

UE Vision, Réalité Augmentée et Applications

Matière : Vision

Parcours Image et Multimédia

Parcours IATI-SIA

3^{ième} année

Introduction au suivi d'objets visuels

Détection et mise en correspondance de points d'intérêt

Sylvie CHAMBON

28 septembre 2023

Remerciements

Ce cours reprend des paragraphes rédigés de ma thèse de doctorat (2005) et également de mon manuscrit d'habilitation (2020). Je remercie donc toutes les personnes qui ont directement ou indirectement contribué à ces deux documents car elles ont également contribué indirectement à la rédaction de ce cours. En particulier, je remercie Alain CROUZIL, mon directeur de thèse, car certains paragraphes sont également inspirés de ses propres notes de cours de vision de 2001-2002 et Guillaume GALÈS, dont j'ai co-encadré la thèse. Certains paragraphes sont également tirés de sa thèse. Je remercie également Marie PELISSIER-COMBESURE qui a relu attentivement ces notes de cours et qui a apporté de nombreuses améliorations. Enfin, merci aux étudiant·e·s des départements SN et EEEA qui proposent régulièrement des améliorations.

Objectifs d'apprentissage et temps de travail

Objectifs d'apprentissage

Les objectifs d'apprentissage de ce cours sont les suivants :

- (1) **Apprendre les notions géométriques fondamentales pour aborder un système de vision :**
 - (a) Connaître le modèle géométrique d'une caméra ;
 - (b) Apprendre (revoir) les principales matrices de transformation pour passer des repères scène \rightarrow caméra \rightarrow image
 - (c) Connaître les techniques classiques de calibrage ;
 - (d) Comprendre et connaître les relations entre caméras d'un capteur stéréoscopique.
- (2) **Découvrir et apprendre les différents éléments liés à la détection de primitives d'intérêt en vision :**
 - (a) Apprendre à identifier les différentes familles de détecteurs telles que proposées dans ce cours : premier ordre, second ordre, basé région, avec ou sans notion de multi-échelle ;
 - (b) Connaître les avantages/inconvénients de chaque approche ;
 - (c) Connaître les principaux descripteurs associés aux détecteurs ;
 - (d) Mettre en place une chaîne complète de détection avec des applications en vision.
- (3) **Découvrir et apprendre différentes techniques de suivi/mise en correspondance :**
 - (a) Apprendre et distinguer les techniques propres au suivi épars ou dense ;
 - (b) Étudier en détails certains algorithmes classiques (SIFT/KLT)
 - (c) Étudier tous les éléments d'une approche par corrélation (connaissance de l'algorithme classique et des mesures les plus courantes).
 - (d) Connaître toutes les contraintes qui peuvent être utilisées pour améliorer le suivi (géométriques ou photométriques).
- (4) **Mettre en œuvre une chaîne de traitement permettant de répondre à un besoin spécifique/une application particulière en vision.** Il s'agit des compétences qu'il faudra développer lors de l'Apprentissage par Problèmes et Projets (APP) et plus précisément, il faudra :
 - (a) Mettre en pratique les compétences acquises en : gestion de projet, communication en groupe ;
 - (b) Apprendre un nouveau mode de travail qui combine autonomie et cadre très précis ;
 - (c) Savoir rédiger des rapports concis et précis sur un projet d'étude ;
 - (d) S'entraîner à extraire les informations pertinentes d'un ensemble de documents (livre, article, vidéo) ;

- (e) Découvrir comment travailler individuellement sur un thème/un sujet d'étude choisi collectivement ;
- (f) Améliorer la pratique pour communiquer les résultats obtenus (à l'écrit et à l'oral) ;
- (g) Savoir prendre du recul face à son travail et à celui des autres ;
- (h) Réutiliser les connaissances acquises au cours de cette UE mais également de tout le cursus à l'N7 pour répondre aux besoins d'une application précise en vision et/ou multimédia.

