement cylindrique (b)

Comme nous pouvons le constater sur le graphique suivant (Figure 123), l'algorithme sans calibration ne donne pas de bons résultats. Les deux autres algorithmes sont relativement équivalents et se rapprochent de la trajectoire idéale avec cependant un léger mieux en ce qui concerne l'approche qui utilise un maillage de Delaunay.

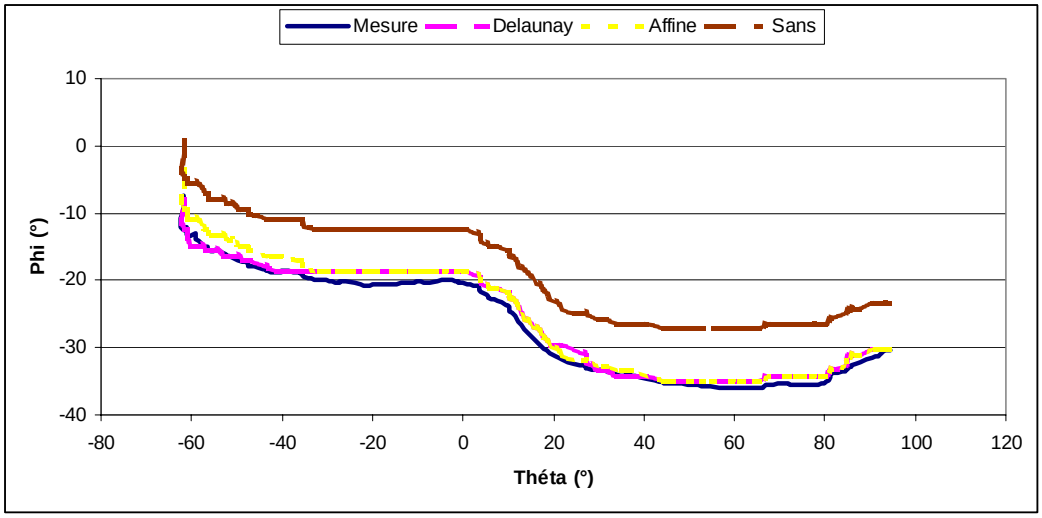


Figure 123 : Comparaison des coordonnées de pilotage issues des trois algorithmes

Il est intéressant de noter que l'approche « Delaunay » est plus efficace au début de la trajectoire ($-60 < \theta < 20$) que sur la fin où elle est relativement confondu avec la méthode affine. Ceci est du au fait que dans la scène, l'algorithme SIFT que nous avons utilisé a détecté plus de point sur la gauche que sur la droite (Figure 120). Le maillage étant plus serré à gauche, l'algorithme de pilotage est plus efficace.

5.3.5. Conclusion

Cette première architecture que nous avons détaillée nous permet de réaliser la surveillance complète d'une scène ainsi que la focalisation sur un objet en mouvement avec simplement une caméra PTZ et un miroir. Cette architecture nous permet, lorsque la caméra est en position de repos, d'assurer la détection de n'importe quel événement. Sans le miroir, nous ne pourrions pas assurer cette détection dans la mesure où la caméra seule ne peut visualiser qu'une portion de la scène à chaque instant. La limite de cette architecture est que si deux événements surviennent l'un après l'autre alors que la caméra n'a pas terminé l'analyse du premier, le deuxième ne sera peut-être pas détecté.

Ce travail de recherche a été accepté à la conférence AVSS 2009 et sera présenté à Gènes en septembre 2009. En l'état, l'algorithme présenté ne nous permet pas de réaliser la poursuite des objets. Dans le paragraphe suivant, nous présentons un algorithme qui permet d'extraire, du flux vidéo, les objets en mouvement dans la scène alors que la caméra est elle-même en mouvement. Nous présenterons également les deux applications que nous avons développées.

5.4. Détection et suivi des objets en mouvement

5.4.1. Introduction

L'application que nous venons de présenter nous permet de détecter un objet en mouvement et de piloter la caméra de façon à se focaliser sur l'objet. L'étape suivante consiste à segmenter cet objet et à le suivre au cours de son déplacement. Parmi les différentes

approches de segmentation, nous nous intéressons tout particulièrement à une classe basée sur la modélisation des composantes statiques de la scène. Notre objectif est de réaliser une classification de des objets à partir de caractéristiques extraites de leur contour, à travers une segmentation fine de ces objets. Nous avons vu dans l'état de l'art que les solutions proposées dans la littérature consistent, pour la plupart, à réaliser un panorama de l'arrière plan la scène. Cependant, réaliser un panorama complet de la scène est particulièrement coûteux en place mémoire. Pour stocker l'ensemble de la scène sans perte d'information sur les faces d'un cube, nous avons montré qu'il faut que la taille minimale de chaque arête du cube soit égale à deux fois la distance focale exprimée en pixel. A titre d'exemple, prenons une distance focale de 800up correspondant à un objectif grand angle de 4.1mm avec un capteur 1/3". Pour une image couleur l'espace mémoire nécessaire est donc de 1600^2 x 3 x 6 soit environ 45MO. Si nous utilisons un algorithme à base de mélange de gaussienne pour modéliser l'arrière plan. Sachant qu'une solution minimaliste (nous y reviendrons dans le paragraphe suivant) nécessite 3 entiers de 16 bits par distribution et qu'il faut au minimum deux distributions, l'espace mémoire est alors de 540 MO. La taille mémoire n'est pas le seul critère limitatif. Pour que le modèle de fond ait un sens, il est nécessaire que le délai entre la modélisation du fond et le calcul de la carte des pixels de l'avant plan sur l'image courante soit le plus court possible. Si ce délai est trop long, plusieurs facteurs peuvent entraîner des modifications dans la scène rendant l'extraction des objets en mouvement difficile voir impossible. Le changement de luminosité est un de ces facteurs. Mettre à jour en permanence l'ensemble de la scène n'est pas envisageable. La solution que nous proposons consiste donc à ne modéliser que la partie du fond visible par la caméra à chaque instant.

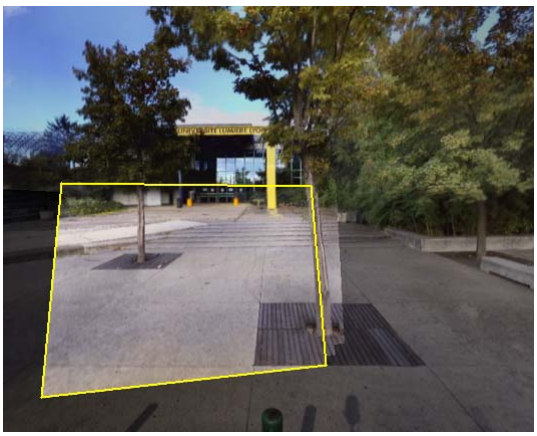


Figure 124 : Extrait d'un panorama

L'exemple ci-dessus (Figure 124) représente une partie du panorama. A chaque instant, la seule partie de la scène vue par la caméra se limite à la zone à l'intérieur du polygone. Le reste n'est plus visible. Nous allons donc considérer que la zone visible et effacer de la mémoire tout le reste.

Parmi les différentes méthodes de segmentation, nous avons fait le choix d'étudier les méthodes basées sur la modélisation d'une image de fond et parmi cette classe de méthodes, nous utilisons l'approche des mélanges de gaussiennes. Avant de présenter notre solution, nous allons nous arrêter sur la modélisation des composantes statiques de la scène basée sur les mélanges de gaussienne.

5.4.2. Les Mélanges de Gaussiennes

Dans les cas des caméras fixes, les mélanges de gaussiennes sont réputées efficaces pour modéliser le fond et extraire les objets en mouvement. Cette méthode, proposée par Stauffer et Grimson [STA99], approxime la variation de chaque pixel par une ou plusieurs distributions gaussiennes. Une solution minimaliste consiste à utiliser simplement deux distributions. Ces deux distributions sont caractérisées par une valeur moyenne μ_1, μ_2 un écart type σ_1, σ_2, et un nombre de point n_1, n_2 correspondant au poids de chaque distribution. Elles représentent, pour l'une, les pixels correspondant à l'image de fond en niveau de gris et pour l'autre les pixels correspondant aux objets en mouvement ou plus exactement aux objets de l'avant plan. Le calcul de la moyenne et de l'écart type fait intervenir le nombre de points. Ce nombre de points permet de définir la distribution qui, à l'instant t, correspond au fond. La paramétrisation de chaque pixel nécessite donc 6 valeurs. Pour permettre le calcul incrémental de la moyenne et de l'écart type, on ne mémorise pas directement la valeur de la moyenne et de l'écart type, mais la somme des valeurs et la somme des différences entre la valeur au carré du pixel et de la moyenne. En toute rigueur, nous devrions mémoriser la somme des valeurs et la sommes du carré des valeurs. Cependant, afin de limiter l'espace mémoire occupé par la construction, nous n'utilisons que des entiers de 16 bits. Dans ce cas, la mémorisation de la somme du carré des valeurs engendre rapidement un débordement alors que dans notre cas, l'écart entre la valeur et la moyenne étant rarement supérieur à 10 niveau de gris, un entier de 16 bits permet de stocker la somme de plusieurs centaines de valeurs.

Soit $A_{k,t}$ la somme des valeurs des pixels au temps t, $B_{k,t}$ la somme de la différence entre la valeur au carré du pixel et de la moyenne au temps t, p_t la valeur du niveau de gris du pixel et $\mu_{k,t}$ la valeur moyenne des pixels pour l'une des distribution avec $k \in \{1,2\}$.

$$A_{k,0}=0 \ ; \ A_{k,t}=A_{k,t-1}+p_t \tag{5.9}$$

$$B_{k,0}=0 \ ; \ B_{k,t}=B_{k,t-1}+(p_t-\mu_{k,t})^2 \tag{5.10}$$

A partir des valeurs A et B, le calcul de la valeur moyenne et une approximation de la variance sont donnés par :

$$\mu_{k,t}=\frac{A_{k,t}}{n_{k,t}} \ ; \ \upsilon_{k,t}=\frac{B_{k,t}}{n_{k,t}} \tag{5.11}$$

Du point de vue mathématique, le calcul que nous faisons de la variance (et donc de l'écart type) est faux mais reste une approximation suffisante dans notre cas.

L'étiquetage des pixels et la mise à jour des paramètres sont relativement simples. Pour chaque pixel de la nouvelle image, nous vérifions s'il appartient au fond. La distribution correspondant au fond est celle dont le nombre de points est le plus important. Pour l'étiquetage, nous vérifions que l'écart entre la valeur du pixel et la valeur moyenne de la gaussienne du fond est inférieur à un seuil fixé à trois fois la valeur de l'écart type. Si l'écart est inférieur au seuil alors le pixel appartient au fond ; sinon il est dans le premier plan, et donc considéré comme faisant partie d'un objet en mouvement. La mise à jour de l'une ou l'autre des distributions suit un schéma un peu différent. Nous cherchons pour cela de quelle distribution la valeur du pixel est la plus proche. Plusieurs calculs sont possibles. Le premier consiste à dire que la mise à jour de telle ou telle distribution correspond à l'étiquetage du pixel mais cela revient à mettre à jour la distribution correspondant au fond uniquement

lorsque l'écart entre le pixel et la valeur moyenne est inférieur au seuil fixé sur l'écart type. Ce calcul a pour effet de resserrer l'écart type au cours du temps mais ne permet pas de prendre correctement en compte une variation progressive de la luminosité. Un autre calcul consiste à fixer le seuil d'appartenance $S_{12,t}$ à une valeur correspondant à la moyenne des valeurs moyennes des deux distributions.

$$S_{12,t}=\frac{\mu_{1,t}+\mu_{2,t}}{2} \tag{5.12}$$

Cela revient à considérer que les deux distributions ont le même écart type. Une autre solution plus formelle consiste à calculer la probabilité d'appartenance du pixel à telle ou telle distribution. Cela revient à calculer :

$$\underset{i}{\operatorname{argmin}}\left(\frac{p_{i,t}-\mu_{i,t}}{\sigma_{i,t}}\right) \tag{5.13}$$

C'est la solution retenue par Stauffer et al. [STA00]. Dans un article, Mardia et al.[MAR88] proposent d'autres algorithmes permettant de déterminer le seuil optimal de séparation de deux distributions.

Cet algorithme donne de bons résultats dans le cas des caméras fixes. Nous l'avons adapté pour l'utiliser dans le cas d'une caméra PTZ en mouvement,.

5.4.3. Modélisation du fond avec une caméra PTZ

Comme nous l'avons expliqué précédemment, l'évolution de chaque pixel du model de fond est décrit par une ou plusieurs distributions gaussiennes. Le nombre de distribution est variable pour chaque pixel au cours du temps et il est fonction du mouvement dans l'image. Chaque distribution est caractérisée par un triplet (μ, σ, n), représentant respectivement la valeur moyenne, l'écart type et l'effectif.

A l'instant t, un pixel de l'image de fond est caractérisé par un ou plusieurs triplets.

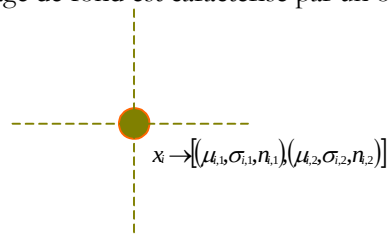


Figure 125 : Caractérisation d'un pixel de fond.

A l'instant t+1, la caméra s'est déplacée et nous considérons ici que nous avons déterminé par ailleurs la matrice de transformation H entre le plan image de l'itération précédence et le plan image de l'itération courante. Soit une image de fond $I_{f,\,t-1}(f_f, \theta_f, \varphi_f)$ calculée lors de l'itération précédente et l'image courante $I_{c,t}(f_c, \theta_c, \varphi_c)$.

(a) Image de fond I_f *(b) Image courante I_c*

Figure 126 : Image de fond et image courante d'une séquence. Les polygones représentent la zone de recouvrement des deux images

Avant de mettre à jour le modèle de fond, nous projetons le modèle de fond de l'itération précédente dans le plan de l'image courante, à partir de la matrice H de transformation.

Figure 127 : Projection de l'image de fond dans le plan de l'image courante

La difficulté est que les « grilles » des deux images ne se superposent pas forcément. Nous devons donc trouver une solution permettant de replacer ou diffuser les informations contenues dans chaque pixel de l'image de fond sur la grille de l'image courante.

Une première solution consiste affecter l'intégralité des caractéristiques d'un pixel sur le plus proche voisin de la grille discrète.

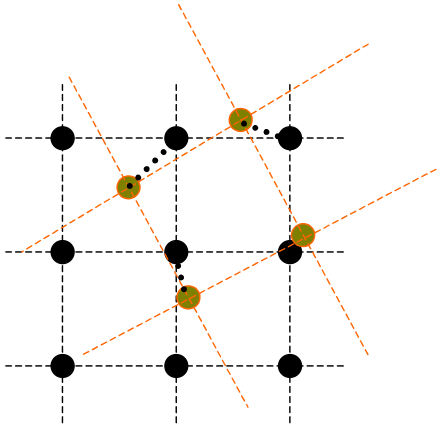
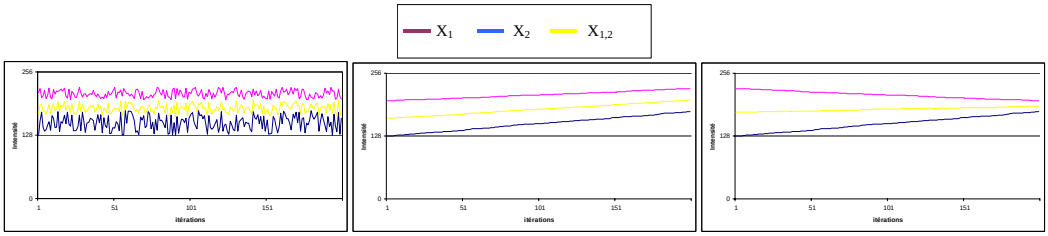


Figure 128 : Application de la solution du plus proche voisin

Une deuxième solution consiste à diffuser l'information d'un pixel sur les quatre pixels voisins de la grille discrète. Comme nous l'avons indiqué dans le paragraphe 3.2 « Interpolation bilinéaire », un calcul possible de cette diffusion est de réaliser une interpolation bilinéaire. Contrairement au cas que nous avions étudié dans ce paragraphe, nous avons à diffuser non seulement une couleur moyenne mais également un écart type. Lorsque nous calculons la fusion de deux distributions, l'ordre dans lequel les valeurs de chaque pixels est considéré n'a pas d'importance pour le calcul de la valeur moyenne, par contre il est important pour le calcul de l'écart type. Or, nous avons perdu la référence temporelle. Afin d'illustrer notre propos, nous allons nous placer dans un cas plus simple qui consiste simplement à fusionner deux pixels voisin. Nous représentons sur les graphiques suivant (Figure 129), l'évolution de deux pixels voisins X_1 et X_2 ainsi que leur fusion $X_{1,2}$ dans trois cas distincts. Pour ces trois cas, nous utilisons le même jeu de valeur qui généré aléatoirement. Dans le premier cas (Figure 129a), les valeurs des pixels X_1 et X_2 sont dans l'ordre dans lequel ils ont été générés. Ce premier cas pourrait correspondre à l'évolution de deux signaux bruité. Dans le deuxième cas (Figure 129b), nous classons les deux tableaux de point dans l'ordre croissant. Ce deuxième cas pourrait correspondre à deux signaux qui augmente continûment de façon aléatoire. Enfin, dans le troisième cas (Figure 129c), l'un des tableaux (X_1) est classé dans l'ordre croissant de ses valeurs et l'autre (X_2) dans l'ordre décroissant. Dans les 3 cas, la fusion $X_{1,2}$ correspond à la moyenne des deux signaux :

$$X_{1,2}=\frac{X_1+X_2}{2}$$



(a) *(b)* *(c)*

Figure 129 : fusion de deux pixels voisins dans trois cas distincts
(a) ordre aléatoire
(b) ordre croissant pour le deux signaux
(c) ordre croissant pour l'un et décroissant pour l'autre

Dans ces trois exemples, le jeu de données étant le même, la valeur moyenne et l'écart type des deux courbes X_1 et X_2 calculées sur l'ensemble des valeurs sont donc les mêmes. Il en est de même pour la valeur moyenne $\mu_{1,2}$ de la fusion $X_{1,2}$ puisque celle ci peut s'écrire :

$$\mu_{1,2}=\frac{1}{n}\cdot\sum_{i=1}^{n}\frac{X_1^i+X_2^i}{2}=\frac{1}{n}\cdot\frac{\sum_{i=1}^{n}X_1^i+\sum_{i=1}^{n}X_2^i}{2} \qquad (5.14)$$

Les deux sommes pouvant être calculées indépendamment, l'ordre des points n'a pas d'importance. Par contre, ce n'est pas le cas de l'écart type, puisque l'expression suivante ne peut pas se décomposer en sommes indépendantes.

$$\sigma_{1,2}^2 = \frac{1}{n} \cdot \sum_{i=1}^{n} \left(\frac{X_1^i + X_2^i}{2} - \mu_{1,2} \right)^2 \tag{5.15}$$

Dans le cas (b), comme nous associons les valeurs les plus faibles des deux signaux ainsi que les valeur les plus fortes, nous avons tendance à augmenter la dynamique de la fusion et donc son écart type. Dans le cas (c), nous associons les valeurs plus faibles de l'un avec les valeurs plus fortes de l'autre et inversement. Nous avons donc tendance à diminuer la dynamique de la fusion et son écart type. Finalement, l'écart type des trois scénarios n'a pas la même valeur :

$$\sigma_{1,2}(c) \leq \sigma_{1,2}(a) \leq \sigma_{1,2}(b) \tag{5.16}$$

Comme nous n'avons pas suffisamment d'information pour calculer correctement l'écart type de la fusion de deux signaux nous considérons que la moyenne des écarts types est une approximation suffisante dans notre cas, validée expérimentalement, et nous écrivons :

$$\sigma_{1,2} \approx \frac{\sigma_1 + \sigma_2}{2} \tag{5.17}$$

Nous généralisons ce principe aux quatre pixels voisins et ainsi cette approximation nous permet d'aligner les pixels de l'image de fond sur la grille de l'image courante par une interpolation bilinéaire. Une fois que l'image de fond est projetée dans le plan de l'image courante (Figure 127), nous pouvons la mettre à jour comme dans le cas d'une caméra fixe.

La taille de l'image de fond étant la même que celle de l'image courante, nous perdons une partie de l'information. C'est à dire, la zone de l'image de fond qui était présente sur l'image précédente et qui a disparu avec le déplacement de la caméra. Sur l'illustration précédente (Figure 127) cette zone a été hachurée avec des lignes diagonales. Cette perte d'informations n'a pas grande importance puisque le mouvement de la caméra est la plupart du temps linéaire et qu'il est fort probable qu'il va se poursuivre dans la même direction. Ce qui est plus gênant, c'est qu'une partie de l'image de fond n'est pas disponible. C'est à dire la zone de l'image courante qui n'était pas présente sur l'image précédente est qui est hachurée avec des croix (Figure 127). Dans cet exemple nous avons volontairement appliqué un mouvement important afin d'illustrer notre propos.

L'approche que nous proposons se généralise à d'autres modèles de fond plus simples comme l'image précédente ou une image calculée par une moyenne glissante. Dans ce cas, il suffit de projeter cette image de fond I_f dans le plan de l'image courante I_c et de calculer la différence entre ces deux images. Dans ce cas, la carte des pixels de premier plan I_m est donnée par :

$$I_m(\mathrm{i,j}) = \begin{cases} 1 \; si \; (I_c(\mathrm{i,j}) - H.I_f(\mathrm{i,j})) > Seuil \\ 0 \; sinon \end{cases} \tag{5.18}$$

où *Seuil* est la valeur limite appliqué à l'ensemble des pixels.