À la fin de ce cours, ainsi que de l'APP associé, vous devez être capable d'analyser un problème de vision donné en : identifiant les difficultés de la scène, de l'environnement et de la tâche à réaliser et en proposant des solutions que vous savez justifier. Ces solutions doivent s'appuyer sur toutes les briques de bases apprises dans le cours. Durant le semestre, nous essaierons de vous proposer des tests de connaissance afin de tester vos apprentissages. De plus, nous vous fournirons des exemples de sujets d'examens pour vous permettre de juger si vous savez analyser un problème de vision donné.

Temps de travail

Pour les sessions de classe inversée, voici une indication des temps de lecture de chaque partie. Ceci reste une indication, il faut retenir que chaque étudiant·e avance à son rythme et que certaines parties sont plus faciles que d'autres à lire.

- Ce chapitre : consacrer au moins 10 minutes et n'hésitez pas à relire !
- Introduction : 10 minutes
- Partie 2 : 2 ou 3h
- Partie 3 : 2 ou 3h

Table des matières

Table des figures	9
I Introduction	11
II Modèles géométriques	15
1 Modèle géométrique d'une caméra	16
1.1 Présentation du modèle	16
1.2 Notions de coordonnées homogènes	17
1.2.1 Représentation homogène d'une droite	17
1.2.2 Représentation homogène d'un point	17
1.3 Transformation repère scène vers repère caméra	17
1.3.1 Introduction de la transformation	17
1.3.2 Matrice des paramètres extrinsèques	18
1.4 Transformation repère caméra vers repère image	18
1.4.1 Projection perspective	18
1.4.2 Matrice des paramètres intrinsèques	19
1.5 Expression de la matrice de projection perspective	19
1.6 Expression des paramètres à partir de la matrice de projection perspective	20
1.7 Calibrage d'une caméra	20
1.7.1 Estimation des paramètres de la matrice de projection perspective	21
1.7.2 Estimation des paramètres intrinsèques	22
1.7.3 En pratique	22
2 Modèle du capteur stéréoscopique	23
2.1 Présentation du modèle	23
2.2 Géométrie épipolaire	23
2.2.1 Contrainte épipolaire	23
2.2.2 Rectification épipolaire	25
III Détection de points d'intérêt	27
3 Introduction	28
3.1 Définition des primitives d'intérêt	28
3.2 Approches de détection et de caractérisation des points d'intérêt en 2D	33
3.2.1 Approches basées premier ordre	34
3.2.2 Approches basées région	34

3.2.3	Approches basées second ordre	34
3.2.4	Introduction des notions d'analyse multi-échelle et multi-résolution	35
3.2.5	Classement des publications significatives et invariance des détecteurs	36
4	Détecteurs de points d'intérêt	40
4.1	Notations	40
4.2	Construction d'une pyramide d'images	40
4.3	Détecteurs du premier ordre	41
4.3.1	Auto-corrélation [Harris 88]	41
4.3.2	Vers le multi-échelle : Harris-Laplace [Mikolajczyk 04]	43
4.4	Détecteurs de régions	43
4.4.1	SUSAN, <i>Smallest Univalve Segment Assimilating Nucleus</i> [Smith 97]	44
4.4.2	FAST, <i>Features from Accelerated Segment Test</i> [Rosten 06]	46
4.4.3	MSER, <i>Maximally Stable Extremal Regions</i> [Matas 02]	47
4.4.4	IBR, <i>Intensity Based Regions</i> [Tuytelaars 04]	47
4.4.5	Vers le multi-échelle : Détecteur de [Kadir 04]	48
4.4.6	Vers le multi-échelle : EBR, <i>Edge-Based Regions</i> [Tuytelaars 04]	48
4.4.7	Vers le multi-échelle : PCBR, <i>Principal Curvature-Based Regions</i> [Deng 07]	49
4.5	Détecteurs du second ordre	49
4.5.1	Beaudet ou Hessien [Beaudet 78]	50
4.5.2	Vers le multi-échelle : Hessien-Laplace [Mikolajczyk 04]	50
4.5.3	Kitchen et Rosenfeld [Kitchen 82]	50
4.5.4	CSS, <i>Curvature Scale Space</i> [Mokhtarian 98]	50
4.5.5	Vers le multi-échelle : SIFT, <i>Scale Invariant Feature Transform</i> [Lowe 04]	51
4.5.6	Vers une variante accélérée de SIFT : SURF, <i>Speeded Up Robust Features</i> [Bay 08]	54
4.6	Descripteurs de points d'intérêt en 2D	55
4.6.1	Descripteur associé à SIFT et SURF	55
4.6.2	HoG, <i>Histogram of Oriented Gradient</i> [Dalal 05]	57
4.6.3	BRIEF, <i>Binary Robust Independent Elementary Features</i> [Calonder 10]	57
4.6.4	LBP, <i>Local Binary Pattern</i> [Ojala 02]	58
4.6.5	Détecteur s'appuyant sur la covariance [Tuzel 06]	58
IV	Mise en correspondance	59
5	Introduction	60
5.1	Définition	60
5.2	Éléments constitutants de la mise en correspondance	60
5.3	Coût de correspondance	62
5.3.1	Coût des contraintes	62
5.3.2	Coût global développé	63
5.4	Algorithmes de mise en correspondance	63
5.4.1	Approche globale classique	64
5.4.2	Algorithme d'une approche locale classique	64
5.4.3	Appariement en utilisant une segmentation en régions	66
6	Mesures de similarité utilisées dans les approches locales classiques	68
6.1	Définitions	68
6.2	Difficultés	68
6.3	Familles de mesures	69
6.4	Corrélation croisée	71

6.5	Corrélation classique	71
6.6	Corrélation des dérivées : Corrélation des champs de gradients	71
6.7	Mesures non paramétriques	72
6.7.1	ISC, <i>Increment Sign Correlation</i>	72
6.7.2	CENSUS, mesure de recensement	72
6.8	Mesures robustes aux occultations	73
6.9	Illustration des résultats sur images de synthèse et images réelles	74
7	Contraintes pour la mise en correspondance	76
7.1	Introduction	76
7.2	Contraintes liées à la géométrie de la scène	76
7.3	Contraintes liées à la géométrie du capteur	76
7.4	Contraintes liées à la réflectance de la surface des objets et les caractéristiques de la source lumineuse	77
7.5	Prise en compte d'une contrainte	77
7.6	Notations utilisées pour les illustrations	78
7.7	Contrainte d'unicité	78
7.8	Contrainte d'ordre	78
7.9	Contrainte de symétrie	79
7.10	Consistance faible	80
7.11	Synthèse et comparaison de quelques contraintes de compatibilité	80
7.12	Limite du gradient de disparité	81
7.13	Contrainte de rang	82
7.14	Contrainte de continuité figurale	83
7.15	Notion d'ambiguïté et d'imprécision	83
8	Algorithme de mise en correspondance associé au détecteur SIFT	85
8.1	Descripteur	85
8.1.1	Affectation d'une orientation	85
8.1.2	Calcul du descripteur	86
8.2	Appariement	86
9	Suivi de zones d'intérêt par KLT (Kanade–Lucas–Tomasi)	87
9.1	Éléments de bibliographie	87
9.2	Contexte–Introduction	87
9.3	Recalage en 1D et sa solution naïve	88
9.4	Solution en 1D généralisable au 2D	89
9.5	Solution en 2D	90
9.6	Généralisation aux transformations affines	91
9.7	Généralisation aux changements de luminosité	92
9.8	Sélection des zones à suivre	92