Ces deux autres solutions sont très simple à calculer, mais comme dans le cas des caméras fixes, la segmentation obtenue avec les mélanges de gaussienne est de bien meilleure qualité.

5.4.4. Résultats

Afin de vérifier la pertinence de notre approche, nous avons réalisé une séquence de test. Dans cette séquence, la caméra effectue une rotation dans le sens trigonométrique autour de l'axe de panorama pour un angle de tangage égal à zéro. La scène est filmée sans objet en mouvement.

Le premier test que nous effectuons consiste à déterminer la qualité du modèle de fond obtenu. Nous utilisons pour cela le rapport signal/bruit PSNR (peak signal to noise ratio) exprimé en db qui pour deux images I_f et I_c en niveaux de gris se définit comme suit :

$$PSNR = 10 \cdot \log_{10}\left(\frac{255^2}{err}\right) \tag{5.19}$$

où *err* est l'erreur quadratique moyenne entre les deux images et est donnée par :

$$err = \frac{1}{n} \cdot \sum_n \left(I_c(i,j) - H \cdot I_f(i,j)\right)^2 \tag{5.20}$$

La définition du PSNR que nous présentons est celle utilisée pour mesurer la qualité de la compression. Traditionnellement, une bonne qualité de compression se traduit par un PSNR compris entre 30 db et 40 db. Pour chaque approche, nous calculons deux valeurs de PSNR. La première est calculée à partir de l'image de fond donnée par le modèle et l'image courante. La deuxième est calculée sur la carte des pixels en mouvement et la carte théorique qui ne doit comporter aucun mouvement. Les résultats que nous obtenons sur notre séquence de test sont donnés dans le tableau suivant :

Modélisation du fond	Fond PSNR (db)	Carte de MVT PSNR (db)
Image précédente	45	38
Moyenne glissante	44	34
MG – Plus proche voisin	28	30
MG – Interpolation	35	38

Tableau 14 : PSNR du modèle de fond et de la carte d'avant plan pour différentes modélisations

Ce tableau montre qu'en l'absence de mouvement, le modèle de fond le plus précis est l'image précédente. En effet, dans ce cas, le bruit est essentiellement dû au calcul de la projection. Sur le calcul de la carte de mouvement, notre approche par interpolation donne d'aussi bons résultats que le calcul de l'image précédente. Il peut paraître paradoxal qu'avec une image de fond moins bonne, nous ayons une carte de qualité comparable. Une étude plus précise des différentes cartes, montre que, quel que soit le modèle de fond utilisé, les défauts se situent essentiellement au niveau des gradients forts de l'image. Dans le cas de l'image précédente ou de la moyenne glissante, un seuil unique doit être défini empiriquement pour l'ensemble des pixels de l'image. Dans le cas des mélanges de gaussiennes, le seuil de chaque pixel est défini en fonction de son écart type de sorte qu'il s'adapte mieux au niveau des forts gradients. Dans ces premiers résultats, nous observons que dans le cas des mélanges de gaussiennes, notre approche par interpolation donne de meilleurs résultats que l'approche par le plus proche voisin.

Nous poursuivons notre test afin de définir la qualité de la segmentation. Dans la même séquence d'images, nous introduisons artificiellement un rectangle au centre de l'image que nous maintenons pendant 10 images. Ce rectangle simule un objet en mouvement dans l'image. L'intérêt est que nous connaissons exactement sa taille et sa position. Sur une période de 20 images, nous avons 10 images sans objet en mouvement suivi de 10 images avec un objet en mouvement.

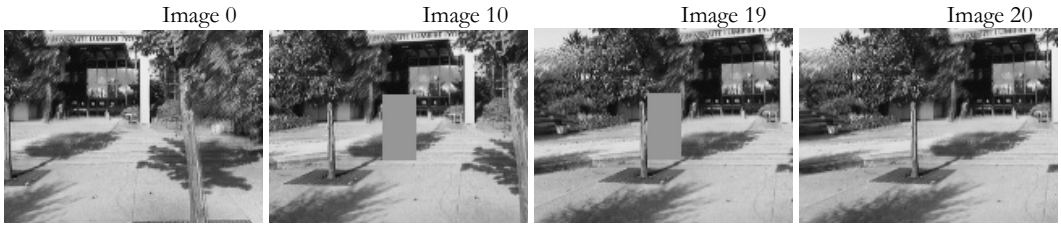


Figure 130 : Extrait de la séquence d'image

Connaissant exactement la taille et la position de l'objet que nous avons placé artificiellement, nous pouvons comparer le calcul de la carte de mouvement avec la carte « idéale ». Nous présentons dans le graphe suivant, le PSNR de la carte de mouvement en fonction de l'indice de l'image dans la séquence pour les trois approches différentes :

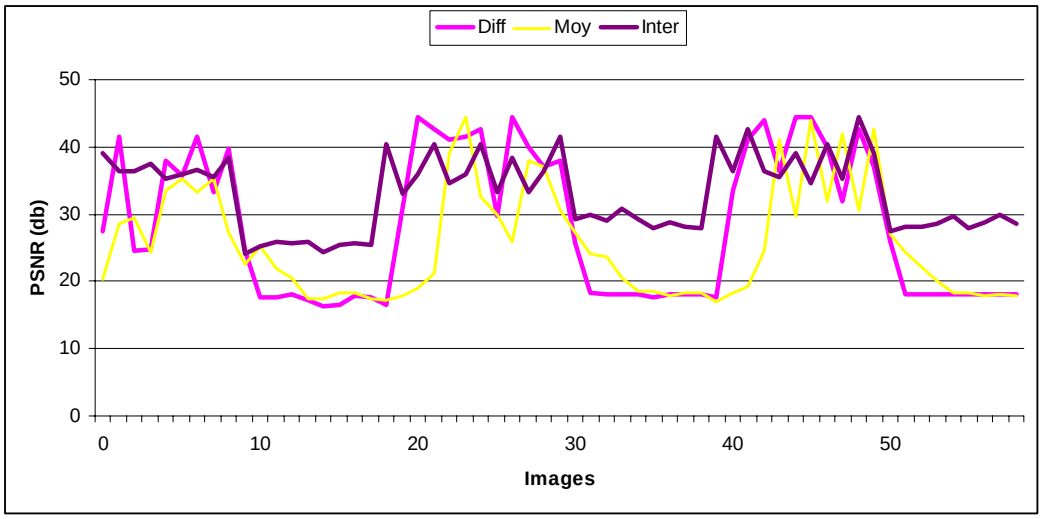


Figure 131 : PSNR de la carte de mouvement calculé à partir des trois approches

où

- Diff correspond à la différence entre l'image courante et l'image précédente,
- Moy est la méthode de la moyenne glissante,
- Inter est notre approche avec interpolation bilineaire.

Sur ce graphique, nous pouvons tout d'abord remarquer que lors de la présence du motif (images 10-19, 30-39 et 50-59) le PSNR des différentes méthodes est plus faible. Cependant, dans le cas de notre approche et en présence du motif, le PSNR est compris entre 25 et 30 alors qu'avec les deux autres solutions, il est inférieur à 20. D'autre part, nous remarquons également que lors de la disparition du motif, le PSNR de notre approche augmente

immédiatement alors qu'avec les deux autres, il y a un délai de réaction. Ceci se traduit par le fait que notre approche détecte immédiatement la disparition du motif. Enfin, la mesure du PSNR est plus stable dans le cas de notre approche qu'avec les deux autres méthodes.

En conclusion, l'utilisation de l'image précédente ou le calcul de la moyenne glissante pour modéliser le fond sont très simples à mettre en œuvre. Elles peuvent être utilisées pour détecter un mouvement dans l'image, par contre elles ne permettent pas une bonne segmentation. Une autre difficulté de ces solutions est qu'il faut fixer arbitrairement un seuil global à l'ensemble des pixels de la carte pour réaliser l'étape de binarisation. Notre approche basée sur les mélanges de gaussienne avec interpolation bilinéaire permet de définir un seuil spécifique à chaque pixel ce qui rend la méthode plus robuste. Sur cet exemple, avec la méthode de l'image précédente, 30% des pixels représentant l'objet en mouvement ont été détectés. Avec le calcul de la moyenne glissante ce taux est de 42% et il atteint 80% avec la méthode des mélanges de gaussiennes. Ces résultats sont conformes à ceux que nous aurions eus avec une caméra fixe. Ils permettent cependant de mettre en évidence que l'approche que nous proposons permet bien de se replacer, après transformation, dans le cas d'une caméra fixe.

5.4.5. Solution globale

La solution globale que nous proposons nécessite, au préalable, une étape de recalage. Nous utilisons pour cela notre approche basée sur la minimisation des distances entre paire de points par la méthode du simplexe. Pour tester la solution globale, nous utilisons une séquence réelle filmée avec notre caméra. Dans cette séquence, la caméra subit une rotation suivant l'axe panoramique. Pendant ce mouvement, un personnage est en train d'évoluer dans la scène. Afin d'illustrer l'apport de chaque étape de notre algorithme, nous présentons les résultats sous la forme suivante. La première colonne du tableau correspond aux images que nous avons extraites de la vidéo. La colonne suivante correspond à un zoom sur le personnage que nous cherchons à segmenter. Les trois colonnes suivantes sont le résultat de la binarisation suivant trois niveaux de complexité de l'algorithme. La colonne (WR) est le résultat de la binarisation en utilisant les mélanges de gaussienne mais sans faire subir de transformation à l'image. Comme entre deux prises de vue, l'ensemble de la scène est modifié, on conçoit aisément que les résultats ne vont pas être exploitables. Pour la colonne (CP) nous utilisons également les mélanges de gaussiennes mais cette fois ci en appliquant une transformation homographique calculée à partir des paramètres de prise de vue transmis par la caméra. Enfin, la colonne (OA) correspond à notre approche globale dans laquelle nous calculons précisément les paramètres de la transformation à l'aide de notre méthode de recalage et où nous appliquons cette transformation au modèle de fond.

Id. Frame	Image	Pers.	WR	CP	OA
311					
316					

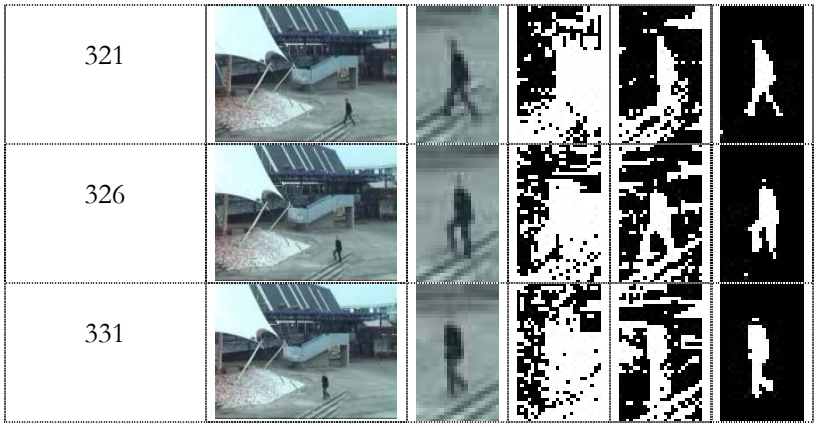


Figure 132 : Exemple de segmentation des objets en mouvement avec une caméra PTZ

Comme nous pouvions l'imaginer, la première méthode (WR) ne donne pas de résultats exploitables. Ceci justifie pleinement l'application d'une transformation. La méthode (CP) n'est guerre plus exploitable. Malgré l'application de la transformation, les paramètres de prise de vue donnés par la caméra manquent de précision. Par contre, en appliquant notre méthode de recalage, la segmentation des objets en mouvement est possible. Nous pouvons alors appliquer les algorithmes de suivi et de classification que nous avons déjà utilisées avec succès sur les caméras fixes.

5.5. Conclusion

Ces travaux de recherche sont à la base de deux applications concrètes que nous avons développées.

La première est un assemblage de ce que nous avons présenté dans les deux paragraphes précédents. Nous utilisons une caméra PTZ et un miroir sphérique. En position de repos, la caméra visualise le miroir pour obtenir une vue complète de la scène. Lorsqu'un objet en mouvement est détecté dans cette position, nous calculons les paramètres de prise de vue et nous pilotons la caméra pour qu'elle se focalise sur l'objet. Nous utilisons alors le deuxième algorithme pour détecter de nouveau l'objet dans cette nouvelle vue et nous réalisons un pilotage automatique de la caméra. Nous avons toutes les briques pour réaliser un recalage précis, la modélisation du fond et l'extraction des objets. A partir de la position de l'objet dans l'image courante et les paramètres précis de la prise de vue, nous pouvons déterminer une estimation de la position de l'objet dans le repère de la caméra. Comme, le centre optique ne se déplace pas avec le mouvement de la caméra, nous n'avons pas l'information de profondeur. Cependant, nous pouvons déterminer les angles de panorama et de tangage de la caméra de façon à ce que l'objet reste au centre de l'image. De même, nous pouvons adapter la distance focale de façon à ce que l'objet ait une certaine taille dans l'image courante. Pour que cette application soit entièrement finalisée, nous avons encore à améliorer l'asservissement de la caméra en prenant en compte le temps de réaction des moteurs.

La deuxième est une application de « balayage ». Le principe de base est de laisser la caméra réaliser une ronde vidéo par un mouvement continu. A partir de notre algorithme nous pouvons extraire les objets en mouvement dans la scène. L'avantage de pouvoir réaliser une segmentation assez fine des objets permet également le suivi de ces objets ainsi que leur

indentification. A partir des points du contour des objets en mouvement, nous calculons un vecteur de caractéristiques comprenant entre autre : le nombre de points du contour, la surface, le centre de masse et le centre de gravité des points du contour, l'excentricité, la compacité, l'angle de l'axe principal, les moments jusqu'à l'ordre 3 et les premiers coefficients de la transformée de Fourier des points du contour. Ce vecteur de caractéristiques constitue la signature de l'objet. Cette signature est utilisée d'une part pour suivre les objets au cours du temps et d'autre part pour permettre l'identification à partir d'un apprentissage supervisé. L'organigramme de cette première application est donné si dessous.

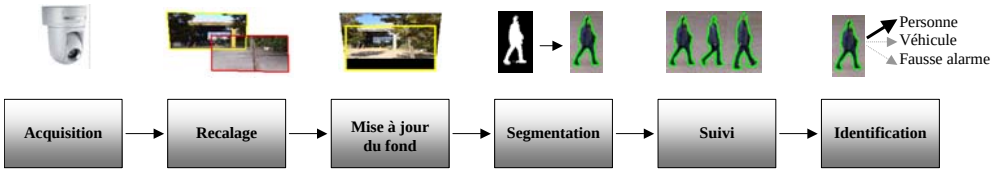


Figure 133 : Organigramme de l'application de scanning

Dans cette application, l'étape de suivi se limite à suivre les objets en mouvement dans le champ de vue de la caméra sans intervenir sur le pilotage de la caméra. Une version de cette application a été adaptée aux caméras thermiques.

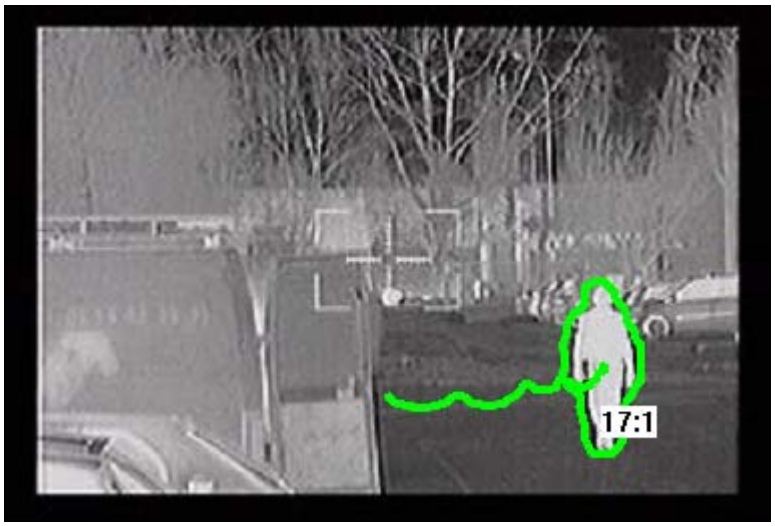


Figure 134 : Exemple de détection avec une caméra PTZ thermique

L'algorithme de base de cette application à fait l'objet d'une publication dans les actes de la conférence VISAPP 2009.

Mosaïque d'images multi-résolution et applications

Chapitre 6 : Conclusion

6. Conclusion

6.1. Bilan des travaux

Un des intérêts des caméras PTZ est qu'il est possible d'interagir sur les paramètres de la prise de vue en fonction des résultats de l'analyse de la vidéo à l'instant précédent. Une première partie de notre travail a été consacré au pilotage et à la calibration la caméra SONY RZ25P. Nous avons pour cela repris se qui ce fait classiquement dans la littérature [FAU93], ce qui nous à permis de déterminer les paramètres permettant de corriger les distorsions géométriques de l'objectif. Pour des distances focales faibles, ces distorsions ont principalement pour effet de courber vers l'extérieur de l'image les droites en fonction de leur éloignement du centre de l'image. La modélisation de ce problème fait l'objet d'une abondante littérature. Les modèles pouvant être utilisés sont plus ou moins complexes et nécessitent plus ou moins de paramètres en fonction du degré de simplification. La calibration nous a également permis de déterminer de modéliser la commande de pilotage de la distance focale. L'un des intérêts des caméras PTZ est que leur centre optique est relativement peu éloigné de leur centre de rotation. Ce qui permet, sous certaines conditions, de considérer que l'ensemble de la scène est acquis à partir d'un centre de projection unique ce qui simplifie la transformation mathématique. Une fois ce travail terminé, nous avons commencé à réaliser nos premiers panoramas et nous avons rapidement été confrontés à différentes difficultés.

La première concerne l'acquisition des images. Cependant, l'un des intérêts des caméras PTZ est que lorsque le centre optique de la caméra est confondu avec le (ou les) centre(s) de rotation, il n'est pas possible d'obtenir une notion de la profondeur même en multipliant les prises des vues pour des angles différents. Dans ces conditions, nous pouvons donc ignorer l'information de profondeur et projeter tous les points sur une sphère de rayon fixé. Notre problématique est donc de parcourir l'ensemble des points d'une sphère à partir de la projection d'images rectangulaires. Nous avons réalisé une étude théorique de la projection des bords des images dans une sphère. A partir de ces équations, nous proposons un parcours de la sphère permettant de minimiser le nombre de prises de vue.

La deuxième difficulté nous a beaucoup plus mobilisés. Dans le cas idéal, une caméra PTZ devrait nous permettre de disposer des paramètres exacts de la prise de vue. Dans la réalité ce n'est pas forcément le cas. Avec certains types de matériel, il n'est tout simplement pas possible de connaître ces paramètres. Avec d'autres caméras, comme celle que nous utilisons, la précision de la commande n'est pas suffisante pour obtenir une projection robuste avec simplement les données de la caméra. Nous devons donc déterminer les paramètres de la prise de vue à partir des données contenues dans les images. L'approche que nous avons retenue est d'utiliser le recalage d'images pour déterminer la matrice de projection. Le but du

recalage d'images est d'aligner géométriquement deux images ou plus de sorte que des pixels respectifs ou leurs dérivés (bords, coin, etc..) représentant la même structure fondamentale puissent être mis en correspondance. L'idée fondamentale derrière la plupart des algorithmes décrits dans la littérature est de mettre en correspondance les images selon leurs propriétés radiométriques ou géométriques en utilisant une fonction spécifique pour évaluer la qualité de la mise en correspondance. Dans un premier temps, nous avons testé plusieurs méthodes décrites dans la littérature. N'ayant pas obtenu des résultats conformes à nos attentes, nous avons commencé par étudier plus précisément les mesures de similarité puis nous avons proposé une méthode de recalage à la fois robuste et rapide.

Afin de proposer une application de visualisation, nous avons cherché à améliorer le rendu et dans un premier temps, à corriger le problème lié à l'illumination de la scène. En fonction de l'angle de prise de vue, la scène peut être plus ou moins éclairée. Pour corriger ce changement de luminosité, nous pouvons soit fixer les paramètres d'ouverture et de gain de la caméra, soit laisser la caméra adapter automatiquement ces paramètres au mieux en fonction de la luminosité de la portion de scène visée. Dans le premier cas si la dynamique entre les zones sombres et les zones fortement éclairées est trop forte, les zones sombres risquent d'apparaître sous-exposées et les zones claires sur-exposées. Dans le second cas, nous allons voir apparaître des pavés de luminosités différentes avec ce que nous appelons communément des coutures. Il existe aujourd'hui des caméras qui adaptent automatiquement le gain localement en fonction de la luminosité. Ces caméras sont particulièrement bien adaptées pour gérer les contre-jours. Dans le cas, malheureusement le plus courant, où il n'est pas possible de fixer le gain de la caméra, nous proposons une solution de correction du gain par diffusion basée sur le calcul d'une carte de distances. Cependant, cette correction de luminosité n'est réellement utile que lorsque nous désirons construire une mosaïque d'images pour ensuite la visualiser. Dans le cas d'une application de détection et de suivi, la fréquence d'acquisition est plus rapide. De fait, le changement de luminosité entre deux images successive est relativement faible et il peut être lissé par la modélisation du fond que nous avons mise en place. Un autre problème que l'on rencontre classiquement lors de la construction d'un panorama avec des images qui ne sont pas acquises dans le même temps, est la suppression des fantômes. A la différence d'un appareil photo, l'avantage d'utiliser une caméra est que nous pouvons traiter un flux vidéo plutôt qu'une image unique. Afin de supprimer les fantômes dans nos panoramas, nous avons contourné la difficulté. Pour chaque prise de vue, nous faisons l'acquisition d'une séquence d'images et nous calculons une modélisation des composantes statiques de la scène observée à partir d'un algorithme basé sur les mélanges de gaussienne [STA99].

Nous avons ensuite étudié plusieurs applications de suivi et de classification d'objets en mouvement. Avant d'aborder le problème du suivi, nous avons eu à résoudre celui de la détection des objets en mouvement. Dans une activité de vidéo surveillance, l'intérêt des caméras PTZ est de permettre une vision très élargie du périmètre à surveiller. Leur principal défaut est que lorsque la caméra est orientée dans une direction, elle ne permet pas de visualiser ce qui ce passe dans le reste de la scène. Dans le cas d'une activité de surveillance urbaine et lorsqu'un opérateur est devant son écran, le pilotage de la caméra en orientation et en zoom permet une surveillance et une levée de doute efficace. Dans le cas d'une détection d'intrusion, il peut être important de surveiller l'ensemble de la zone en permanence de façon à détecter de façon certaine tout événement et de ne piloter la caméra que lorsqu'un événement survient. Plusieurs auteurs utilisent dans ce cas deux caméras. Une caméra fixe avec un objectif grand angle ou de type fish-eye ou encore une caméra associée à un miroir sphérique ou parabolique et une caméra PTZ asservie par la première. Nous proposons une première solution basée sur l'utilisation d'une seule caméra PTZ associé à un miroir

sphérique. Dans la plupart des cas de détection d'intrusion la nuit ou le week-end, il n'y a absolument aucune activité dans le périmètre à surveiller, par contre, dès qu'il se passe quelque chose, le système doit réagir immédiatement pour contrôler la nature de cet événement et déclencher une alerte si nécessaire. Pour cette simple détection de mouvement, une résolution élevée de l'image n'est pas spécialement nécessaire. Elle sera nécessaire pour l'analyse, mais pas pour la phase de détection. Dans ce cas, la caméra est en position de repos et vise le miroir sphérique. La vision totale et permanente de l'espace permet d'assurer qu'un événement isolé sera automatiquement détecté. En analysant les pixels en mouvement dans cette image omnidirectionnelle, il nous est possible de déterminer la position dans l'espace de l'objet en mouvement et de piloter la caméra en conséquence. Dans le cadre de cette première application, nous proposons une calibration automatique de l'ensemble Camera/Miroir permettant de faciliter la mise en œuvre de notre solution.

Cette première phase de détection et de focalisation terminée, nous nous sommes intéressés à la segmentation des objets en mouvement et à leur suivi. Notre idée directrice est, à travers une segmentation fine des objets, de permettre une classification de ces objets à partir de caractéristiques extraites de leur contour. La plupart des ces algorithmes de segmentation d'objet en mouvement ont été développés pour les caméras fixes ne peuvent pas être utilisés directement sur des caméras PTZ. Parmi ces algorithmes de détection, nous nous sommes intéressés plus particulièrement aux algorithmes basés sur la modélisation d'une image de fond ne comportant que les composantes statiques de la scène. Parmi ces algorithmes, les mélanges de gaussiennes ont prouvé leur efficacité. Nous avons donc étudié une généralisation de ces algorithmes, appliquée aux caméras PTZ en mouvement. Une fois le modèle de fond projeté sur le plan de l'image courante, il peut être utilisé comme un modèle de caméra fixe. Nous pouvons alors utiliser les méthodes de segmentation, de suivi et d'identification décrit dans la littérature pour les caméras fixes. Finalement, la seule différence dans le traitement des informations est qu'il ne faut pas considérer la position des objets en pixel dans l'image mais en coordonnées sphériques. Cette généralisation nous a permis d'obtenir de bons résultats pour le suivi des personnes dans une scène en mouvement.

6.2. Perspectives

Même si l'ensemble de ces travaux de recherche a donné lieu à la fois à des publications dans les actes de conférences internationales mais aussi à des applications commerciales, il reste beaucoup de travail à faire. L'une des premières pistes d'améliorations que nous allons étudier concerne l'asservissement de la caméra. Actuellement, l'algorithme de poursuite que nous avons mis en place fonctionne en boucle ouverte. Il n'est donc pas très robuste. Pour atténuer le défaut de pilotage, nous devons volontairement diminuer la distance focale de façon à augmenter l'angle solide. La poursuite reste possible mais nous perdons en résolution. Par ailleurs, nous n'exploitons pas tout le potentiel de pilotage de notre caméra et nous aimerions calculer automatiquement des cartes de profondeur de la scène observée. Comme nous l'avons indiqué à plusieurs reprises, si le centre de projection est unique entre les différentes prises de vue, nous ne pouvons obtenir de notion de profondeur. Or, le calibrage de notre matériel nous a montré que suivant la distance focale, la position du centre optique n'est pas le même. Nous allons donc chercher à exploiter cette particularité de notre matériel. L'idée étant de réaliser deux acquisitions d'une même portion de la scène avec deux prises de vues différentes et de comparer l'écart de la position des pixels représentant le même point par rapport à l'écart théorique. Cependant, d'autres défauts de construction vont devoir être correctement pris en compte. Il s'agit du décentrage et d'un défaut d'inclinaison

entre le capteur CCD et l'objectif. Pour la construction de panorama robuste comme pour les applications que nous avons développées, ces défauts ont pu être négligés. Dans une application de métrologie pixelique voire subpixélique, ces défauts doivent être modélisés et correctement pris en compte. Nous envisageons également une autre voie pour établir cette carte de profondeur. Il s'agit d'utiliser la profondeur de champ en contrôlant la distance focale et l'ouverture.

Mosaïque d'images multi-résolution et applications

Chapitre 7 : Annexes

« Si la géométrie oblige à contempler l'essence, elle nous convient ; si elle s'arrête au devenir, elle ne nous convient pas. (…) Elle a pour objet la connaissance de ce qui est toujours et non de ce qui naît et périt. Par suite, mon noble ami, elle attire l'âme vers la vérité, et développe en elle cet esprit philosophique qui élève vers les choses d'en haut les regards que nous abaissons à tort vers les choses d'ici-bas. Il faut donc, autant qu'il se peut, prescrire aux citoyens de ta Callipolis de ne point négliger la géométrie ; elle a d'ailleurs des avantages secondaires qui ne sont pas à mépriser. Ceux que tu as mentionnés, et qui concernent la guerre ; en outre, pour ce qui est de mieux comprendre les autres sciences, nous savons qu'il y a une différence du tout au tout entre celui qui est versé dans la géométrie et celui qui ne l'est pas. » **Platon** (428-347 av. J. C.), La République, Livre VII, 526

7. Annexes

7.1. Modèles mathématiques

7.1.1. Avant propos

Nous rappelons dans ce chapitre quelques éléments fondamentaux de l'optique géométrique ainsi que quelques définitions. Après ces quelques notions simples, nous présentons un résumé du modèle sténopé. Pour plus d'information, le lecteur pourra se référer aux ouvrages de O. Faugeras, "Three-Dimensional Computer Vision: a Geometric View-point" [FAU93], de Hartley, "Multiple View Geometry in Computer Vision." [HAR04] et B. Balland, « Optique géométrique : Imagerie et instruments » [BAL07].

7.1.2. Eléments d'optique géométrique

Nous allons formaliser ici quelques concepts simples de l'optique géométrique.

7.1.2.1 Les bases de l'optique géométrique

Le concept de rayon de lumière introduit par Euclide est l'élément de base de l'optique géométrique. Ce rayon de lumière n'a pas d'existence physique, mais il indique la direction de propagation de la lumière. La propagation de la lumière est étudiée en lui associant une infinité de rayons de lumière indépendants les uns des autres. Un ensemble de rayons de lumière provenant d'une même source est appelé faisceau. Cette représentation a pour principal intérêt de dissocier le trajet de la lumière de l'onde électromagnétique dont elle est composée, ceci afin de permettre de traiter les problèmes d'optiques par de simples constructions géométriques. Avec le concept de rayon de lumière, l'optique géométrique repose sur quelques principes et lois simples. Tout d'abord, le principe de la propagation rectiligne dans un milieu transparent, homogène et isotrope. C'est l'un des premiers principes à avoir été énoncé. L'air, par exemple, est un milieu qui ne répond pas à cette définition puisque son indice absolu varie en fonction de la température et de la pression. Une autre limite du domaine de validité de la propagation rectiligne de la lumière correspond au phénomène de diffraction que l'on peut observer lorsque l'on fait passer un faisceau de lumière à travers un trou mince. Un autre principe important est l'indépendance des rayons

de lumière. Pour simplifier, si deux faisceaux de lumière provenant de deux sources différences et dont les directions se croisent en un point, le point d'intersection ne provoque pas d'interférence particulière entre les rayons. Un dernier principe fondamental correspond au *retour inverse* de la lumière. Le trajet emprunté par la lumière dans un sens est le même dans l'autre sens (réciprocité du rayon incident). Parmi les lois de l'optique géométrique, celles de Snell-Descartes sont particulièrement importantes. Elles décrivent les phénomènes lumineux engendrés par l'impact de la lumière sur la matière : la réflexion et la réfraction. Pour reprendre la vision de Descartes, la réflexion correspond au « rebond » de la lumière sur une surface plane. De façon plus formelle, le rayon réfléchi est dans le plan d'incidence, plan défini par le rayon incident et la normale à la surface réfléchissante. La réfraction est la déviation des rayons lumineux passant obliquement d'un milieu transparent à un autre milieu transparent d'indice différent. Lors de la réfraction sur la surface dioptrique, le rayon de lumière coupe la normale et se réfracte selon une direction définie par l'angle i_2 lié à l'angle d'incidence i_1 par la relation des sinus :

$$n_1 \cdot \sin i_1 = n_2 \cdot \sin i_2$$

où n_1 et n_2 sont les indices de réfraction des deux milieux.

7.1.2.2 Systèmes optiques

Un système optique est un ensemble de milieux transparents, homogènes et isotropes d'indices de réfraction différents disposés les uns à la suite des autres. Ils sont séparés par des surfaces polies, appelées dioptres, de formes géométriques simples (plans, sphères) ou plus complexes (paraboloïdes, ellipsoïdes, hyperboloïdes). Ces surfaces peuvent être réfringentes (ou réfractantes). C'est à dire qu'une fraction plus ou moins importante de l'énergie incidente est transmise d'un milieu à l'autre. Dans le cas où la *quasi* totalité de l'énergie incidente est réfléchie, on parle alors de surface réfléchissante (miroir). Les systèmes optiques peuvent être classés en trois catégories :

- les systèmes dioptriques ne comportant que des surfaces réfringentes. Ces systèmes ont une face d'entrée et une face de sortie distincte.
- les systèmes catoptriques ne comportant que des surfaces réfléchissantes.
- les systèmes catadioptriques comportant à la fois des éléments réfringents et réfléchissants.

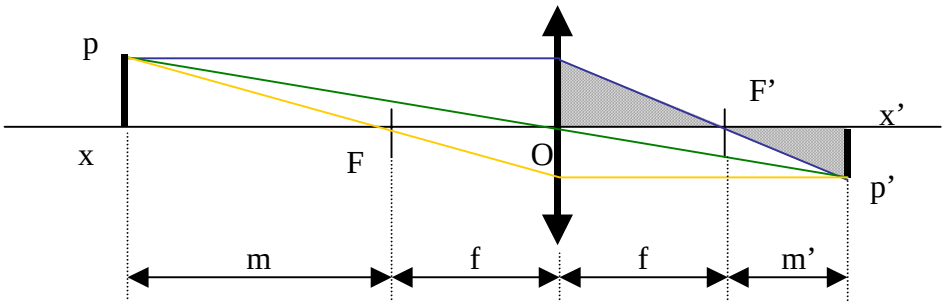
7.1.2.3 Conditions de Gauss

Afin de simplifier l'étude des phénomènes optiques, nous nous placons généralement dans les conditions de Gauss. C'est en publiant ses « Dioptrische Untersuchungen » entre 1838 et 1841 que Carl Friedrich Gauss (1777 – 1855) introduit les notions de plans principaux et points principaux. Son objectif est, sous réserve de certaines conditions (dites conditions de Gauss), de disposer d'un outil mathématique permettant justement de faciliter l'étude des instruments optiques (télescope, jumelle). L'approximation de Gauss consiste, en lumière monochromatique, à n'utiliser que les trajets des rayons pour lesquels le système fonctionne dans des conditions de stigmatisme approché. Le stigmatisme réel stipule que l'image d'un point est un point. Le stigmatisme est dit approché lorsque l'image d'un point est une tache suffisamment petite pour être considérée comme un point. Pour être dans les conditions de Gauss :

- les points objets doivent être voisins de l'axe optique,
- les rayons utilisés pour la formation des images sont très peu inclinés.

7.1.2.4 Grandissement

Si l'on se place dans les conditions de Gauss, alors on considère que la diffraction de la lumière et les aberrations de la lentille sont négligeables. On peut tracer le diagramme suivant :



F : foyer objet
F' : foyer image
f : distance focale

Quelques propriétés :

- Un rayon parallèle à l'axe optique qui atteint la lentille passe par le foyer image en sortant ,
- Un rayon qui passe par le centre optique de la lentille n'est pas dévié,
- Un rayon qui passe par le foyer objet et qui atteint la lentille ressort parallèle à l'axe optique.

On note g le grandissement correspondant au rapport entre la projection de l'objet sur le plan image et l'objet.

$$g=\frac{\overline{x'p'}}{\overline{xp}}$$

D'après le théorème de Thales, les triangles oxp et ox'p' étant semblables, le grandissement s'exprime aussi de la forme :

$$g=\frac{m+f'}{m+f}$$

Sur le diagramme, on remarque également que les deux triangles grisés sont également semblables. Il en va de même pour les deux triangles associés coté objet. Le grandissement peut s'exprimer :

$$g=\frac{m'}{f}=\frac{f}{m}$$

De cette dernière relation, on en déduit :

$$m\times m'=f^2$$

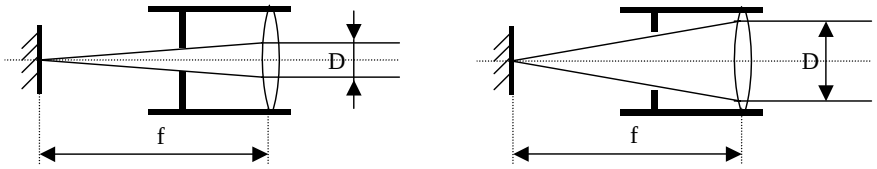
7.1.2.5 Ouverture ou Diaphragme

Le nombre d'ouverture n, contrôlé par le diaphragme, détermine la quantité de lumière qui atteint le capteur CCD. Il détermine aussi la profondeur de champ. Le nombre d'ouverture correspond au rapport entre la distance focale et le diamètre du faisceau utile au niveau de la lentille.

$$n=\frac{f}{D}$$

où n est le nombre d'ouverture, f la distance focale et D le diamètre du faisceau au niveau de la lentille.

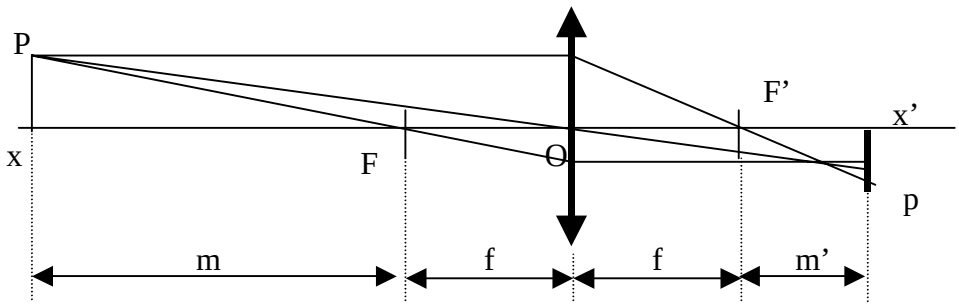
Lorsqu'on ferme le diaphragme, on augmente le nombre d'ouverture.



Diaphragme fermé, ouverture grande ouverture petite *Diaphragme ouvert,*

7.1.2.6 Netteté et profondeur de champ

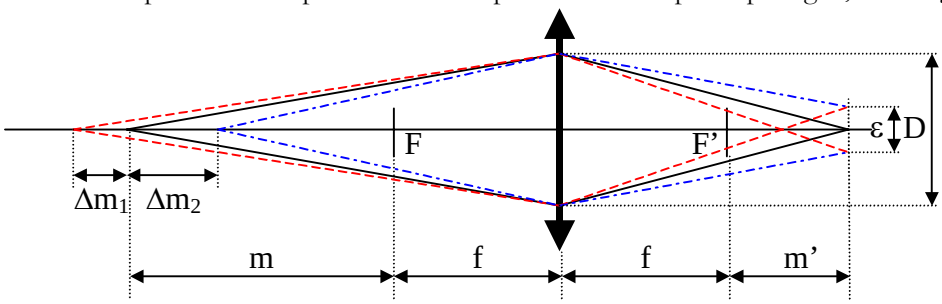
Si l'on place le plan image à une distance m' du foyer image, alors seuls les points situés à une distance m du foyer objet seront net. Dans le diagramme suivant, l'objet est placé plus loin que la distance m. On remarque alors que le point P ne se projette plus sur un point p unique sur le plan image, mais sur un petit disque. On a donc une image floue. Nous attirons l'attention du lecteur sur la signification du terme « image » que nous employons. En optique, une image est forcément nette, sinon c'est une tache. Dans ce qui suit, nous utilisons le terme « image » suivant sa signification dans la vie courante : reproduction visuelle d'un objet réel.



La profondeur de champ est l'incrément distance pour lequel l'image reste nette. En toute rigueur, cette distance est donc nulle quelle que soit la distance focale. Dans la pratique, on se donne une tolérance. On considère que l'image est nette tant que le diamètre de la tache correspondant à la projection d'un point p unique est inférieur à ε. En photographie, pour une pellicule 24x36, on fixe généralement :

$\varepsilon = 30\mu m$

Dans le cas d'une caméra numérique, la valeur ε sera celle du diamètre du cercle inscrit dans une cellule correspondant à un pixel. Pour un capteur ¼' et 752 pixels par ligne, ε = 4.3µm



Si Δm1 et Δm1 correspondent à l'incrément de distance pour lequel la projection d'un point sur le plan image est inférieure ou égale à ε, la profondeur de champ est donnée par :

$$\Delta m_1 + \Delta m_2 = \frac{2 \cdot f^2 \cdot \varepsilon \cdot n \cdot m(m+f)}{f^4 - \varepsilon^2 \cdot n^2 \cdot m^2} \quad \text{avec} \quad n = \frac{f}{D}$$

De cette équation, nous pouvons tirer les enseignements suivant :

- si l'ouverture augmente, c'est à dire si on ferme le diaphragme, la profondeur de champ augmente,
- si la focale diminue, la profondeur de champ augmente,
- si la distance de mise au point m augmente, la profondeur de champ augmente.

7.1.3. Le modèle Sténopé

7.1.3.1 Camera observa

L'acquisition d'une image à partir d'un capteur CCD est un phénomène complexe. Le phénomène optique est plus simple à comprendre puisqu'il s'agit d'un phénomène lumineux naturel que l'on peut expérimenter facilement. C'est l'expérience très simple de la chambre noire. Aristote est vraisemblablement le premier à avoir expérimenté ce phénomène. Le 25 janvier 1544 à Louvain, Reinerus Gemma-Frisius a utilisé le principe de la chambre noire ou « caméra observa » pour observer une éclipse du soleil. Il utilisera plus tard l'illustration suivante dans son livre « de radio astromomica et géométrico » publié en 1545. On pense que c'est la première illustration de la chambre noire.