Table des figures

1.1	Modèle géométrique de la caméra.	16
2.1	Modèle géométrique du capteur stéréoscopique binoculaire.	24
2.2	Géométrie épipolaire.	26
3.1	Exemples de travaux de recherche sur les primitives d'intérêt en 2D.	28
3.2	Exemples de projets d'enseignement sur les primitives d'intérêt en 2D.	29
3.3	Étapes classiques du fonctionnement d'un détecteur.	31
3.4	Classement des approches de détection de points d'intérêt en 2D.	33
3.5	Illustration de l'introduction d'une analyse multi-échelle dans une pyramide d'images (multi-résolution).	35
3.6	Résultats obtenus avec quatre détecteurs de points d'intérêt en 2D.	39
4.1	Comportement de l'auto-corrélation.	42
4.2	Masque circulaire utilisé par le détecteur SUSAN.	44
4.3	SUSAN : Comportement de la fonction de similarité.	45
4.4	Comportement du détecteur de SUSAN.	45
4.5	Contrainte du centre.	46
4.6	Contrainte de continuité.	46
4.7	Illustration du cercle utilisé pour le détecteur FAST.	47
4.8	Illustration du parallélogramme utilisé dans le détecteur EBR.	49
4.9	Illustration des critères utilisés dans EBR.	49
4.10	Octaves et échelles pour le détecteur SIFT.	52
4.11	Algorithme de détection des points SIFT.	53
4.12	Utilisation de l'image intégrale.	54
4.13	Illustration du <i>Box filtering</i>	55
5.1	Mise en correspondance par corrélation.	65
5.2	Mise en correspondance locale de primitives.	65
5.3	Étapes des méthodes de mise en correspondance fondées sur les régions.	67
6.1	Transformation utilisée par CENSUS.	73
6.2	Principe des mesures robustes.	74
7.1	Contrainte d'unicité.	79
7.2	Contrainte d'ordre.	80
7.3	Contrainte de symétrie.	81
7.4	Comparaison des contraintes.	82
7.5	Contrainte de rang.	83
7.6	Mesure d'ambiguïté.	83
7.7	Mesure d'imprécision.	84

8.1	Histogramme des orientations pour SIFT.	86
9.1	Suivi par KLT.	88

Première partie

Introduction

Ces notes de cours sont la suite de celles fournies pour l'année **2021–2022**, pour l'Unité d'enseignement « Image, Modélisation et rendu »(IMR). Nous utiliserons donc ici les mêmes notations. Alors que l'année dernière nous avons abordés des notions plus proches du traitement d'images bas niveau, cette année nous aborderons des notions de traitement d'images haut niveau comme la **détection de points d'intérêt**. Il s'agit donc de notions plus proches de la vision par ordinateur, comme le **calibrage**, la **mise en correspondance** et le **suivi de points d'intérêt**.

Vision par ordinateur

La vision par ordinateur s'articule autour de trois objectifs principaux :

- La **reconnaissance des formes** – Reconnaître des motifs, des objets, des personnes.
- L'**analyse du mouvement** – Cela peut consister à détecter les mouvements d'un véhicule, d'un personnage, d'une foule.
- La **reconnaissance du relief** – Cela peut consister à mesurer la distance entre un objet et le capteur, la distance entre deux objets, à fournir une carte de profondeur d'une scène entière.

Pour tenter d'atteindre ces objectifs, différentes voies peuvent être suivies. Elles constituent des thèmes de recherche qui se distinguent par les caractéristiques du dispositif, par le type et le nombre d'images fournies par le capteur, par les informations disponibles sur la scène et sur le capteur et par la nature des résultats visés.

L'**analyse du mouvement** se caractérise par un dispositif dans lequel il existe un mouvement relatif entre le capteur et les objets de la scène. Des séquences d'images fournies par le capteur sont analysées afin d'extraire automatiquement des informations sur :

- Le **mouvement perçu** – Cela correspond à la projection du mouvement dans les images. Nous parlons d'analyse du mouvement 2D qui peut, par exemple, passer par :
 - Le **calcul du flux optique** – Estimation des champs des déplacements associés aux pixels des images de la séquence.
 - Le **suivi d'objets** – Calcul des positions d'un ou plusieurs objets dans chaque image de la séquence.
- Le **mouvement réel** – Mouvement 3D du capteur ou des objets de la scène.