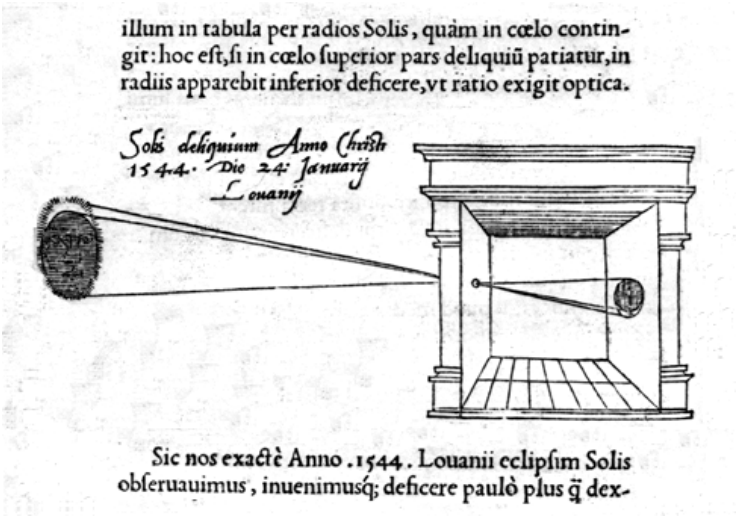


Illustration de la « Camera observa » - Reinerus Gemma-Frisius – 1545

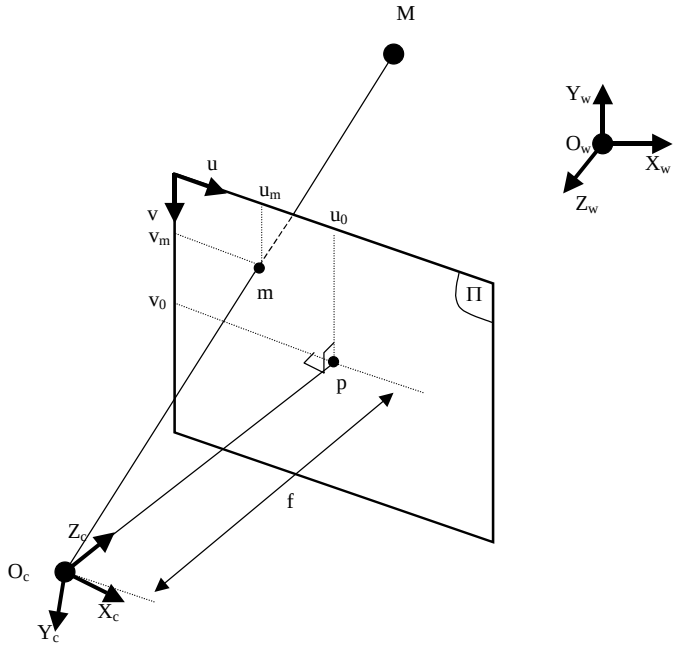
Les peintres du XVI^{ième} siècle connaissaient bien ce phénomène et comprirent très vite le bénéfice qu'ils pouvaient en retirer. Ils construisirent d'immenses chambres noires, parfois transportable, qui leur permettaient, comme avec un calque, de reproduire des paysages extérieurs. Le trou, appelé sténopé, ne devait pas être trop grand pour donner une projection nette, mais pas trop petit non plus pour laisser passer suffisamment de lumière. Il fut employé par de nombreux artistes, dont Giovanni Baptista della Porta, Vermeer, Guardi et Antonio Canal, dit Canaletto, qui l'utilisa notamment pour mettre en perspective ses célèbres paysages des canaux de Venise. Le dispositif fut amélioré en utilisant une lentille convergente qui donnait le même résultat avec une luminosité et une netteté meilleure. Le système se perfectionna en plaçant des boîtes coulissantes les unes dans les autres et par l'utilisation d'un miroir incliné à 45° pour redresser l'image. On essaya à maintes reprises de fixer l'image sur un support. En 1826, Nicéphore Niepce réussi à obtenir, après une exposition de 8 heures, une image stable sur une plaque d'étain enduite de bitume de Judée. La photographie venait de naître.

7.1.3.2 Définitions

Le modèle sténopé (pinhole) est le modèle projectif le plus souvent utilisé. Il permet un certain nombre de simplifications. Ce modèle est défini par deux éléments : un point, appelé centre optique et un plan ne contenant pas le point, appelé plan rétinien.

Dans ce modèle, le repère de la scène et le repère de la caméra sont composés d'axes orthogonaux. Le repère de la scène W est défini par son origine O_w et par ses trois axes (X_w, Y_w, Z_w). Le repère de la caméra C est défini par son origine O_c et par ses trois axes (X_c, Y_c, Z_c). Le point origine O_c correspond également au centre optique de la caméra. L'axe optique est normal au plan rétinien appelé aussi plan image Π et coupe le plan au point p. A des fins de simplification, nous considèrons que l'axe optique est confondu avec l'axe Z_c. Le plan Π est défini par deux vecteurs orthogonaux u et v. Dans ce repère, le point p a pour coordonnées (u_0, v_0). La distance O_cP correspond à la distance focale f.

La projection d'un point M de la scène de coordonnées (x_w, y_w, z_w) sur le plan image correspond au point m de coordonnée (u_m, v_m).



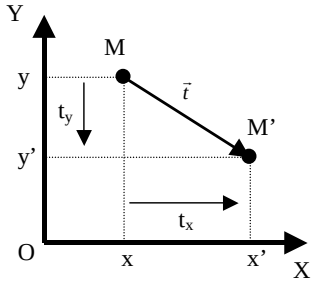
Dans cette représentation simplifiée du modèle sténopé, les paramètres extrinsèques sont les paramètres permettant le changement de repère de la scène vers le repère camera. Les paramètres intrinsèques sont la distance focale f et la position du point p dans le repère image. Il est à noter que le point p ne correspond pas forcement au centre de l'image. Dans la pratique, les constructeurs de caméras s'efforcent de faire correspondre ces deux points, mais des imperfections dans la réalisation de l'optique et le jeu mécanique nécessaire au montage engendrent un léger décalage dont il faudra tenir compte en fonction de la précision visée.

7.1.3.3 Paramètres extrinsèques

Le point M est défini par ses coordonnées (x_w, y_w, z_w) dans le repère de la scène, mais il peut également être défini par ses coordonnées (x_c, y_c, z_c) dans le repère de la caméra. Le plan image étant orthogonal à l'axe optique qui est lui-même confondu à l'axe Zc, un changement de repère est inévitable. Les paramètres extrinsèques sont donc les paramètres qui permettent le calcul du changement de repère entre le repère de la scène et le repère de la caméra.

7.1.3.3.1 Transformation rigide 2D

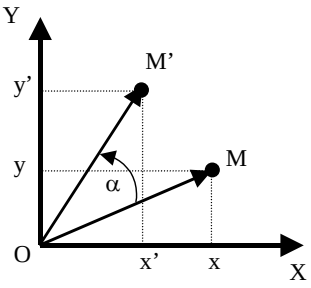
Dans le cas simple d'une transformation 2D, la translation de vecteur $\vec{t}$ d'un point M, s'écrit $\overrightarrow{OM'} = \overrightarrow{OM} + \vec{t}$,



soit en notation matricielle : $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$

En notation matricielle, la rotation α d'un point M, s'écrit :

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \bullet \begin{bmatrix} x \\ y \end{bmatrix}$$



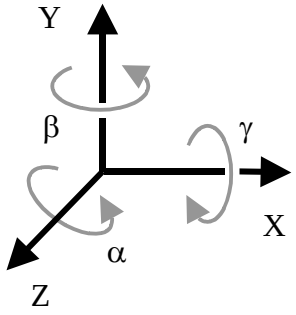
La transformation rigide 2D, composée d'une translation et d'une rotation, s'écrit donc :

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \bullet \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

7.1.3.3.2 Transformation rigide 3D

Dans le cas 3D qui nous intéresse, la transformation rigide se compose de trois translations et d'une décomposition des rotations autour des trois axes.

- L'angle α autour de l'axe Z (de X vers Y) correspond au roulis (roll)
- L'angle β autour de l'axe Y (de Z vers X) correspond au lacet (pan)
- L'angle γ autour de l'axe X (de Y vers Z) correspond au tangage (tilt)



En notation matricielle, la décomposition des rotations s'écrit :

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \bullet \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Le développement de l'expression ci-dessus donne le résultat suivant :

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \cos\alpha\cdot\cos\beta & (\cos\alpha\cdot\sin\beta\cdot\sin\gamma - \sin\alpha\cdot\cos\gamma) & (\sin\alpha\cdot\sin\gamma + \cos\alpha\cdot\sin\beta\cdot\cos\gamma) \\ \sin\alpha\cdot\cos\beta & (\cos\alpha\cdot\cos\gamma + \sin\alpha\cdot\sin\beta\cdot\sin\gamma) & (\sin\alpha\cdot\sin\beta\cdot\cos\gamma - \cos\alpha\cdot\sin\gamma) \\ -\sin\beta & \cos\beta\cdot\sin\gamma & \cos\beta\cdot\cos\gamma \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Cette matrice de transformation est notée R.

Nous pouvons remarquer que cette matrice R est une matrice orthogonale puisque quels que soient les angles α, β et γ, le produit :

$$R^tR = Id$$

où Id est la matrice identité.

Intuitivement, on peut formuler que quelle que soit la rotation que l'on fait subir au trièdre formant le repère, l'orthogonalité des axes formant ce repère est conservée après la transformation. Cette propriété d'orthogonalité de la matrice R, nous permet également d'écrire que la matrice inverse est égale à sa transposée :

$$R^{-1} = R^{t}$$

La transformation rigide 3D s'écrit donc :

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} \cos\alpha\cdot\cos\beta & (\cos\alpha\cdot\sin\beta\cdot\sin\gamma - \sin\alpha\cdot\cos\gamma) & (\sin\alpha\cdot\sin\gamma + \cos\alpha\cdot\sin\beta\cdot\cos\gamma) \\ \sin\alpha\cdot\cos\beta & (\cos\alpha\cdot\cos\gamma + \sin\alpha\cdot\sin\beta\cdot\sin\gamma) & (\sin\alpha\cdot\sin\beta\cdot\cos\gamma - \cos\alpha\cdot\sin\gamma) \\ -\sin\beta & \cos\beta\cdot\sin\gamma & \cos\beta\cdot\cos\gamma \end{bmatrix} \bullet \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

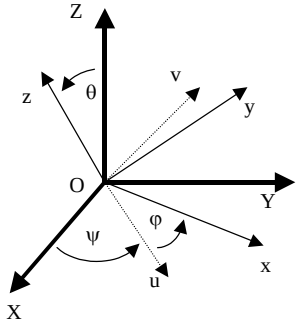
ou de manière simplifiée, en introduisant les coordonnées homogènes :

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

En utilisant la notation $r = \begin{pmatrix} r_1 & r_2 & r_3 \end{pmatrix}$, cette transformation s'écrit :

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} r_1 & t_x \\ r_2 & t_y \\ r_3 & t_z \\ 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

Les trois angles α, β et γ sont dit du cadran. Il ne faut pas les confondre avec les angles d'Euler. Les angles d'Euler permettent également la transformation d'un référentiel OXYZ en référentiel Oxyz. Cependant, les trois angles ψ, θ et φ portent des noms liés à leur application en astronomie.



- ψ est l'angle de précession
- θ est l'angle de nutation
- φ est l'angle de la rotation propre

Cette décomposition fait intervenir un référentiel intermédiaire Ouvz.

Les paramètres extrinsèques sont donc au nombre de 6 : trois angles de rotation α, β, γ et trois translations t_x, t_y, t_z.

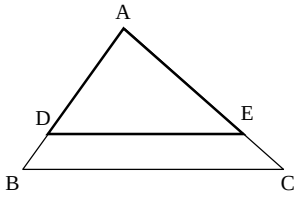
7.1.3.4 Paramètres intrinsèques

Une fois le changement de repère effectué, nous avons besoin de calculer la projection des points du repère caméra sur le plan image. Dans le cas du modèle simplifié, on rappelle que l'axe optique est confondu avec l'axe Z_c et que le plan image est orthogonal à cet axe, c'est-à-dire parallèle au plan X_cY_c. Un calcul supplémentaire permet de convertir les unités métriques du repère de la caméra en unités pixels de l'image. Les paramètres intrinsèques de la camera sont donc les paramètres qui permettent ces transformations.

7.1.3.4.1 Projection perspective

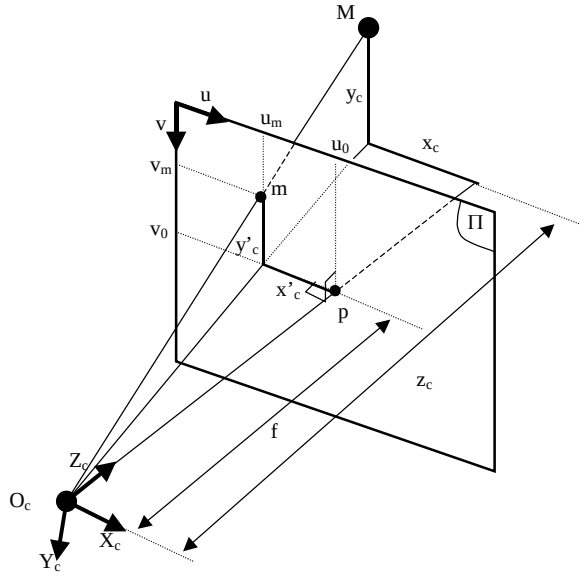
D'après le théorème de Thales, si dans un triangle ABC, D est un point de [AB], E est un point de [AC] et si les segments [BC] et [DE] sont parallèle alors on a la relation :

$$\frac{AD}{AB}=\frac{AE}{AC}=\frac{DE}{BC}$$



Donc d'après Thales :

$$\begin{cases}\frac{x_{\acute{c}}}{f}=\frac{x_c}{z_c}\\ \frac{y_{\acute{c}}}{f}=\frac{y_c}{z_c}\end{cases}\Rightarrow\begin{cases}x_{\acute{c}}=f\frac{x_c}{z_c}\\ y_{\acute{c}}=f\frac{y_c}{z_c}\end{cases}$$

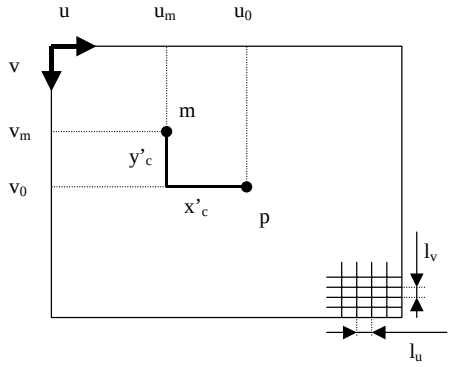


7.1.3.4.2 Changement d'unité

Le changement d'unité permet la transformation de l'unité métrique du repère caméra à l'unité pixelique du repère (u,v) de l'image. Contrairement au repère caméra, le repère de l'image n'est pas homogène du fait de la nature de même de la matrice CCD.

Cette transformation s'écrit simplement :

$$\begin{cases} u_m = u_0 + \frac{x_c}{l_u} \\ v_m = v_0 + \frac{y_c}{l_v} \end{cases}$$



où l_u est la largeur d'un pixel, l_v la hauteur d'un pixel, u_0 et v_0 les coordonnées du point p, centre optique de l'image.

En regroupant les deux expressions, on obtient :

$$\begin{cases} u_m = u_0 + \frac{f}{l_u}.\frac{x_c}{z_c} \\ v_m = v_0 + \frac{f}{l_v}.\frac{y_c}{z_c} \end{cases} => \begin{cases} u_m = u_0 + k_u.\frac{x_c}{z_c} \\ v_m = v_0 + k_v.\frac{y_c}{z_c} \end{cases} \text{ avec } \begin{cases} k_u = \frac{f}{l_u} \\ k_v = \frac{f}{l_v} \end{cases}$$

Les paramètres intrinsèques sont donc au nombre de 4 :

- k_u, la distance focale en pixels horizontaux,
- k_v, la distance focale en pixels verticaux,
- u_0, v_0, les coordonnées du centre optique de l'image.

De façon à rendre homogènes les matrices des paramètres intrinsèques et extrinsèques, les expressions des paramètres intrinsèques peuvent se mettre sous la forme :

$$\begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} k_u & 0 & u_0 & 0 \\ 0 & k_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \bullet \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \text{ avec } \begin{cases} u = \frac{U}{W} \\ v = \frac{V}{W} \end{cases}$$

Soit, en notation simplifiée :

$$\begin{bmatrix} U \\ V \\ W \end{bmatrix} = [A] \bullet \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

où A est la matrice des paramètres intrinsèques.

7.1.3.5 Formulation complète

En regroupant les expressions des paramètres intrinsèques et extrinsèques, la formulation complète s'écrit :

$$\begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} k_u & 0 & u_0 & 0 \\ 0 & k_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \bullet \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

L'expression de la projection perspective s'écrit généralement sous la forme :

$$\begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \bullet \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

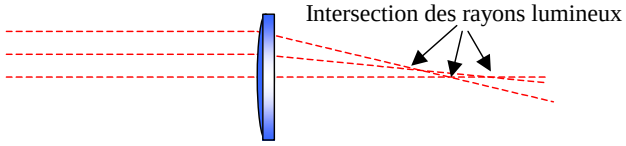
avec :

$$\begin{cases} m_{11} = k_u \cdot \cos\alpha \cdot \cos\beta - u_0 \cdot \sin\beta \\ m_{12} = k_u \cdot (\cos\alpha \cdot \sin\beta \cdot \sin\gamma - \sin\alpha \cdot \cos\gamma) + u_0 \cdot \cos\beta \cdot \sin\gamma \\ m_{13} = k_u \cdot (\sin\alpha \cdot \sin\gamma + \cos\alpha \cdot \sin\beta \cdot \cos\gamma) + u_0 \cdot \cos\beta \cdot \cos\gamma \\ m_{14} = k_u \cdot t_x + u_0 \cdot t_z \\ m_{21} = k_v \cdot \sin\alpha \cdot \cos\beta - v_0 \cdot \sin\beta \\ m_{22} = k_v \cdot (\cos\alpha \cdot \cos\gamma + \sin\alpha \cdot \sin\beta \cdot \sin\gamma) + v_0 \cdot \cos\beta \cdot \sin\gamma \\ m_{23} = k_v \cdot (\sin\alpha \cdot \sin\beta \cdot \cos\gamma - \cos\alpha \cdot \sin\gamma) + v_0 \cdot \cos\beta \cdot \cos\gamma \\ m_{24} = k_v \cdot t_y + v_0 \cdot t_z \\ m_{31} = -\sin\beta \\ m_{32} = \cos\beta \cdot \sin\gamma \\ m_{33} = \cos\beta \cdot \cos\gamma \\ m_{34} = t_z \end{cases}$$

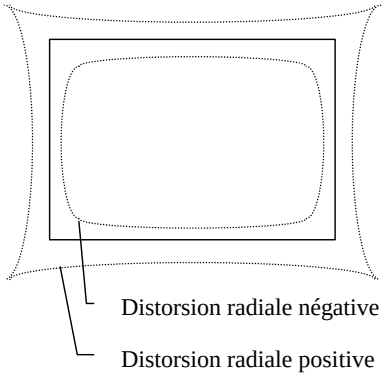
Dans cette expression, on remarque que les coefficients m_{31}, m_{32}, m_{33} et m_{34} ne dépendent pas des paramètres intrinsèques de la caméra.

7.1.3.6 Distorsion de l'image

Dans la plupart des applications, seule la distorsion radiale pose réellement un problème. Cette distorsion de l'image se manifeste généralement par la courbure vers l'extérieur de l'image des droites verticales et horizontales. L'effet de la distorsion est d'autant plus visible que l'on s'éloigne du centre de l'image. L'exemple de distorsion radiale que l'on peut observer facilement est celui du juda des portes d'habitation. Lorsque nous regardons à travers, toutes les lignes droites apparaissent courbes. Ces aberrations sont essentiellement dues à la géométrie de la lentille. Les rayons de lumière qui passent par les bords de la lentille sphériques classiques convergent en un point légèrement décalé par rapport au rayon qui passe par le centre. Ce phénomène bien connu est également appelé « aberration sphérique ».

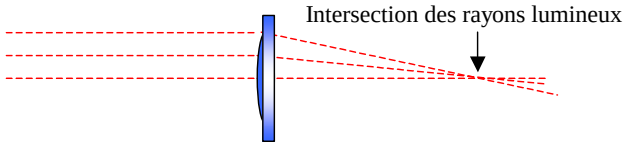


Distorsion radiale avec les lentilles sphériques classiques



Effet de la distorsion radiale sur l'image

Ce type d'aberration peut être corrigé analytiquement. Nous verrons par la suite un modèle mathématique permettant la correction. Cependant, cette correction se fait avec une perte d'information. Une autre solution consiste à diminuer l'angle d'ouverture des lentilles de façon à retomber dans le cas de la dioptrique de Gauss. L'inconvénient est que cette solution diminue la luminosité des images. La plupart des constructeurs proposent aujourd'hui des lentilles asphériques. Une surface non sphérique permet de faire converger les rayons lumineux du bord de la lentille et du centre vers un foyer unique.



Correction de la distorsion avec une lentille asphérique

La distorsion radiale peut être modélisée par un polynôme de degré n. Dans la pratique, on utilise un polynôme de degré 4 en ne gardant que les puissances paires. Cette simplification est réputée pour être suffisante. La formulation de la distorsion radiale est donc la suivante :

$$\begin{cases} \hat{u} = u + (u - u_0)(k_1 \cdot r^2 + k_2 \cdot r^4) \\ \hat{v} = v + (v - v_0)(k_1 \cdot r^2 + k_2 \cdot r^4) \end{cases}$$

où $r = \sqrt{(u-u_0)^2 + (v-v_0)^2}$

7.1.3.7 Formulation complète

Ces aberrations sont essentiellement dues à la géométrie de la lentille. La distorsion peut être modélisée de la façon suivante :

$$\begin{cases} u_m = u_0 + k_u \bullet \dfrac{x_c + \Delta x_c}{z_c} \\ v_m = v_0 + k_v \bullet \dfrac{y_c + \Delta y_c}{z_c} \end{cases}$$

avec :

$$\begin{cases} \Delta x_c = k_1 \cdot x \cdot (x_c^2 + y_c^2) + p_1 \cdot (3x_c^2 + y_c^2) + 2p_2 \cdot x_c \cdot y_c + s_1 \cdot (x_c^2 + y_c^2) \\ \Delta y_c = k_2 \cdot y \cdot (x_c^2 + y_c^2) + p_2 \cdot (3x_c^2 + y_c^2) + 2p_1 \cdot x_c \cdot y_c + s_2 \cdot (x_c^2 + y_c^2) \end{cases}$$

où :

- (k_1, k_2) sont les coefficients de la distorsion radiale,
- (p_1, p_2) sont les coefficients de la distorsion tangentielle due au décentrage de l'axe optique,
- (s_1, s_2) sont les coefficients de la distorsion tangentielle due à la nature du prisme.

90 % de la distorsion est due au paramètre k_1.

7.1.4. Résolution du système linéaire

L'expression de la projection perspective s'écrit sous la forme :

$$\begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \bullet \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

soit en décomposant:

$$\begin{aligned} U &= m_{11} \cdot x_w + m_{12} \cdot y_w + m_{13} \cdot z_w + m_{14} \\ V &= m_{21} \cdot x_w + m_{22} \cdot y_w + m_{23} \cdot z_w + m_{24} \\ W &= m_{31} \cdot x_w + m_{32} \cdot y_w + m_{33} \cdot z_w + m_{34} \end{aligned}$$

sachant que

$$\begin{cases} u = \dfrac{U}{W} \\ v = \dfrac{V}{W} \end{cases}$$

pour chaque point dans le repère du monde, on obtient deux équations :

$$\begin{cases} u = \dfrac{m_{11} \cdot x_w + m_{12} \cdot y_w + m_{13} \cdot z_w + m_{14}}{m_{31} \cdot x_w + m_{32} \cdot y_w + m_{33} \cdot z_w + m_{34}} \\ v = \dfrac{m_{21} \cdot x_w + m_{22} \cdot y_w + m_{23} \cdot z_w + m_{24}}{m_{31} \cdot x_w + m_{32} \cdot y_w + m_{33} \cdot z_w + m_{34}} \end{cases}$$

que l'on peut mettre sous la forme :

$$\begin{cases} m_{11} \cdot x_w + m_{12} \cdot y_w + m_{13} \cdot z_w + m_{14} - u \cdot m_{31} \cdot x_w - u \cdot m_{32} \cdot y_w - u \cdot m_{33} \cdot z_w = u \cdot m_{34} \\ m_{21} \cdot x_w + m_{22} \cdot y_w + m_{23} \cdot z_w + m_{24} - v \cdot m_{31} \cdot x_w - v \cdot m_{32} \cdot y_w - v \cdot m_{33} \cdot z_w = v \cdot m_{34} \end{cases}$$

Pour n points dans le repère du monde, on obtient 2.n équations que l'on peut écrire sous la forme matricielle suivante :

$$\begin{bmatrix} & & & & & & \vdots & & & & \\ x_i & y_i & z_i & 1 & 0 & 0 & 0 & 0 & -u_i \cdot x_i & -u_i \cdot y_i & -u_i \cdot z_i \\ 0 & 0 & 0 & 0 & x_i & y_i & z_i & 1 & -v_i \cdot x_i & -v_i \cdot y_i & -v_i \cdot z_i \\ & & & & & & \vdots & & & & \end{bmatrix} \bullet \begin{bmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{24} \\ m_{31} \\ m_{32} \\ m_{33} \end{bmatrix} = \begin{bmatrix} \vdots \\ u_i \cdot m_{34} \\ v_i \cdot m_{34} \\ \vdots \end{bmatrix}$$

ce qui donne en notation simplifiée :

$$K \cdot M = L$$

où K est une matrice (2n x 11), M est un vecteur de 11 dimensions et L un vecteur de 2n dimensions.

Pour la résolution de ce système linéaire, on fixe le paramètre $m_{34} = 1$. Ce paramètre correspond à la translation tz. Cette simplification est lourde de conséquence puisqu'elle divise tous les paramètres de la matrice K par la valeur réelle de la translation. Les paramètres de la matrice M seront donc calculés à un facteur d'échelle près.

Lorsque le nombre d'équation est supérieur au nombre d'inconnues, la solution de ce système linéaire est la solution au sens des moindres carrés qui est donnée par :

$$M = \left(K^t \cdot K\right)^{-1} \cdot K^t \cdot L$$

Pour résoudre un système à 11 inconnues et sachant qu'avec un point de l'image nous disposons de deux équations, ceci implique que 6 points minimum, tous non coplanaires sont nécessaires.

7.2. Représentations des mosaïques d'images

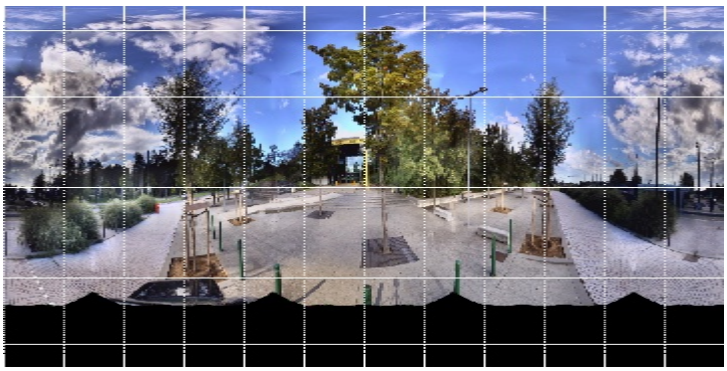
Le monde de la cartographie a développé plusieurs représentations du globe terrestre que nous pouvons utiliser pour représenter une mosaïque d'image. Ces différentes représentations permettent soit de conserver les surfaces (transformation équivalente) soit de conserver les angles (transformation conforme). Dans tous les cas, elles déforment les longueurs. Elles ont été développées pour répondre à un besoin. Par exemple, la navigation exige des cartes dont les directions sont précises alors que le cadastre privilégie les rapports de surface. La cartographie n'entrant pas dans notre domaine de compétence nous nous sommes inspiré d'un ouvrage particulièrement intéressant de Jean Lefort [LEF04], « *L'aventure cartographique* » paru aux éditions « Pour la science ».

7.2.1. Projections par développements

7.2.1.1 Projection équivalente de Lambert

Johan Heinrich Lambert (1728 – 1777) est un mathématicien, physicien et philosophe allemand d'origine française. Ses travaux en mathématiques lui permettent de démontrer l'incommensurabilité du nombre π en 1768. Il pose également les fondements de la trigonométrie sphérique en 1770. Il est connu en géographie pour les modèles de cartes qui portent son nom. Nous présentons ici une de ses projections cylindriques équivalentes dont l'équation est donnée ci-dessous :

$$\begin{cases} X = \theta \\ Y = \sin\varphi \end{cases}$$



Projection équivalente de Lambert

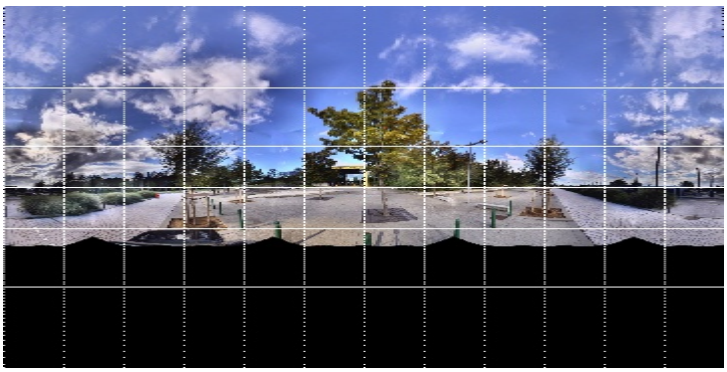
7.2.1.2 Projection conforme de Mercator

Cette projection a été présentée en 1569 par Gerhard Kramer (1512-1594), connu généralement sous le nom de Mercator. Les cartes de Mercator ont été d'un usage universel en navigation durant tout le temps de la marine à voile. Un extrait de la revue générale des sciences pures et appliquées de 1892 [THO92] explique la projection de Mercator en ces termes : « *On peut se représenter la carte de Mercator comme obtenue de la façon suivante : sur toute la surface d'un globe sauf au pôle, on appliquerait une feuille très mince et extensible, telle que serait, par exemple, une pellicule de caoutchouc, si cette substance éminemment extensible pouvait être dépourvue de son*

élasticité; on couperait la feuille suivant un méridien, par exemple celui qui est 180° de Greenwich; puis on étendrait chaque hémisphère dans tous les sens, excepté le long de l'Equateur, jusqu'à rendre tous les cercles de latitude égaux en longueur à la circonférence de l'équateur; suivant le méridien on étendrait la feuille dans le même rapport, de façon à maintenir à angles droits les intersections des méridiens avec les parallèles. La feuille ainsi modifiée, étant posée à plat ou roulée comme une feuille de papier, ce serait là la carte de Mercator ». Plus formellement, la projection conforme de Mercator s'exprime de la façon suivante :

$$\begin{cases} X = \theta \\ Y = \log\left[\tan\left(\frac{\pi}{4}+\frac{\varphi}{2}\right)\right] \end{cases}$$

Nous pouvons remarquer qu'à l'instar de la projection cylindrique, la projection de Mercator ne permet pas la représentation des pôles.

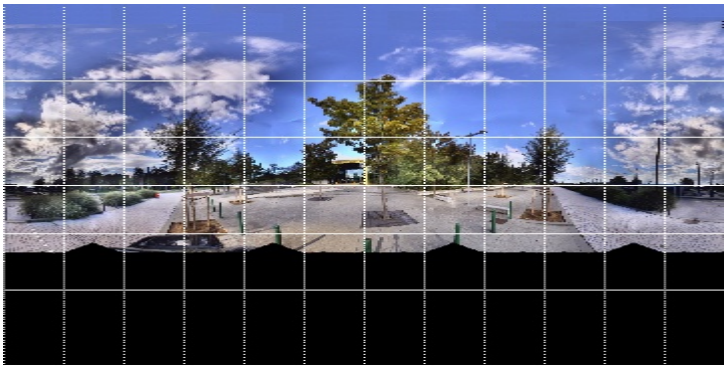


Projection de Mercator

7.2.1.3 Projection de Braun

Par rapport à la projection cylindrique classique, la projection de Braun permet de répondre à la problématique du stockage des pôles. L'équation de cette projection est donnée ci-dessous :

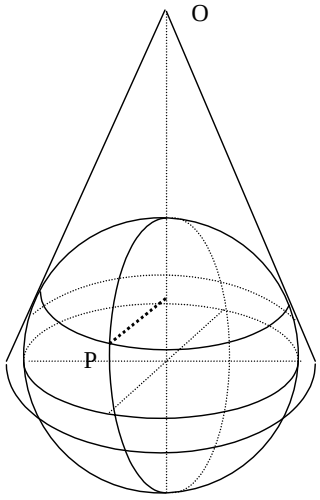
$$\begin{cases} X = \theta \\ Y = 2\cdot\tan\frac{\varphi}{2} \end{cases}$$



Projection de Braun

7.2.2. Projection conique

Les projections coniques correspondent à la projection de la sphère sur un cône tangent à la sphère. Ce n'est pas réellement une projection au sens mathématique. Dans le vocabulaire de la cartographie, une projection conique est une projection qui transforme les méridiens en rayons partant de l'origine O.



Principe de la projection conique

En général, pour les projections coniques, l'expression analytique correspond à une formulation polaire qui correspond au développement du pôle sur le plan :

$$\begin{cases} X = \rho \cdot \sin\phi \\ Y = \rho_0 - \rho \cdot \cos\phi \end{cases}$$

où ρ et ϕ sont des fonctions des coordonnées graphiques.

Les projections coniques n'étant pas particulièrement utilisées pour la représentation des mosaïques d'images, nous présentons simplement deux exemples utilisés en cartographie : la projection conforme de Lambert et la projection de Bonne. Nous avons choisi de représenter la projection de Lambert parce qu'elle est utilisée par l'IGN pour représenter les

cartes de France. Quand à la seconde, nous l'avons simplement choisi pour l'esthétique de la construction en forme de cœur.

7.2.2.1 Projection conique conforme de Lambert

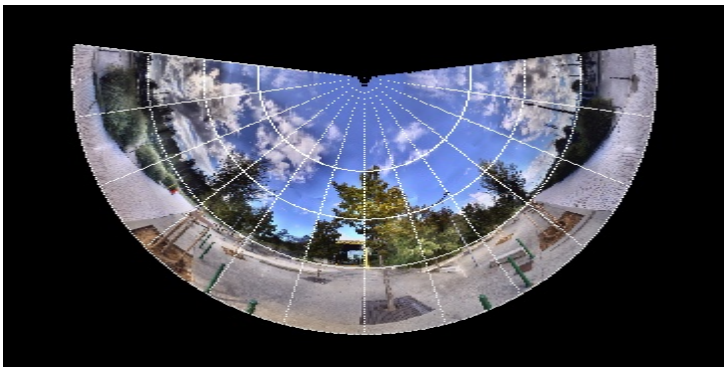
En cartographie, les projections coniques de Lambert sont extrêmement connues, notamment en France. Selon le décret n°2000-1276 du 26/12/2000 publié au Journal officiel n°300 du 28/12/2000, la projection Lambert 93 s'applique au système de référence planimétrique.

Les équations de la projection conique conforme de Lambert sont les suivantes :

$$\begin{cases} \rho = \dfrac{\cos\varphi_1}{n} \cdot \left[\dfrac{\tan\left(\frac{\pi}{4}+\frac{\varphi_1}{2}\right)}{\tan\left(\frac{\pi}{4}+\frac{\varphi}{2}\right)} \right]^n \\ \phi = n \cdot \theta \end{cases}$$

avec

$$n = \frac{\ln\left[\frac{\cos\varphi_1}{\cos\varphi_2}\right]}{\ln\left[\frac{\tan\left(\frac{\pi}{4}+\frac{\varphi_2}{2}\right)}{\tan\left(\frac{\pi}{4}+\frac{\varphi_1}{2}\right)}\right]}$$

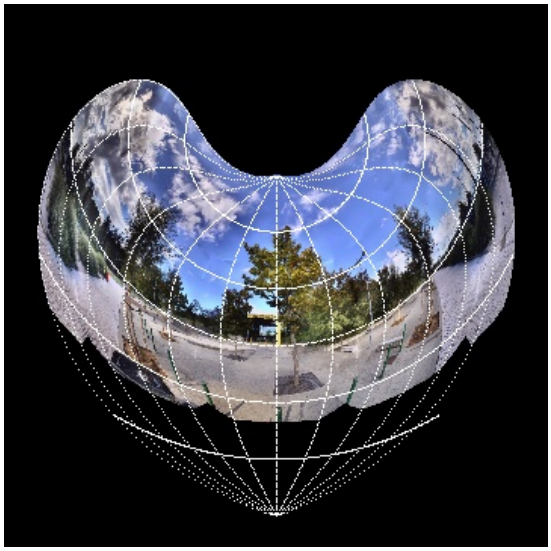


Projection conique conforme de Lambert avec $\varphi_1 = 20°$ et $\varphi_2 = 45°$

7.2.2.2 Projection de Bonne

La projection de Bonne n'est pas réellement une projection conique. Elle est généralement classée dans cette catégorie en raison de sa formulation analytique sous forme polaire. Cette représentation a été utilisée tout au long du XIX^{ième} siècle pour représenter les cartes d'état major. Elle a été remplacée au début du XX^{ième} siècle par la projection conique conforme de Lambert. Les équations de cette projection sont les suivantes :

$$\begin{cases} \rho &= \frac{1}{\tan\varphi_0} - (\varphi - \varphi_0) \\ \phi &= \frac{\theta \cdot \cos\varphi}{\rho} \end{cases}$$



Projection de Bonne

7.2.3. Autres projections planes

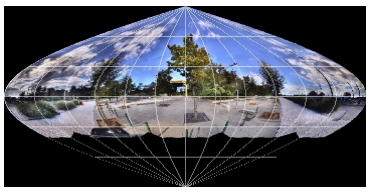
Pour en terminer avec les projections planes, nous présentions ci-dessous quelques constructions, qui ne se classent dans aucunes des catégories précédentes. Nous nous sommes arrêtés sur des critères purement esthétiques et non pour une quelconque utilisation pour le mosaïquage.

Projection Sinusoïdale

Equivalente

Equation :

$$\begin{cases} X &= \theta \cdot \cos\varphi \\ Y &= \theta \end{cases}$$



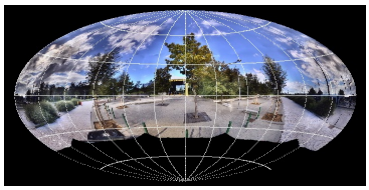
Projection sinusoïdale

Projection de Hammer-Aïtoff

Equivalente

Equation :

$$\begin{cases} X &= 2\sqrt{2} \cdot \dfrac{\cos\varphi \cdot \sin\dfrac{\theta}{2}}{\sqrt{1+\cos\varphi \cdot \cos\dfrac{\theta}{2}}} \\ Y &= \sqrt{2} \cdot \dfrac{\sin\varphi}{\sqrt{1+\cos\varphi \cdot \cos\dfrac{\theta}{2}}} \end{cases}$$



Projection de Hammer-Aïtoff

Projection de collignon

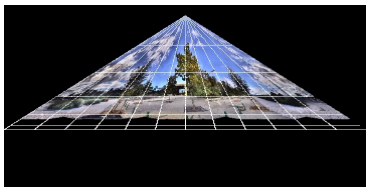
Equivalente

Equation :

$$\begin{cases} X &= \dfrac{2 \cdot \theta s}{\sqrt{\pi}} \\ Y &= \sqrt{\pi} \cdot (1 - s) \end{cases}$$

avec

$$s = \sqrt{1 - \sin\varphi}$$



Projection de Collignon

7.3. Recalage d'images appliqué aux caméras PTZ

7.3.1. Méthodes denses

7.3.1.1 Méthode de Szeliski

Une méthode de recalage classique utilisée pour la construction d'une mosaïque d'images est proposée pour la première fois par Szeliski dans [SZE94]. Cette méthode permet de calculer la matrice d'homographie H d'une image I_1 dans I_0 par itérations successives en recherchant la minimisation d'une fonction de coût. Lorsque certaines conditions sont respectées, la transformation homographique H qui relie une image I_1 à une image I_0 s'exprime avec 8 coefficients. On rappelle que la transformation est définie à un facteur près.

$$x' \sim H \cdot x = \begin{bmatrix} m_0 & m_1 & m_2 \\ m_3 & m_4 & m_5 \\ m_6 & m_7 & 1 \end{bmatrix} \bullet \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

où x est un point de l'image I_1 défini par ces coordonnées homogènes (u, v, 1) dans l'image et x' sa projection homographique dans l'image I_0 et défini par ces coordonnées homogènes (u', v', 1). Le signe ~ indique la relation d'équivalence, à un facteur d'échelle près, entre le point x' et la transformation du point x.

Les coordonnées (u', v') du point x' dans l'image I_1 se déduisent simplement :

$$u' = \frac{m_0 \cdot u + m_1 \cdot v + m_2}{m_6 \cdot u + m_7 \cdot v + 1}, \qquad v' = \frac{m_3 \cdot u + m_4 \cdot v + m_5}{m_6 \cdot u + m_7 \cdot v + 1}$$

Dans l'approche de Szeliski, la matrice H va être mise à jour itérativement à partir d'une matrice de correction D en utilisant l'équation suivante :

$$H \leftarrow (Id + D) H$$

où la matrice de correction D est définie comme suit :

$$D = \begin{bmatrix} d_0 & d_1 & d_2 \\ d_3 & d_4 & d_5 \\ d_6 & d_7 & 0 \end{bmatrix}$$

A chaque itération, Szeliski calcule une nouvelle transformation de l'image I_1 en fonction des paramètres précédemment calculés en utilisant la relation :

$$x' \sim (Id + D) \cdot H \cdot x$$

où x est une pixel de I_1 original et x' le pixel de l'image transformée. Nous pouvons également calculer l'image transformée directement à partir de l'image précédente, c'est-à-dire calculer:

$$\tilde{I}_1(x_i) = I_1(x'_i)$$

à partir de :

$$x'' \sim (Id + D)x$$

en développant l'équation, cela revient à calculer :

$$u'' = \frac{(1+d_0)u + d_1 \cdot v + d_2}{d_6 \cdot u + d_7 \cdot v + 1}, \qquad v'' = \frac{d_3 \cdot u + (1+d_4)v + d_5}{d_6 \cdot u + d_7 \cdot v + 1}$$

Le calcul des paramètres de la matrice de correction s'effectue en recherchant la minimisation d'une fonction de coût. Szeliski propose d'utiliser la fonction de coût suivante :

$$E(d) = \sum_i \left(\tilde{I}_1(x''_i) - I_1(x_i)\right)^2$$

Cette fonction de coût est l'expression de la différence au carré de chaque pixel commun entre l'image I_0 et l'image transformée de I_1 (SSD). Un développement de Taylor de cette expression donne :

$$E(d) \sim \sum_i \left[\tilde{I}_1(x_i) - I_0(x_i) + \nabla \tilde{I}_1(x_i) \frac{\partial x''_i}{\partial d} d\right]^2$$

On note :

- e_i, l'erreur d'intensité à l'ordre 0, soit $e_i = \tilde{I}_1(x_i) - I_0(x_i)$,
- d, le vecteur des coefficients de la matrice D mis en ligne, d = (d_0, d_1, d_3, d_4, d_5, d_6, d_7)^T
- $g_i^T = \nabla \tilde{I}_1(x_i)$, l'image du gradient de l'image transformée de I_1 à x_i,
- $J_i = \frac{\partial x''_i}{\partial d} = \begin{bmatrix} u & v & 1 & 0 & 0 & 0 & -u^2 & -u \cdot v \\ 0 & 0 & 0 & u & v & 1 & -u \cdot v & -v^2 \end{bmatrix}^T$, le jacobien de x_i" par rapport à d

L'expression de la fonction de coût peut donc se réecrire sous la forme :

$$E(d) \sim \sum_i \left[g_i^T \cdot J_i^T \cdot d + e_i\right]^2$$

La solution de ce système peut être déterminer à partir de la méthode des moindres carrés donnés en annexe. L'expression est alors de la forme A.d = b avec :

$$A = \sum_i J_i \, g_i \, g_i^T J_i^T \quad \text{et} \qquad B = \sum_i e_i J_i g_i$$

L'algorithme est finalement assez simple. La matrice H de départ est initialisée soit avec des paramètres approchés de l'homographie par une autre méthode ou par des informations issues de la caméra soit avec la matrice identité. Nous calculons ensuite l'image du gradient, la matrice jacobienne et la matrice de l'erreur d'intensité à partir de l'image I_1 transformé par la matrice H de départ. Si la matrice H de départ est initialisée avec la matrice identité, il n'est bien évidement pas nécessaire de calculer l'image transformée. En utilisation la méthode des moindres carré, ces matrices permettent de déterminer les valeurs de la matrice de correction D que l'on peut appliquer à la matrice H. Par itérations successives, on affine les valeurs de la matrice H. L'algorithme s'arrête sur un critère d'arrêt qui peut être un nombre maximum

d'itérations atteint, un calcul sur la somme ou la somme au carré des valeurs de la matrice d'erreur d'intensité ou encore sur la somme ou la somme au carré des paramètres de la matrice D.