La **reconnaissance du relief** concerne l'extraction automatique d'informations sur la structure 3D de la scène à partir d'une ou de plusieurs images. Nous parlons alors de **reconstruction 3D** qui peut être réalisée à partir de **stéréovision monoculaire**, **binoculaire** (abordée dans ce cours) ou **multi-oculaire**, suivant le nombre d'images disponibles. Dans le cadre des cours proposés dans l'« UE Problèmes inverses pour la 3D » vous aborderez des techniques de *Shape From Shading* qui relève de la vision monoculaire mais également des techniques de **stéréophotométrie** à partir de multiples images. Dans le cadre de cette matière, « Vision », nous parlerons de **stéréovision binoculaire**.

Stéréovision binoculaire

Nous pouvons considérer deux cas :

- Les images ont été acquises à deux instants différents (c'est le cas d'images obtenues depuis un satellite, à deux passages différents).
- Les images ont été acquises au même instant ou à des instants suffisamment proches pour qu'aucun changement ne se soit produit dans la scène.

Nous nous plaçons dans le second cas. Nous pouvons distinguer trois étapes fondamentales pour retrouver le relief de la scène :

- Le **calibrage** – Cela consiste à trouver les paramètres des capteurs.
- La **mise en correspondance** ou **appariement** – Le but est de retrouver les points homologues entre les deux images, c'est-à-dire les projections des mêmes points de la scène.

- La **reconstruction 3D** – À partir des paramètres des capteurs et des correspondances de points, un modèle 3D de la scène est reconstruit.

La mise en correspondance s'avère être une tâche délicate dont la qualité du résultat détermine directement la qualité de la reconstruction 3D. Elle rencontre de nombreuses difficultés, notamment en présence de changements de luminosité entre les deux images, d'objets dont la surface est uniforme, d'occultations et de bruit dans les images. En raison de ces difficultés, les méthodes de mise en correspondance sont amenées à exploiter toutes les informations disponibles afin de faciliter la recherche et la détermination des correspondants. Des informations liées à des hypothèses concernant la formation des images sont utilisées pour réduire la taille des zones des images dans lesquelles les correspondants sont recherchés. Il s'agit d'informations qui découlent directement du modèle géométrique du capteur. C'est aussi sur ce modèle géométrique que repose l'étape de triangulation permettant la reconstruction 3D.

Dans ce cours, nous aborderons, en particulier, de manière synthétique le **modèle géométrique de la caméra**, et les **notions de calibrage**, avant de se concentrer sur les aspects de **mise en correspondance**, puis, plus généralement de **suivi de primitives d'intérêt**.

Notions de détection et de suivi

Un domaine de recherche lié à celui de la mise en correspondance concerne, de manière plus général, la détection et le suivi de points d'intérêt. Pour faire du suivi d'objets entre images, il est nécessaire de réaliser les choix suivants, dans le contexte de l'application visée :

- (1) **Éléments à suivre** : points/pixels, régions, objets.

En anglais, nous parlons de *features*, *regions/points of interest*.

- (2) **Critères utilisés pour le suivi** : intensité, couleur, forme.

Nous distinguons ainsi les critères **photométriques** (couleur de l'objet) des critères **géométriques** (forme de l'objet).

- (3) **Hypothèses sur les mouvements 2D apparents**¹ (**caméra, objets, scène**)

- Déplacements (transformations) rigides (conservation des distances et des orientations) : translation, rotation ;
- Déplacements (transformations) non-rigides (non conservation des distances et des orientations) : affine, homographie (effet de zoom, mise à l'échelle), transformations quadratique, polynomiale, à base radiale.

Toutes ses hypothèses permettent de faire une prédiction du mouvement et d'utiliser des approches probabilistes (robustes ou non).

- (4) **Hypothèses sur les déformations** : (objets déformables ou non déformables).