7.3.1.2 Méthode du simplexe

Cette méthode est un grand classique la recherche d'une minimisation dans un espace à plusieurs dimensions. Elle a été introduite par Nelder et Mead en 1965 [NEL65]. Cette méthode repose sur l'utilisation d'un polyèdre de dimension n+1 que nous allons modifier à chaque itération pour se rapprocher du minimum d'une fonction de coût. Dans paragraphe consacré à la limitation de l'espace de recherche, nous avons montré que si les paramètres de prise de vue de l'une des deux images sont connus, l'espace de recherche n'a plus que 3 dimensions. En conséquence, le polyèdre du simplexe possède 4 sommets, c'est à dire un tétraèdre.

L'algorithme du simplexe présenté par Nelder et Mead fait intervenir 4 coefficients qui sont : le coefficient de réflexion ρ, d'expansion χ, de contraction γ et de rétrécissement σ. Le domaine de validité de ces coefficients est donné par :

$$\rho > 0, \quad \chi > 1 \ , \ 0 < \gamma < 1 \text{ et } 0 < \sigma < 1$$

Traditionnellement, on donne implicitement à ces paramètres les valeurs suivantes :

$$\rho = 1, \quad \chi = 2 \ , \ \gamma = ½ \text{ et } \sigma = ½$$

Dans notre application, nous cherchons à recaler le mieux possible deux images. Il nous faut donc une mesure de coût exprimant la différence entre ces deux images. Au regard des tests que nous avons réalisé sur les mesures de similarité, nous avons fait le choix d'utiliser la mesure de corrélation croisée normalisée entre l'image I et l'image transformée de J. Ce que nous pouvons écrire :

$$C(I,J,T)=\frac{\sum_{x\in\Omega} I(x)\cdot J(T(x))}{\sqrt{\sum_{x\in\Omega} I(x)^2}\cdot\sqrt{\sum_{x\in\Omega} J(T(x))^2}}$$

Dans la mesure où tous les pixels sont utilisés pour la fonction de coût, la méthode est bien dans la catégorie des méthodes denses.

La première étape de la méthode du simplexe consiste à initialiser les sommets du polyèdre. Pour chaque sommet k, nous fixons arbitrairement les valeurs θ, φ, f de la transformation homographique, nous calculons les images transformées de J et nous calculons la fonction de coût C_{Sk}. Dans la pratique, les valeurs de θ, φ, f ne sont pas fixées totalement de façon arbitraire puisque nous connaissons une position approchée de la caméra. Nous choisissons donc des points au voisinage de celui donné par la caméra.

Ces points sont alors classés dans l'ordre croissant de leur fonction de coût.

$$C_{s0} \leq C_{s1} \leq \ldots \leq C_{sn}$$

Partant du principe que le sommet dont la fonction de coût est la plus faible est plus proche du minimum que le point dont la fonction de coût est la plus forte, le principe de l'algorithme est de s'éloigner de ce dernier point. Pour cela les auteurs de la méthode proposent d'essayer le point p_0 dans la direction symétrique à Sn par rapport au centre c du polyèdre formé par les autres points. Ce point p_0 est donné par :

$$p_0 = c + \rho(c - S_n)$$

En fonction de la valeur de la fonction de coût C_{p0} de ce point, deux cas peuvent se présenter :

- $C_{p0} \leq C_{s0}$, c'est-à-dire que le point p_0 est plus proche du minimum que le meilleur sommet du simplexe, on essaye donc d'aller plus loin dans cette direction en calculant la fonction de coût C_{p1} du point p_1. Ce point est donné par :

  $$p_1 = c + \chi(c - S_n)$$

  A nouveau, deux cas peuvent alors se présenter :
  - $C_{p1} \leq C_{p0}$, on remplace le sommet S_n par le point p_1. On applique donc une expansion au simplexe
  - $C_{p1} > C_{p0}$, on remplace le sommet S_n par le point p_0. On applique donc une réflexion au simplexe

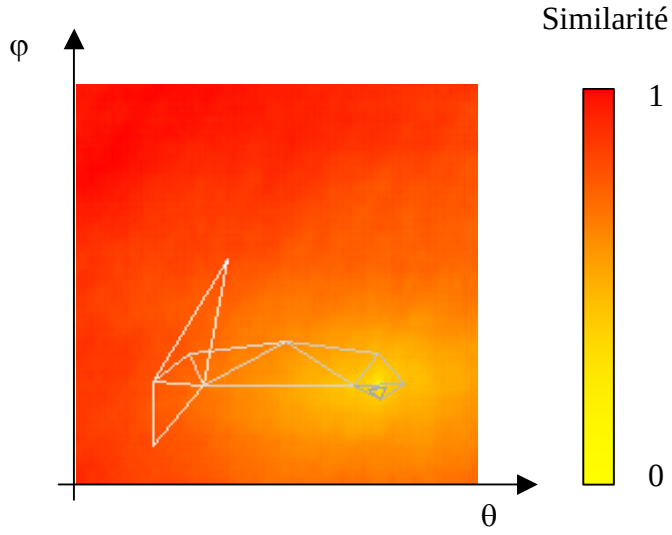
- $C_{p0} > C_{s0}$, on essaie les points p_2 et p_3, placés de part et d'autre de c, dont les coordonnées sont données par :

  $$p_{2,3} = c \pm \gamma(c - S_n)$$

  A nouveau, deux cas peuvent se présenter :
  - si $C_{p2} \leq C_{sn-1}$ ou $C_{p3} \leq C_{sn-1}$, alors on remplace S_n par p_2 ou p_3. On applique alors une contraction au simplexe
  - sinon, le point S_0 étant le point le plus proche du minimum, on rétrécit le simplexe autour de ce point. Les coordonnées de tous les autres points du simplexe sont données par :

    $$S_k \rightarrow c - \sigma(c - S_k) \text{ pour k = 1 à n}$$

Une fois le (ou les) point(s) du sommet recalculé(s), on réordonne les points en fonction de la valeur du coût et on reprend l'algorithme. On arrête les itérations lorsque la taille du simplexe atteint un certain minimum fixé ou que la fonction de coût du sommet le plus faible atteint elle aussi un minimum ou encore lorsqu'un certain nombre d'itérations ont été effectuées.



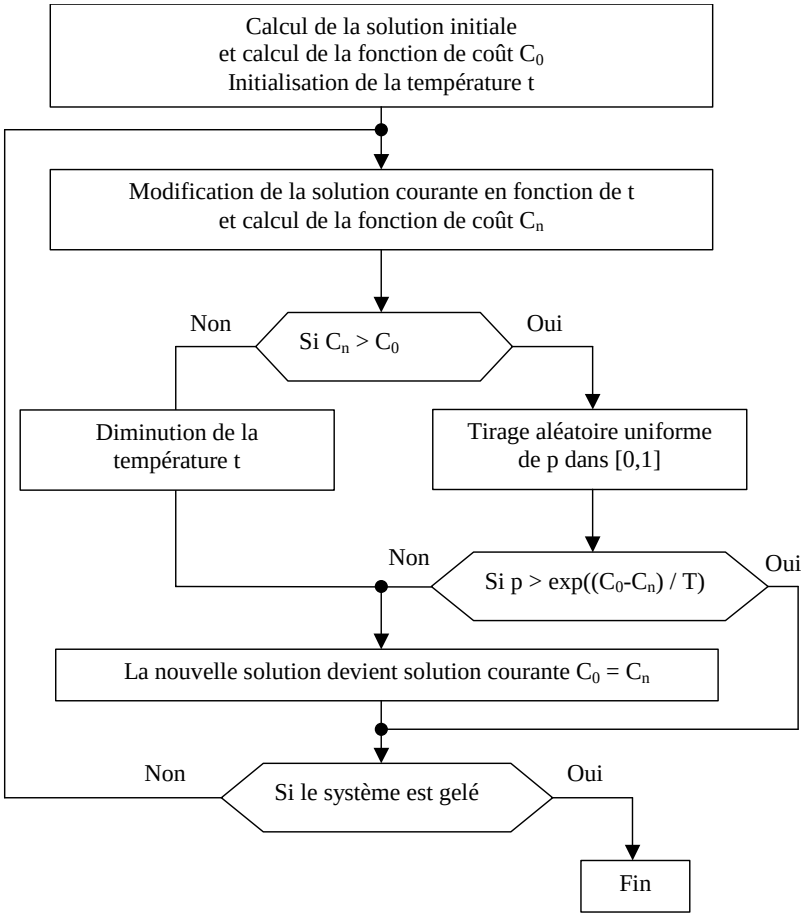
Exemple de progression du simplexe

La méthode du simplexe que nous avons implémenté est beaucoup plus rapide que la méthode que Szeliski. Le temps de calcul est de l'ordre de 300 ms. Elle est surtout plus efficace lorsque les déplacements entre les deux images sont importants. L'utilisation de la mesure corrélation croisée normalisée permet d'éviter à la méthode de rester coincée dans un minimum local.

7.3.1.3 Recuit simulé

Le recuit simulé est une méthode de recherche stochastique qui s'inspire d'un procédé utilisé en métallurgie. Afin d'obtenir un arrangement régulier des atomes à l'état solide, on part de l'état liquide et on refroidit le métal très lentement. Si l'arrangement des atomes n'est pas satisfaisant, on réchauffe le métal sans aller jusqu'au point de fusion, mais suffisamment pour laisser une certaine liberté de mouvement aux atomes. Ce réchauffement est appelé recuit. L'algorithme du recuit simulé s'inspire de cette approche. Le principe consiste à explorer aléatoirement l'espace de recherche autour d'une position courante en fonction d'une « température » t. Plus la température est élevée, plus on s'autorise à explorer loin de la solution courante. Plus la température est basse, plus l'espace de recherche est restreint. A partir d'une température élevée, on diminue progressivement la température à chaque fois que la fonction de coût de la nouvelle solution est meilleure que la solution courante. Cependant, on s'autorise aléatoirement à accepter comme solution courante, une solution dont la fonction de coût est supérieure à la solution courante. Le but est de permettre à l'algorithme de franchir un minimum local. Cet algorithme ne garanti pas d'atteindre le minimum local, mais dans la mesure où la température est élevée, il peut permettre de s'en approcher.

L'organigramme de cet algorithme peut se résumer de la façon suivante :



Organigramme de l'algorithme du recuit simulé.

Comme pour la méthode du simplexe, la transformation T est calculée à partir des trois paramètres θ, φ, f dont nous cherchons l'estimation. Pour générer une transformation à l'itération n, nous modifions les paramètres à partir de l'expression suivante :

$$\theta_n = \theta_{n-1} + a_{\theta n} \cdot E_\theta \cdot t_n$$

où θ_{n-1} est la valeur de l'angle de panorama à l'itération précédente,
$a_{\theta n}$ est une valeur aléatoire comprise entre –1 et 1,
E_θ est fonction de la dynamique de l'espace de recherche,
t_n est la température à l'itération n comprise entre 0 et 1.

Un calcul équivalent est réalisé pour f et φ. L'algorithme se termine lorsque la solution est gelée, c'est à dire lorsque la température atteint un minimum. La condition d'arrêt peut être également liée directement à la mesure de similarité.

L'implémentation de cet algorithme que nous avons testé n'est pas celle qui donne les meilleurs résultats. Cependant, elle reste robuste même pour les déplacements importants. La principale difficulté que nous avons rencontrée correspond à la fonction de décroissance de la température à utiliser. Le temps de calcul est équivalent à la méthode du simplexe.

7.3.1.4 Algorithme génétique

Les algorithmes génétiques sont des algorithmes d'optimisation stochastiques. A l'instar de l'algorithme du recuit simulé, les algorithmes génétiques s'inspirent d'un phénomène naturel. Ici, ce sont les principes d'évolution et de sélection naturelle proposées par Darwin que l'on va chercher à reproduire. Dans la nature, les individus se croisent et se reproduisent en interagissant les uns avec les autres tout en s'adaptant à l'environnement. Les premiers travaux sur les algorithmes génétiques datent des années 1960-1970 avec John Holland. C'est en 1989 que D.E Golberg popularise ces algorithmes [GOL89] en décrivant leur utilisation dans le cadre de la résolution de problèmes concrets. Le champ d'application des algorithmes génétiques est très vaste. On les retrouve principalement dans le cas d'une recherche de minimisation pour laquelle les méthodes basées sur la descente de gradient sont inefficaces. Mais ils sont également utilisés dans le cadre de l'apprentissage supervisé [MIT96], l'estimation du mouvement [ZAI01], la segmentation d'image [BOS01], la compression ou encore des domaines plus éloignés comme la programmation automatique, la théorie des graphes, l'économie ou la finance. Pour Lerman et Ngouenet [LER95], Les algorithmes génétiques diffèrent des algorithmes classiques d'optimisation et de recherche essentiellement en quatre points fondamentaux :

- Les algorithmes génétiques utilisent un codage des éléments de l'espace de recherche et non pas les éléments eux-mêmes.
- Les algorithmes génétiques recherchent une solution à partir d'une population de points et non pas à partir d'un seul point.
- Les algorithmes génétiques n'imposent aucune régularité sur la fonction étudiée (continuité, dérivabilité, convexité…). C'est un des gros atouts des algorithmes génétiques.
- Les algorithmes génétiques ne sont pas déterministes, ils utilisent des règles de transition probabilistes.

Le succès de ces algorithmes est justifié par leur simplicité de mise en œuvre et par leur performance. Ils ne fournissent pas nécessairement la solution globale mais permettent de généralement de s'approcher suffisamment prés de la solution optimale.

Les algorithmes génétiques cherchent donc à reproduire le processus de sélection naturelle à partir de deux constats. Le premier est que seul les individus les mieux adaptés perdure au cours du temps. Le deuxième est que le renouvellement des populations est assuré par ces éléments les plus adaptés par un processus de croisement et de mutation. Ils reposent donc sur 3 grands principes qui sont la sélection, le croisement et la mutation. L'implémentation de ces algorithmes nécessite de définir également un critère de performance (appelé fitness) ainsi que le codage du génotype et le nombre d'individu représentant la population. Le génotype caractérise chaque individu par l'intermédiaire des gènes. Un individu représente donc un point dans l'espace de recherche. Le critère de performance est utilisé pour mesurer l'adaptabilité de l'individu face à l'environnement dans lequel évolue le logiciel. Suivant le type de situation, un même codage du génotype ne donnera pas forcément les mêmes résultats. C'est en fonction de la valeur de ce critère de performance que l'on considère que l'individu est viable et qu'il peut ce reproduire.

Une fois que le critère de performance et que le codage du génotype sont définis, l'implémentation de l'algorithme est assez simple et suit les étapes suivantes :

- 1°) Initialisation de la population. Les gènes de chaque individu sont sélectionnés aléatoirement dans tout l'espace de recherche ou éventuellement dans une zone restreinte de l'espace de recherche en fonction des données dont on dispose. Le but est d'assurer une couverture

la plus complète possible. A chaque individu est associé sa mesure du critère de performance.

- 2°) Sélection. A partir de la mesure du critère de performance, on sélectionne les individus qui maximisent la valeur du critère de façon à générer une population intermédiaire. Cependant, le processus de sélection ne doit pas « tomber » dans l'élitisme. Comme dans la nature, il arrive que des individus décrétés « mauvais » arrivent à survivre. Le processus de sélection doit donc introduire une part d'aléatoire. La solution régulièrement utilisée est la sélection par roulette de casino (wheel selection). Le principe consiste à associer à chaque individu un segment proportionnel à la valeur de son critère de performance. Le segment total est ensuite normalisé. Les individus sont alors sélectionnés par un tirage aléatoire de distribution uniforme compris entre 0 et 1. En règle générale, l'individu qui maximise le critère de performance est systématiquement sélectionné. C'est cette population intermédiaire qui va être utilisée dans les étapes suivantes.
- 3°) Croisement. Le croisement consiste à mélanger les génotypes de deux individus parents de façon à générer un individu enfant. Les parents sont sélectionnés aléatoirement parmi la population intermédiaire issue du processus de sélection. Dans le cas où les gènes sont constitués de valeur réelle, la combinaison des gènes parents est réalisée par interpolation linéaire à partir d'un poids fixé aléatoirement.
- 4°) Mutation. La mutation est appliquée aléatoirement sur certain enfant issu de l'étape croisement. Elle consiste à sélectionner un ou plusieurs gènes par un processus aléatoire et à leur appliquer un bruit gaussien dans la limite de l'espace de recherche. Ce processus de mutation permet d'assurer la propriété d'ergodicité de parcours de l'espace. Le processus ne garantit pas que tout l'espace de recherche sera exploré, mais il offre la possibilité à l'algorithme d'atteindre n'importe quel point de l'espace. Dans certaine configuration, il peut être intéressant d'appliquer une décroissance au bruit gaussien appliqué aux gènes et ainsi limiter l'exploration au cours des itérations successives.
- 5°) A partir de la nouvelle population issue de la sélection, du croisement et de la mutation, on réitère ces trois étapes jusqu'à ce qu'un critère d'arrêt soit satisfait.

Dans notre cas particulier, le génotype de chaque individu est composé de trois gènes. Ces trois gènes correspondent aux trois inconnues (f_J, φ_J et $\Delta\theta_{IJ}$) de la transformation homographique entre deux images. Nous pourrions très bien généraliser l'algorithme aux 6 paramètres de la transformation. Pour le critère de performance, nous utilisons la mesure de similarité et pour le critère d'arrêt nous fixons un seuil sur la mesure de similarité du meilleur individu de chaque génération. Comme évoqué à l'étape 2, nous préservons systématiquement le meilleur individu à chaque génération. Nous assurons ainsi une décroissance monotone de l'algorithme.

Comme pour le recuit simulé, les tests que nous avons effectués avec cette méthode ne sont pas ceux qui donnent les meilleurs résultats, mais la méthode reste robuste pour les déplacements important. L'algorithme est simple à mettre en œuvre. La difficulté cependant et d'une part de fixer la taille de la population et d'autre part de définir la proportion de cette population qui sera sélectionné, muté et croisé. Une population importante assure une convergence plus rapide. Cependant, chaque itération est plus longue à calculer. En revanche, une population faible converge moins rapidement mais est plus rapide à calculer.

7.3.2. Méthodes éparses

7.3.2.1 Détecteur de Harris

La détection de points d'intérêts n'est pas un problème récent et plusieurs solutions existent déjà. Dans [SCH00], le lecteur trouvera la comparaison de plusieurs détecteurs réalisée par Schimd et Mohr. Cependant, l'algorithme le plus cité dans la littérature est celui de Harris [HAR88] qui découle des travaux de Moravec [MOR77]. Ce détecteur s'attache à localiser des points où les variations différentielles sont importantes dans plus d'une direction de façon à ne pas inclure les contours.

L'algorithme de Harris s'appuie sur un filtrage gaussien de l'image et sur le calcul de la matrice M défini comme suit :

$$M=\begin{bmatrix} \left(\frac{\partial I}{\partial x}\right)^2 & \frac{\partial I}{\partial x}\cdot\frac{\partial I}{\partial y} \\ \frac{\partial I}{\partial x}\cdot\frac{\partial I}{\partial y} & \left(\frac{\partial I}{\partial y}\right)^2 \end{bmatrix}=\begin{bmatrix} A & C \\ C & B \end{bmatrix}$$

A partir de cette matrice M, Harris propose de calculer, en chaque point, le coefficient R donné par :

$$R=Det(M)-kTrace(M)^2=(AB-C^2)-k(A+B)^2$$

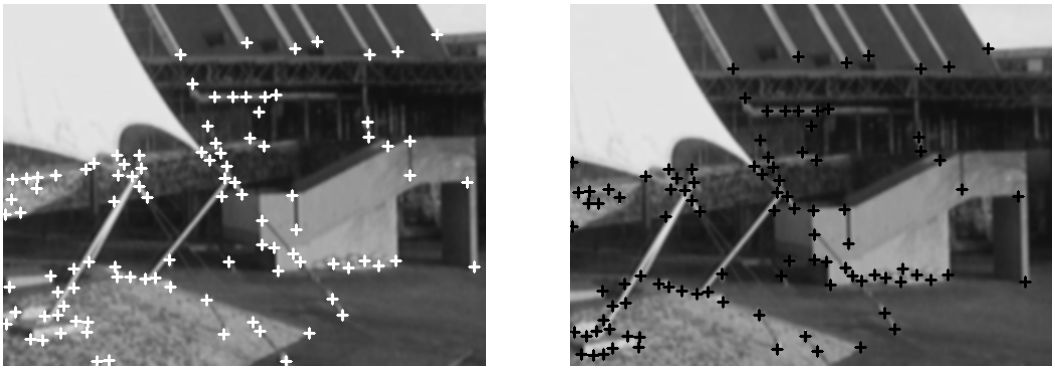
L'interprétation des valeurs des R est la suivante :

- R >> 0 → point d'intérêt
- R << 0 → contour
- R faible → région plate

Les points d'intérêts correspondent aux maxima locaux positifs de ce coefficient.

Cet algorithme fait intervenir le paramètre k. Des études théoriques ont montré que la valeur optimale de ce paramètre est 0.04. Le résultat de la mesure de Harris est cependant très sensible à une faible variation de ce paramètre. Ce détecteur, bien qu'étant le plus utilisé, n'est pas toujours facile à mettre en œuvre, puisque cinq paramètres sont utilisés : le filtre gaussien, le filtre dérivatif, le paramètre k, le voisinage des maxima locaux et le seuil final. Une adaptation de ce détecteur est proposée par Zheng [ZHE99].

A l'issu de cette première étape, nous disposons de deux ensembles de points.



Détection des points d'intérêts dans deux images avec le détecteur de Harris

7.3.2.2 Le descripteur SIFT

Le détecteur SIFT (*Scale Invariant Feature Transform*) proposé par Lowe [LOW04] est considéré aujourd'hui comme l'un des plus performants. L'intérêt de ce détecteur est qu'il est invariant aux transformations affines (changement d'échelle, rotation et translation) et qu'il permet de calculer un descripteur robuste. Ceci le rend particulièrement efficace dans le cas de recalage rigide.

L'algorithme d'extraction de point d'intérêt et le calcul des descripteurs se déroulent en deux étapes :

- Détection de maximum ou minimum locaux dans l'espace d'échelle (scale – space) et localisation des points d'intérêts.
- Choix de l'orientation et calcul des descripteurs

La première étape consiste à détecter les points qui sont stables dans l'espace d'échelle. Pour cela Lowe calcul l'opérateur DoG (Difference of Gaussian) dans l'espace d'échelle puis extrait les extrema locaux. La représentation E dans l'espace d'échelle d'une image I est obtenue par convolution de l'image I avec une gaussienne G(x, y, σ).

$$E(x,y,\sigma)=G(x,y,\sigma)*I(x,y)$$

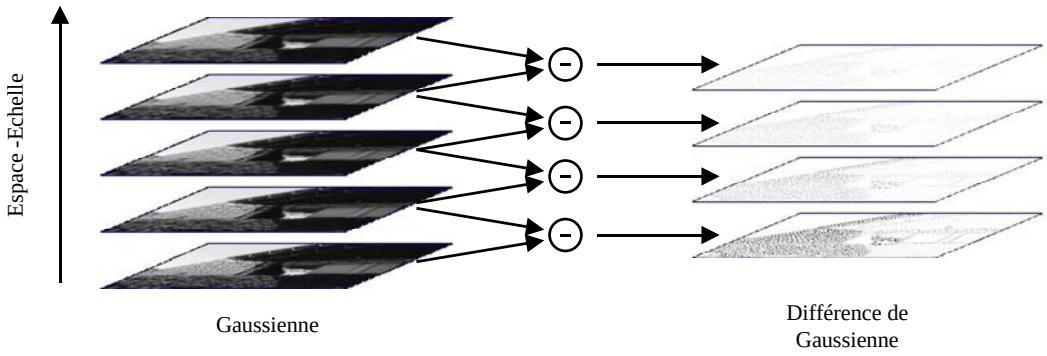
où * est l'opérateur de convolution en x, y et

$$G(x,y,\sigma)=\frac{1}{2\cdot\pi\cdot\sigma^2}\cdot e^{-(x^2+y^2)/2\cdot\sigma^2}$$

L'opérateur DoG s'écrit alors :

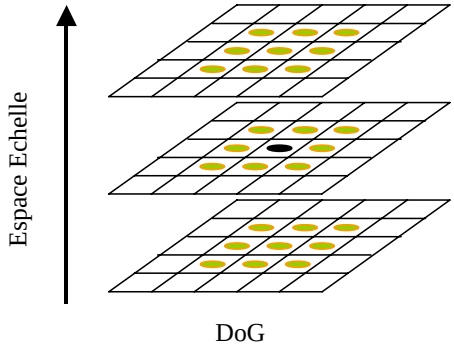
$$DoG(x,y,\sigma)=E(x,y,s\cdot\sigma)-E(x,y,\sigma)$$

où s est une constante multiplicative permettant de définir un nombre d'intervalle par octave (pour passer d'un octave à l'octave voisin la largeur et la hauteur de l'image sont multipliées 2).



Construction de l'opérateur DoG d'après l'article de Lowe [LOW04]

Lowe explique qu'il utilise cet opérateur parce qu'il existe des algorithmes efficaces pour réaliser l'opération de convolution et qu'ensuite il suffit de calculer une différence. Pour chaque octave de l'espace d'échelle, l'image initiale est convoluée par les masques gaussiens pour fournir les images qui seront ensuite soustraite les unes aux autres. La taille de l'image initiale est ensuite divisée par 4 (2x2) afin de calculer l'image initiale de l'octave suivant. L'ensemble des images DoG ainsi calculé permet de constituer la base d'images utilisées pour la recherche des points d'intérêt. Chaque pixel d'une image DoG est comparée avec ces 8 voisins ainsi qu'avec les 9 pixels des images DoG immédiatement supérieure et inférieure dans l'espace d'échelle. Soit en tout 27 comparaisons. Si le pixel est à un minima ou un maxima local, il est un candidat possible. Une phase supplémentaire permet d'éliminer tous les candidats qui sont situé dans des zones insuffisamment contrastées.



Recherche des extrema dans l'espace d'échelle

L'étape suivante consiste à calculer un descripteur local. Le descripteur SIFT est particulièrement robuste aux petits changements de translation, d'illumination et de transformations affines. Le principe retenu par Lowe consiste à calculer un histogramme des orientations du gradient au voisinage du point d'intérêt. La région d'un point d'intérêt est décomposée en 4x4 parties et chacune des sous parties est elle-même décomposée en 8 secteurs angulaires. L'histogramme ainsi obtenu est pondéré par la norme du gradient en ce point. Un point est donc décrit par un vecteur de 4x4x8 soit 128 dimensions. La norme du gradient ainsi que son orientation sont calculées à partir de l'image convoluée par le masque gaussien de l'espace d'échelle le plus grand de façon à garantir l'invariance en échelle. Le calcul de la norme $m(x, y)$ et de l'orientation $\theta(x, y)$ du gradient sont donnés par :

$$m(x,y)=\sqrt{\left(E(x+1,y)-E(x-1,y)\right)^2+\left(E(x,y+1)-E(x,y-1)\right)^2}$$

$$\theta(x,y)=\tan^{-1}\frac{E(x+1,y)-E(x-1,y)}{E(x,y+1)-E(x,y-1)}$$

L'algorithme SIFT est très efficace cependant, le temps de calcul ainsi que l'espace mémoire nécessaire à son exécution sont assez important, en tout cas incompatible avec une application temps réel. Dans son article Lowe indique un temps de calcul de l'ordre de 300ms. L'implémentation que nous avons réalisé donne un temps de calcul de l'ordre la seconde. Cependant, notre but n'a pas été d'utiliser cet algorithme dans notre application mais simplement de disposer d'un algorithme de comparaison reconnu comme efficace et précis par la communauté.

7.3.2.3 Le descripteur « Local jet »

Le descripteur « Local jet » est moins robuste que le descripteur SIFT. Il reste cependant le descripteur le plus utilisé, d'une part parce qu'il a été développé plus tôt et d'autre part parce qu'il est plus rapide à calculer et plus facile à mettre en oeuvre. Le principe de ce descripteur est décrit par Koenderink et van Doorn dans [KOE87]. Il réside dans le fait qu'une fonction peut-être décrite localement par ses dérivées en une série de Taylor. Un point dans une image peut donc être décrit par ses dérivées au voisinage. Le descripteur « Local jet » $J_n(x, y, \sigma)$ est alors défini jusqu'à l'ordre n, pour le point (x, y) avec la taille de gaussienne σ, de la façon suivante :

$$J_n(x,y,\sigma)=\left\{L_{i_1,\ldots,i_k}(x,y,\sigma)/k=0,\ldots,n\right\}$$

où $L_{i1\ldots ik}(x, y, \sigma)$ désigne la dérivée $k^{ième}$ de l'image I par rapport aux variables $i_1, \ldots, i_k$ (x et y dans notre cas). En pratique, une solution simple pour stabiliser le calcul de dérivation consiste à utiliser les dérivées d'une fonction de lissage de type gaussien. La fonction de lissage est :

$$G(x,y,\sigma)=\frac{1}{2\cdot\pi\cdot\sigma^2}\cdot e^{-(x^2+y^2)/2\cdot\sigma^2}$$

et ses dérivées $G_{i1\ldots ik}(x, y, \sigma)$ sont données par :

$$G_{i_1,\ldots,i_k}(x,y,\sigma)=\frac{\partial^n}{\partial_{i_1}\ldots\partial_{i_k}}G(x,y,\sigma) \quad k = 0 \ldots n$$

$L_{i1\ldots ik}(x, y, \sigma)$ est donc la convolution de l'image I avec la différentielle $G_{i1\ldots ik}(x, y, \sigma)$ de la gaussienne $G(x, y, \sigma)$.

$$L_{i_1,\ldots,i_k}(x,y,\sigma)=G_{i_1,\ldots,i_k}(x,y,\sigma)*I(x,y,\sigma)$$

Il est à noter que la taille σ de la gaussienne détermine la quantité de lissage effectué. Ce σ peut aussi être considéré comme un facteur d'échelle. Il est utilisé dans [KOE87] afin de définir un « local jet » multi-échelle (*multi-scale local jet*). Dans [SCH97] les auteurs appliquent une normalisation de la taille de l'image de façon à obtenir des dérivées invariantes en rotation. Leur principal argument est que le calcul des dérivées est influencé par le rapport

largeur/hauteur des pixels de l'image. Ils appliquent donc un facteur de réduction correspondant à ce rapport sur les colonnes de l'image par interpolation linéaire. Dans le même article les auteurs proposent des invariants différentiels basés sur le calcul des « local jet » jusqu'à l'ordre 3.

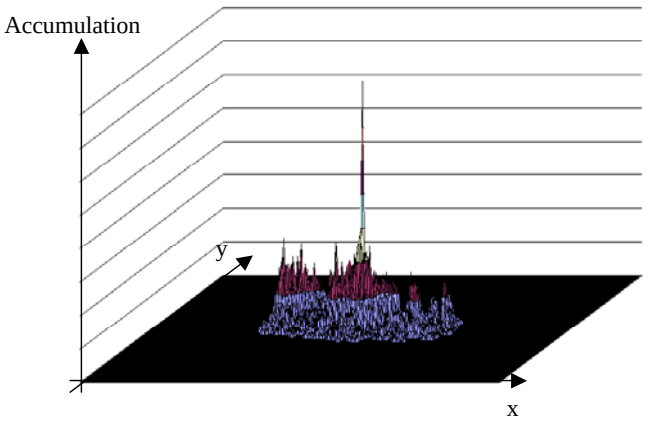
7.3.2.4 Hough

Une autre alternative consiste à utiliser une approche assez répandue en traitement d'image. Elle consiste à rechercher le maximum de répétition d'une occurrence dans un espace d'accumulation. L'espace d'accumulation est communément appelé « transformation de Hough généralisée ». Son application la plus classique est la recherche de courbes paramétrées et plus simplement de droites [HOU62]. Le principe est d'associer à l'image un espace d'accumulation lié aux paramètres de la courbe recherchée. La détection de courbe dans l'image se résume à la recherche de maxima dans l'espace d'accumulation. Dans le cas simple de la recherche de droites, de type $y = ax + b$, dans une image, l'espace des paramètres a deux dimensions qui sont les coefficients a et b de la droite. A un point de coordonnée (x, y) dans l'image, est associé une droite d'équation $b = y – ax$ dans l'espace des paramètres.

Cette transformation trouve également son application dans le cadre du recalage d'image. Dans [SEE02] les auteurs utilisent cette transformation afin d'estimer les paramètres de la transformation affine entre deux images. L'idée de leur algorithme est d'appairer, par une transformation mathématique, des éléments d'informations appartenant aux deux images à recaler à partir de paramètres géométriquement invariables. Par exemple, à partir des coordonnées de deux points dans une image I, il est possible de calculer la distance Euclidienne entre ces deux points. Quel que soit l'angle de rotation de l'image ou sa translation dans les deux axes, cette distance reste la même. Par contre cette distance est sensible au facteur d'échelle. Si nous prenons ces deux même points dans l'image J et que nous calculons la distance, le rapport des deux distances correspond au facteur d'échelle. Le principe de cet algorithme est de calculer les rapports entre la distance de deux points de l'image I et toutes les combinaisons possibles de couples de points dans l'image J et d'incrémenter un vecteur d'accumulation en fonction de la valeur du rapport. Les points étant stochastiquement indépendant, la première passe avec un couple de point de I va remplir le vecteur de façon plus ou moins uniforme. La même opération est réalisée avec tous les autres couples de point de I. La même case de l'accumulateur est incrémentée à chaque fois que les paires de points sont correctement appariées, alors que les autres sont incrémentées de façon aléatoire. Dans [SHA07], les auteurs recherchent également les paramètres de la transformation affine, mais cette fois ci en exploitant les gradients dans les deux images.

Les approches ci-dessus ne sont pas directement transposables dans notre cas, puisque nous sommes dans le cas d'une transformation homographique. Il y a donc une déformation perspective de l'image. Les rapports de distances entre les points ne sont donc pas constant. Cependant, nous pouvons nous placer dans le cas où l'influence de la perspective n'est pas trop importante. Ce sera le cas lorsque les angles de tangage φ_1 et φ_2 sont relativement faibles et que le décalage de l'angle de panorama $\Delta\theta_{21} = \theta_2 - \theta_1$ entre les deux prises de vue est faible également. Les essais que nous avons effectués montrent que pour $-20° < \varphi_{1,2} < 20°$ et pour $\Delta\theta_{21} < 5°$ le décalage des points d'intérêts peut être approximé par deux translations. Nous utilisons donc cette approximation pour réaliser l'appariement des points basée uniquement sur leur position dans l'image. Le principe consiste donc à placer dans un accumulateur de type Hough à deux dimensions, l'écart de position entre toutes les paires de points possibles

entre les deux images. Pour chaque point d'intérêt de l'image I, nous calculons la différence de positionnement avec chacun des points de l'image J et nous plaçons le résultat dans la matrice d'accumulation. Les points d'intérêts dans les deux images étant stochastiquement indépendant, deux points appairés incrémentent toujours la même case de l'accumulateur alors que deux points non appariés incrémentent les cases de façon aléatoire. Après avoir calculé tous les appariements possibles, il nous suffit de rechercher le pic d'accumulation pour déterminer le décalage constant entre les points appariés.



Exemple d'accumulateur de Hough.

Le pic correspond au décalage recherche.

A ce stade nous n'avons que le décalage entre les points, il nous reste encore à former les paires. Il suffit pour cela de vérifier pour chaque point d'intérêt de I si il existe un point d'intérêt de J dans le voisinage en tenant compte du décalage. Une fois que les n paires de points sont identifiés, il ne nous reste plus qu'à résoudre le système de 2n équations par la méthode des moindres carrés. Nous obtenons ainsi directement les paramètres de la transformation homographique entre les deux images.

Une autre solution que nous avons testée, consiste à accumuler les paramètres f_2, θ_2, φ_2 de chaque transformation possible entre 4 paires de points. Le principe consiste à constituer un sous ensemble aléatoire de 4 paires de points, au même titre que l'algorithme RANSAC. A partir de ce sous ensemble nous pouvons calculer la matrice de transformation et, à partir des 9 paramètres de cette matrice, estimer les paramètres f_2, θ_2, φ_2 connaissant les paramètres f_1, θ_1, φ_1 (cf : Angles de rotation à partir de la matrice d'homographie). Les paramètres calculés sont alors placés dans l'accumulateur et nous réitérons le processus avec un autre sous ensemble de points. Nous ne réalisons pas un parcours exhaustif des paires de points possibles. La complexité en $O(n^{16})$ rendrait l'algorithme incompatible avec le temps réel que nous nous sommes fixés. Nous utilisons tout d'abord la base des paires de points possibles que nous avons évoquée au chapitre « liste des points possibles » ce qui limite la complexité. Ensuite nous réalisons un certain nombre de tirage aléatoire avec comme critère de ne pas sélectionner deux points qui soient trop proche l'un de l'autre. A partir d'une centaine de points d'intérêts, les essais que nous avons effectués montrent qu'avec 200 tirages aléatoires, la tendance dans l'accumulateur est suffisamment forte pour une première approximation des paramètres de la transformation. A partir de la cette solution intermédiaire H_2', nous pouvons déterminer précisément l'ensemble des paires de points. C'est à dire les points pour lesquels la distance entre $p_{i,I}$ et $H_2'.p_{j,J}$ est inférieure à un seuil.

$$\left\| p_{i,I} - H'_2 \cdot p_{j,J} \right\| < \varepsilon$$

L'ensemble des paires de points permettant de calculer la solution finale.

Mosaïque d'images multi-résolution et applications

Partie 8 : Bibliographie

8. Bibliographie

[ABI03] Abidi B.R., Koschan A.F., Kang S., Mitckes M., Abidi M.A., Automatic Target Acquisition and Tracking with Cooperative Static on PTZ Video Cameras, Multisensor Surveillance Systems: The Fusion Perspective, G.L. Foresti, C. Regazzoni, P. Varshney (Eds.), Kluwer Academic Publishers, pp. 43-59, 2003.

[AGA05] Agarwal A., Jawahar C.V., Narayanan P.J., A Survey of Planar Homography Estimation Techniques Tech. Rep. IIIT/TR/2005/12. 2005.

[ALK05] Alkaabi, S., Deravi, F., Block matching in rotated images, Electronics Letters Volume 41, Issue 4, pp: 181 – 182, 2005

[BAL07] Balland B., Optique géométrique : Imagerie et instruments, Brochè, ISBN : 978-2-88074-689-6, 2007

[BAN98] Banks J., Bennamoun M., Corke P., Kubik K., Evaluation of new and existing confidence measures for stereo matching, Image and Vision Computing New Zealand (IVCNZ)'98, pp: 252–261, 1998.

[BAN01] Banks J., Corke P., Quantitative Evaluation of Matching Methods and Validity Measures for Stereo Vision, The International Journal of Robotics Research, Vol. 20, No. 7, pp. 512-532, 2001.

[BAR03] Bartoli A., Dalal N., Horaud R., "Motion panoramas", Technical Report RR-4771, INRIA, 2003.

[BAR03-2] Bartoli A., Dalal N., Horaud R., Des séquences vidéo aux panoramas de mouvement, actes Coresa 2003.

[BAS01] Basu A., Baldwin J., A real-time panoramic stereo imaging system and its applications, Panoramic Vision: Sensors, theory, and Applications Springer-Verlag New York, Secaucus, NJ, 123-141, 2001.

[BEN01] R. Benosman, S.B. Kang. Panoramic Vision: Sensors, Theory, Applications. Springer, 2001.

[BER92] J.R. Bergen, P. Anandan, K.J. Hanna, R. Hingorani, Hierarchical model-based motion estimation, Proceedings of the 2nd European Conference on Computer Vision, Santa Margherita Ligure, Italy, pages 237–252, 1992.

[BER01] P. Bertolino, G. Foret, D. Pellerin, Détection de personnes dans les vidéos pour leur immersion dans un espace virtuel, 18ème colloque sur le traitement du signal et des images (GRETSI'01), T2, pages 153-156, Toulouse, France, Sept. 2001.

[BEV05] Bevilacqua, A., di Stefano, L., Azzari, P., An effective real-time mosaicing algorithm apt to detect motion through background subtraction using a ptz camera. (2005) 511-516

[BEV06] Bevilacqua, A., Azzari, P., High-quality real time motion detection using ptz cameras. (2006) 23-23

[BHA00] Bhat, K.S., Saptharishi, M., Khosla, P.K., Motion detection and segmentation using image mosaics, IEEE International Conference on Multimedia and Expo (III). (2000) 1577-1580

[BIS06] A. Biswas, P. Guha, A. Mukerjee, K.S. Venkatesh, Intrusion Detection and Tracking with Pan-Tilt Cameras, IET International Conference on Visual Information Engineering 2006, pp 565-571, 2006.

[BOS01] Lo Bosco, A genetic algorithm for image segmentation, Proceedings. 11th International Conference on Image Analysis and Processing, Volume , Issue , 26-28 Sep 2001 Page(s):262 – 266

[BOU99] J.-Y. Bouguet,, Pyramidal Implementation of the Lucas-Kanade Feature Tracker, OpenCV Documentation, Microprocessor Research Labs, Intel Corporation, 1999

[BRO92] L.G. Brown. A survey of image registration techniques. ACM Computing Surveys, 24(4):325–376, 1992.