- (5) **Hypothèses sur les variations photométriques** : Au cours du suivi, nous pouvons rencontrer une variation de l'intensité de l'objet suivi, pour les raisons suivantes : mouvement de l'objet, occultations, variation de l'éclairage de la scène (en extérieur car la luminosité varie, de manière générale, lorsque les points de vue sont différents d'une image à l'autre), qualité du capteur. De très nombreuses méthodes font l'hypothèse d'invariance photométrique. Mais, il est possible de prendre en compte un modèle affine et/ou un modèle s'appuyant sur une base d'apprentissage.

- (6) **Méthode de suivi ou choix de la méthode de suivi**

- (a) dense/éparse ;

1. Quand nous observons une scène en 3D, c'est-à-dire dans le monde 3D, nous pouvons visualiser des mouvements. Lorsque cette scène est filmée ou photographiée, nous obtenons une succession d'image 2D (pas de notion de profondeur ou de 3D). Ainsi, nous parlons du fait que nous observons des mouvements 2D apparents dans l'image.

- (b) **éparse suivie de propagation (détection de points d'intérêt, d'objets d'intérêt) ;**
- (c) Par opposition aux méthodes « mixtes » permettant de réaliser suivi et détection simultanément.

Dans le cas de l'analyse de flux vidéo ou de stéréovision binoculaire, il est assez commun et logique d'utiliser un modèle de mouvement faible et d'intensité constante (cela est dû au fait que la vidéo a, en général, une cadence de 24 images par seconde, au minimum, et donc, en pratique, peu de mouvement et de changement d'intensité). Toutefois, dans certaines conditions dégradées, ces hypothèses ne sont plus vérifiées (caméra basse résolution avec une sensibilité faible, mouvements rapides, objets déformables).

Deuxième partie

Modèles géométriques

Chapitre 1

Modèle géométrique d'une caméra

1.1 Présentation du modèle

Le modèle **sténopé**, cf. figure 1.1, ou modèle du « trou d'aiguille » (*pinhole*), est constitué du *plan image*, le plan de projection, et du *centre optique* **O** de la caméra ou centre de projection¹. La droite passant par **O** et perpendiculaire au plan image est l'*axe optique*. L'intersection de cette droite avec le plan image, notée (u_0, v_0) est ce que nous appelons le **point principal**. Le centre optique **O** est situé à une distance f du plan image, c'est ce que nous appelons la **focale**.

Un point **P** de la scène, de coordonnées $(X \ Y \ Z)$, dans le repère caméra, se projette sur le plan image en un point **p**, de coordonnées $(u \ v)$, dans le repère image, qui est l'intersection de la droite (**OP**) avec le plan image. Cette projection peut être représentée par l'équation suivante :

$$\lambda \begin{pmatrix} u & v & 1 \end{pmatrix}^T = \mathbf{M} \begin{pmatrix} X & Y & Z & 1 \end{pmatrix}^T, \lambda \in \mathbb{R}^*, \quad (1.1)$$

où **M** est la matrice de projection perspective qui contient les paramètres du modèle. Le **calibrage** d'une caméra consiste, entre autres, à estimer cette matrice. C'est ce que nous présentons dans la suite de ce chapitre.