[BRO97] M. Bro-Nielsen, Rigid Registration of CT MR and Cryosection Images Using a GLCM Framework, First Joint Conference on Computer Vision, Virtual Reality and Robotics in Medicine and Medical Robotics and Computer-Assisted Surgery, pp171-180, Springer Verlag, 1997

[BRO03] Brown, M., Lowe, D.G., Recognising panoramas, Proceedings of the Ninth IEEE International Conference on Computer Vision, 2003

[BUR06] Burghardt, Tilo Calic, Janko, Real-time Face Detection and Tracking of Animals, Neural Network Applications in Electrical Engineering, pp. 27-32, 2006

[CAN03] Frank M. Candocia, Simultaneous Homographic and Comparametric Alignment of Multiple Exposure-Adjusted Pictures of the Same Scene, IEEE Transactions on Image Processing, vol. 12, n° 12, 2003

[CHA05] Chambon S., Mise en correspondance stéréoscopique d'images couleur en présence d'occultations, Thèse de doctorat, Université Paul Sabatier, Toulouse, 2005.

[CHE95] S.E. Chen, QuickTime VR : An Image-based Approach to Virtual Environment Navigation, SIGGRAPH '95, pp. 29-38, 1995.

[CHE08] Chung-Hao Chen Yi Yao Page, D. Abidi, B. Koschan, A. Abidi, M., Heterogeneous Fusion of Omnidirectional and PTZ Cameras for Multiple Object Tracking, IEEE Transactions on Circuits and Systems for Video Technology, Volume: 18, Issue: 8, pp: 1052-1063, 2008

[CHU04] Albert C. S. Chung, J. Alison Noble, and Paul Summers, Vascular Segmentation of Phase Contrast Magnetic Resonance Angiograms Based on Statistical Mixture Modeling and Local Phase Coherence, IEEE Transaction on Medical Imaging, vol. 23, No. 12, Dec. 2004.

[CLA01] X. Clady, F. Collange, F. Jurie, P. Martinet, Object tracking with a pan-tilt-zoom camera: application to car driving assistance, Proceedings 2001 ICRA., pp : 1653 - 1658 vol.2, 2001

[COH89] L.Cohen, L.Vinet, P.T. Sander et A. Gagalowicz, Hierarchical region based stereo matching in Proceedings of the International Conference on Computer Vision and Pattern Recognition, San Diego CA, pages 416-421, juin 1989

[DAR05] Dar-Shyang Lee, Effective Gaussain Mixture Learning for Video Background Substraction, IEEE Transaction On Pattern Analysis And Machine Intelligence, vol.27, no. 5, May 2005.

[DEM03] Demonceaux C., Kachi-Akkouche D., Optical Flow estimation in omnidirectional images using wavelet approach, Conference on Computer Vision and Pattern Recognition Workshop, Vol. 7, 2003.

[DOU02] Matthijs Douze, Vincent Charvillat, Bernard Thiesse, Des mosaïques plus robustes, plus précises, RFIA'02, Angers, janvier 2002

[DOU03] Matthijs Douze, Philippe Puech, Vincent Charvillat, Jean Conter, Suivi temps-réel de séquences vidéo dans un panoramique pour le codage par objets, Coresa 2003, Lyon, janvier 2003

[FAU93] O. Faugeras, Three-Dimensional Computer Vision: a Geometric View-point, The MIT Press, 1993.

[FIA02] Mark Fiala, Anup Basu, Feature extraction and calibration for stereo reconstruction using non-svp optics in a panoramic stereovision sensor, Proceedings of the 3rd IEEE Workshop on Omnidirectionnal vision, pp 79-86, 2002.

[FIS81] M. A. Fischler and R. C. Bolles, Random sample consensus : a paradigm for model fitting with applications to image analysis and automated cartography, ACM, 24(6) :381-395, 1981.

[FLE91] M.M. Fleck, A topological stereo matcher, The International Journal of Computer Vision, vol 6.3, pp 197-226, 1991

[FUM05] Saito Fumihiko, Registration by Block Matching Based on Weighted Gray-levels Correlation, Journal of the Institute of Image Electronics Engineers of Japan, ISSN:0285-9831, Vol.34, N°5, pp 645-652, (2005)

[GER70] A. Germain, Traité des projections des cartes géographiques, Editeur Arthus Bertrand, ~1870

[GLE03] Gledhill D., Tian G. Y., Taylor D., Clarke D., Panoramic imaging - a review. Computers and Graphics, 27 (3). pp. 435-445, 2003

[GOL89] Goldberg D.E., Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Educational Publishers Inc, ISBN-10: 0201157675, 1989

[GOU00] V. Gouet, Mise en Correspondance d'Images en Couleur : Application à la synthèse de vues intermédiaires, PhD thesis, Université Montpellier II, 2000.

[GRA09] Gracias N., Mahoor M., Negahdaripour S., Gleason A., Fast image blending using watersheds and graph cuts, Image and Vision Computing, Volume 27, Issue 5, Pages 597-607, 2009

[HAN02] Kar-Han Tan, Hong Hua, Narendra Ahuja. Multiview panoramic cameras using a mirror pyramid, In IEEE Workshop on Omnidirectional Vision, 2002

[HAN07] Hannuksela, J. and all., Fast Registration Methods for Super-Resolution Imaging, Finnish Signal Processing Symposium, 2007

[HAR88] Harris C., Stephens M., A Combined Corner and Edge Detector, Proceedings of The Fourth Alvey Vision Conference, Manchester, England (1988) 147-151

[HAR00] I. Haritaoglu, D. Harwood, L.S. Davis, real-time surveillance of people and their activities, IEEE Transaction On Pattern Analysis And Machine Intelligence Vol. 22, pages 809–830, 2000.

[HAR04] Hartley, R.I., Zisserman, A., Multiple View Geometry in Computer Vision. Second edn. Cambridge University Press, ISBN: 0521540518 (2004)

[HEE88] Heeger D.J., Optical flow using spatiotemporal filters, International Journal of Computer Vision, pp 297–302, 1988.

[HIL94] D.Hill, C. Studholme, D. Hawkes, Voxel similarity measures for automated image registration, Visualization in Biomedical Computing, vol. 2359 SPIE Proceedings, pp 205-216, SPIE Press, 1994

[HON91] J. Hong, X. Tan, B. Pinette, R. Weiss, and E.M. Riseman, Image-based homing, IEEE International Conference on Robotics and Automation, pp 620 –625, vol.1, 9-11 April 1991.

[HOR81] Horn-Schunk, Determining Optical Flow, Artificial Intelligence, vol.17, pp 185-204, 1981

[HOU62] HOUGH P. V. C., Method and Means of Recognizing Complex Patterns, U.S. Patent n° 3, 069 654, 18 December 1962.

[HSU96] S. Hsu, P. Anandan, Hierarchical Representations for Mosaic Based Video Compression, Proc. Picture Coding Symp., pp. 395-400, Mar. 1996.

[HU05] Jwu-Sheng Hu, Tzung-Min Su, Robust environmental change detection using PTZ camera via spatial-temporal probabilistic modelling, IEEE International Conference on Mechatronics, pp : 50- 55, 2005.

[HUA98] H.-C. Huang, Y.-P. Hung, Panoramic stereo imaging system with automatic disparity warping and seaming, Graphical Models and Image Processing, 60(3):196–208, May 1998.

[IBN72] Ibn-al-Haitham, Opticea Thesaurus libri septem a Federico Risnero Basilae, Livre 1, Chap. v, pp 7, 1572

[IRA95] M. Irani, P. Anandan and S. Hsu, Mosaic Based Representations of Video Sequences and Their Applications, Proc. of ICCV '95, pp. 605-611, Jun. 1995.

[IRA96] M. Irani, P. Anandan, J. Bergen, R. Kumar, et S. Hsu. Efficient representations of video sequences and their applications, Signal Processing : Image Communication, special issue on Image and Video Semantics : Processing, analysis, and Application, 8(4), 1996.

[JOH98] D. Johnson, Applied Multivariate Methods for Data Analysis. Belmont, CA: Duxbury, 1998.

[KAN99] Kanatani K, Ohta N. Accuracy bounds and optimal computation of homograph for image mosaicing applications. The Seventh International Conference on Computer Vision, 1999.

[KAN03] S. Kang, J. Paik, A. Koschan, B. Abidi, M. A. Abidi, Real-time video tracking using PTZ cameras, Proc. of SPIE 6th International Conference on Quality Control by Artificial Vision, Vol. 5132, pp. 103-111, Gatlinburg, TN, May 2003.

[KIM04] Kyungnam Kim, TH Chalidabhongse, D Harwood, L Davis, Background modeling and subtraction by codebook construction, Image Processing, 2004. ICIP '04. 2004 International Conference on, Vol. 5 (2004), pp. 3061-3064 Vol. 5.

[KOE87] J.J. Koenderink and A.J. Van Doorn. Representation of local geometry in the visual system. Biological Cybernetics, 1987.

[LAL07] M. Lalonde and al., A system to automatically track humans and vehicles with a PTZ camera, Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, 2007.

[LAM03] Chi-Wai Lam, Lai-Man Po, Chun Ho Cheung , A new cross-diamond search algorithm for fast block matching motion estimation, Neural Networks and Signal Processing, 2003. Proceedings of the 2003 International Conference on Volume 2, pp: 1262 - 1265, 2003

[LEE93] C.Y. Lee, D.B. Cooper et D. Keren, Computing correspondence based on region and invariants without feature extraction and segmentation in Proceedings of the International Conference on Computer Vision and Pattern Recognition, pages 655-656, 1993

[LEE99] K.S. Lee, Y.F. Fung, K.H. Wong, S.H. Or, T.K. Lao, Panoramic Video Representation using Mosaic Image , Proc. of CISST'99, pp. 390-396, Las Vegas, USA, June 1999.

[LEF04] Jean Lefort , L'aventure cartographique (Broché), Pour la science, Collection Bibliothèque scientifique, 21 septembre 2004, ISBN-10: 2842450698

[LEM07] Victor Lempitsky, Denis Ivanov, Seamless Mosaicing of Image-Based Texture Maps, cvpr , pp. 1-6, 2007.

[LER95] I.C. Lerman, R.F. Ngouenet, Algorithmes génétiques séquentiels et parallèles pour une représentation affine des proximités, Rapport de recherche de l'INRIA RR2570, Janvier 1995

[LEV04] A. Levin, A. Zomet, S. Peleg and Y. Weiss, Seamless image stitching in the gradient domain. ECCV'04.

[LIU02] Qiong Liu and all, Flyspec: A multi-user video camera system with hybrid human and automatic control, In ACM Multimedia 2002, pp 484—492, 2002

[LUC81] Lucas-Kanade, An iterative Image Registration Technique Width an Application to Stereo Vision, Proceeding of Imaging Understanding Workshop, pp 121-130, 1981

[LUO97] Luong Q.T., Faugeras O., Self-calibration of a moving camera from point correspondences and fundamental matrices, International Journal of Computer Vision, pp 261-289, 1997

[LOW04] Lowe D.G.: Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60(2) (2004) 91-110

[LUC81] B. Lucas and T. Kanade, An iterative image registration technique with application to stereo vision ». In Int. Joint Conf. on Artificial Intelligence, pages 674–679, 1981

[MAI81] Bernard Maitte, La lumière, Points Sciences - Seuil - 1981 - rééd. 1986 – 1990 - 2002

[MAN94] S. Mann and R.W. Picard, Virtual Bellows: Constructing High Quality Stills from Video, ICIP, Vol. 1, pp. 363-367, 1994

[MAN95] S. Mann and R. W. Picard, On being 'undigital' with digital cameras: Extending dynamic range by combining differently exposed pictures, in Proc. of IS&T 48th Annual Conference, May 1995, pp. 422–428.

[MAR88] Mardia K.V., Hainsworth T.J., A Spatial Thresholding Method for Image Segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 10, n6, pages 919-927, Nov 88.

[MIA08] Mian A., Realtime face detection and tracking using a single Pan, Tilt, Zoom camera, Image and Vision Computing New Zealand, pp 1-6, 2008

[MIG05] Joshua Migdal and Tomas Izo and Chris Stauffer, Moving Object Segmentation using Super-Resolution Background Models, Workshop on Omnidirectional Vision, 2005

[MIT96] Melanie Mitchell. An introduction to genetic algorithms. The MIT Press, 1996.

[MIT04] Anurag Mittal and Nikos Paragios, Motion-based background subtraction using adaptive kernel density estimatio, CVPR 2004, pp 302-309, 2004

[MOR77] H. Moravec, Towards automatic visual obstacle avoidance, 5ème Procq. Joint Conf. Artificial Intell., Cambridge, page 584, 1977

[NAG96] Nagai A., Kuno Y., Shirai, Y., Surveillance system based on spatio-temporal information, International Conference on Image Processing, pp. 593 – 596, vol.2, 1996

[NAY97] S.K. Nayar. Omnidirectional video camera, In Proceedings of the 1997 DARPA Image Understanding Workshop, May 1997

[NEG07] Negri P., Clady X., Prevost L., Benchmarking Haar and Histograms of Oriented Gradients features applied to vehicle detection, International Workshop on Intelligent Vehicle Control Systems, 2007

[NEL65] J. A. Nelder and R. Mead, A simplex method for function minimization, Computer Journal 7 (1965), 308 - 313.

[PAV01] I. Pavlidis, V. Morellas, P. Tsiamyrtzis, S. Harp, Urban Surveillance Systems: From the Laboratory to the Commercial World, IEEE Proceedings, Vol. 89, No. 10, pages 1478-1497, Oct. 2001.

[PEE02] P. Peer and F. Solina, Panoramic depth imaging: Single standard camera approach, International Journal of Computer Vision, vol. 47, pp. 149-- 160, 2002.

[PER01] Perner P., Motion Tracking of Animal for Behavior Analysis, Visual Form 2001, lnai 2059, Springer Verlag 2001, pages 779-787, 2001

[RAM05] S. Ramalingam, S. K. Sturm, P.and Lodha. Towards complete generic camera calibration. In Proceedings. IEEE CVPR, 2005.

[REM06] Remondino, F., Fraser, C., Digital camera calibration methods: considerations and comparisons, International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, volume = XXXVI, 2006

[ROB07] L Robinault, S. Bres, S. Miguet, Panoramic mosaicing optimization. ICIAP 07, pp. 548-553. 2007.

[ROB09] L Robinault, S. Bres, S. Miguet., Real time foreground object detection using PTZ camera. VISAPP'09, V1, pp 609-614, 2009.

[ROU98] Joseph O'Rourke, Computational Geometry in C (2nd Ed.). Cambridge University Press, September 1998.

[SAR00] D.Sarrut, Recalage multimodal et plate-forme d'imagerie médicale à accés distant, Thèse de Doctorat, Université Lumière Lyon2, 2000

[SAT04] T. Sato, S. Ikeda, M. Kanbara, A. Iketani, N. Nakajima, N. Yokoya, and K. Yamada, High-resolution video mosaicing for documents and photos by estimating camera motion, Proc. SPIE Electronic Imaging, Vol. 5299, pp. 246-253, Jan. 2004

[SEE02] G. Seedahmed, L. Martucci, Automated Image Registration Using Geometrically Invariant Parameter Space Clustering, ISPRS Commission III, Symposium Sept. 2002, Graz, Austria

[SCH97] C. Schmid, R. Mohr. Local grayvalue invariants for image retrieval. PAMI, 19(5) :530-534, 1997.

[SCH00] C. Schmid, R. Mohr, C. Bauckhage, Evaluation of Interest Points Detectors, International Journal of Computer Vision, vol 37(2), p151-172, 2000

[SHA04] H. Shah, D. Morrell, An Adaptive Zoom Algorithm For Tracking Targets Using Pan-Tilt-Zoom Cameras, in Proc. of International Conference on Acosutics, Speech and Signal Processing, pp : 721-724, 2004.

[SHA07] Shams, R., Barnes, N., and Hartley, R. 2007. Image Registration in Hough Space Using Gradient of Images. In Proceedings of the 9th Biennial Conference of the Australian Pattern Recognition Society on Digital Image Computing Techniques and Applications, pp 226-232, 2007.

[SHI94] J. Shi, C. Tomasi, Good Features to Track, In IEEE Int. Conf. on Computer Vision and Pattern Recognition, pages 593–600, 1994

[SHU97] H. Shum and R. Szeliski, Panoramic Image Mosaics, Technical report number MSR-TR-97-23, Microsoft Research, 1997

[SHU00] H. Shum, R. Szeliski, Construction of Panoramic Image Mosaics with Global and Local Alignment, International Journal of Computer Vision, Vol. 36(2), pp. 101-130, Feb. 2000

[SIN04] Sinha, S., Pollefeys, M., Kim, S.: High-resolution multiscale panoramic mosaics from pan-tilt-zoom cameras. In: Proceedings of the 4th Indian Conference on Computer Vision, Graphics and Image Processing. pp: 28-33, 2004

[SOA01] S. Soatto, H. Jin, P. Favaro, Real-time Feature Tracking and Outlier Rejection with Changes in Illumination, IEEE ICCV'01, 2001.

[SOU96] D. Southwell, A. Basu, M. Fiala, J. Reyda. Panoramic stereo, International Conference on Pattern Recognition, 1996.

[SPI98] Spinei A., Pellerin D., H´erault J., Spatiotemporal energy-based method for velocity estimation », Signal Processing, pp 347–362, 1998.

[STA99] Stauffer C., Grimson W., Adaptative background mixture models for real-time tracking, CVPR99, 1999

[STA00] Stauffer C., Grimson L., Learning Patterns of Activity Using Real-Time Tracking, IEEE Transaction On Pattern Analysis And Machine Intelligence, vol.22, no. 8, Aug. 2000

[STU02] Peter Sturm, Mixing catadioptric and perspective cameras, Proceedings of the 3rd IEEE Workshop on Omnidirectionnal Vision, pages 37-44, June 2002.

[SUG04] Y. Sugaya and K. Kanatani, Extracting moving objects from a moving camera video sequence, Symposium on Sensing via Image Information, pages 279–284, June 2004.

[SZE94] R. Szeliski Image Mosaicing for Tele-Reality Applications, Proc. IEEE Workshop on Applications of Computer Vision, IEEE CS Press, Los Alamitos, Calif., pp. 44-53, 1994.

[SZE97] R. Szeliski, H.-Y. Shum, Creating Full View Panoramic Mosaics and Environment Maps, Computer Graphics (SIGGRAPH 97), pp 251-258, 1997.

[TAN04] Tan K.H, Hua H., Ahuja N., Multiview Panoramic Cameras Using Mirror Pyramids. IEEE Trans. Pattern Anal. Mach. Intell. 26, 7 (Jul. 2004), 941-946, 2004

[THO92] Sir Willian Thomson, Généralisation de la projection de Mercator à l'aide d'instrument électrique, Revue générale des sciences pures et appliqués, T3/N°1-24, 1892

[TOR96] P. Torr and A. Zisserman, MLESAC: A new robust estimator with application to estimating image geometry, Computer Vision and Image Understanding, vol. 78, pp 138-156, 2000

[TOR99] Torralba A.B., H´erault J., An efficient neuromorphic analog network for motion estimation, IEEE Transactions on Circuits and Systems-I, Feb 1999.

[TRI00] Triggs B., McLauchlan P., Hartley R., Fitzgibbon A, Bundle Adjustment : A Modern Synthesis, Vision Algorithms: Theory and Practice, Springer-Verlag, pp 298-372, 2000

[TRU04] Trujillo M., Izquierdo E., A robust correlation measure for correspondence estimation, International Symposium on 3D Data Processing, Visualization and Transmission, 3DPVT, pp 155-162, 2004.

[TSI01] Y. Tsin, V. Ramesh, and T. Kanade. Statistical calibration of ccd imaging process. In Proc. of ICCV, volume 1, pages 480–487, 2001.

[UYT02] M. Uyttendaele, A. Eden, and R. Szeliski, Eliminating ghosting and exposure artifacts in image mosaics, In Proceedings of the Interational Conference on Computer Vision and Pattern Recognition, volume 2, pp 509-516, Kauai, Hawaii, December 2001.

[VIO01] Viola P. Jones M., Rapid object detection using a boosted cascade of simple features, CVPR, volume I, pages 511–518, 2001.

[XIO97] Y. Xiong, K. Turkowski, Creating image based vr using a self-calibrating fisheye lens, In CVPR97, pages 237--243, 1997

[YAG90] Y. Yagi, S. Kawato, Panoramic scene analysis with conic projection In IROS90, 1990.

[ZAI01] M. Zaim, A. El ouaazizi, R. Benslimane, Genetic Algorithms Based Motion Estimation, 2001.

[ZHE99] Zhiqiang Zheng , Han Wang , Eam Khwang Teoh, Analysis of gray level corner detection, Pattern Recognition Letters, v.20 n.2, p.149-162, Feb. 1999

[ZHO04] Yongjin Zhou, Mingxi Wan, Hui Xue, A tunable incremental factor augmented inverse image alignment method in fundus angiogram registration and mosaicing, Computerized Medical Imaging and Graphics, Volume 28, Issue 4, Pages 219-224, 2004

[ZHU99] Zhigang Zhu, Guangyou Xu, Edward M. Riseman, Allen R. Hanson, Fast Generation of Dynamic and Multi-Resolution 360-Degree Panorama from Video Sequences, icmcs, p. 9400, 1999 IEEE International Conference on Multimedia Computing and Systems (ICMCS'99) - Volume 1, 1999

[ZIT05] Zitovà B., Flusser J., Sroubek P., Image Registration: A Survey and Recent Advances, ICIP 2005