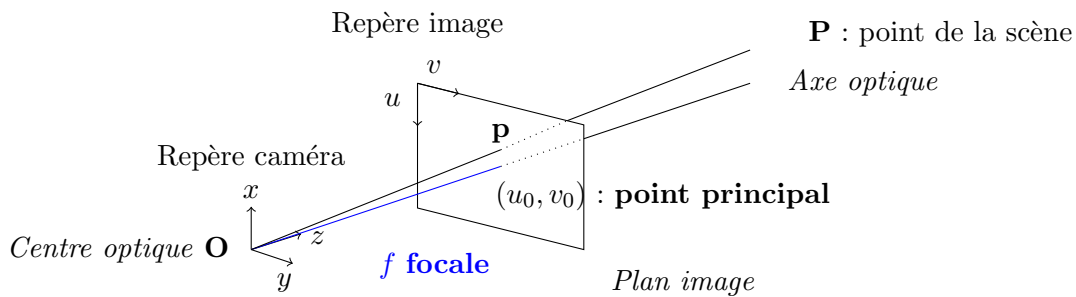


FIGURE 1.1 – Modèle géométrique de la caméra – Nous pouvons noter que le centre optique **O** peut être placé, par rapport à l'axe optique, avant ou après le plan image (c'est-à-dire entre le plan image et la scène). Nous choisissons dans toute la présentation de ce cours de placer le centre optique avant le plan image.

Plus généralement, pour faire correspondre les points objets de la scène avec les points représentant les mêmes points objets dans l'image, il est nécessaire de décomposer le problème en estimation de deux transformations :

1. Sur le schéma, nous avons placé le centre optique « avant » le plan image. Nous pouvons le placer avant ou après, cela n'a pas d'importance. Par convention, sur les représentations, il est placé avant le plan image, pour faciliter la lecture des figures.

1. Transformation du repère scène au repère caméra ;
2. Transformation du repère caméra au repère image.

Ces notions ont déjà été manipulées au cours de l'UE IMR de 2A mais nous les reprenons avec de nouveaux détails dans ce cours.

Cette partie de cours a été rédigée en s'appuyant sur ce livre de référence dans le domaine : [Hartley 04]. Dans la suite, nous utilisons souvent la notion de coordonnées homogènes. En préambule, nous rappelons brièvement à quoi correspondent les coordonnées homogènes².

1.2 Notions de coordonnées homogènes

Un point \mathbf{p} d'un plan est représenté par des coordonnées $(x \ y)$ dans \mathbb{R}^2 . Si nous considérons \mathbb{R}^2 comme un espace vectoriel alors $(x \ y)$ est un vecteur. Ainsi un point est assimilé à un vecteur. Dans la suite nous notons \mathbf{v} un vecteur colonne et \mathbf{v}^T , son transposé, un vecteur ligne. Ainsi, un point dans le plan est représenté par le vecteur colonne $(x \ y)^T$.

1.2.1 Représentation homogène d'une droite

Une droite dans le plan est représentée par l'équation : $ax + by + c = 0$. Ainsi, naturellement, nous pouvons voir qu'une droite peut être représentée par $(a \ b \ c)^T$. Nous pouvons remarquer que les droites $ax + by + c = 0$ et $(ka)x + (kb)y + (kc) = 0$ représentent la même droite, pour tout $k \neq 0$. Ainsi les vecteurs $(a \ b \ c)^T$ et $k(a \ b \ c)^T$ représentent strictement la même droite. Ces deux vecteurs sont donc considérés comme strictement équivalents.

Une classe d'équivalence de vecteurs, respectant cette relation d'équivalence, est appelée vecteur homogène. L'ensemble des vecteurs de cette classe d'équivalence dans $\mathbb{R}^3 - (0 \ 0 \ 0)^T$ définit ce que nous appelons l'espace projectif³ \mathbb{IP}^2 . Comme le vecteur $(0 \ 0 \ 0)^T$ ne correspond à aucune droite, il est naturellement exclu.

1.2.2 Représentation homogène d'un point

Un point $\mathbf{p} = (x \ y)^T$ appartient à une droite $\mathbf{l} = (a \ b \ c)^T$ si et seulement si $ax + by + c = 0$. Cela peut s'écrire également sous la forme d'un produit scalaire : $(x \ y \ 1)(a \ b \ c)^T = (x \ y \ 1)\mathbf{l} = 0$. Cela signifie que le point \mathbf{p} peut également être représenté par un vecteur de taille 3 dont la dernière coordonnée est 1. La notation $\mathbf{p} = (x \ y \ 1)^T$ correspond donc aux **coordonnées homogènes** de \mathbf{p} .

De plus, pour tout $k \neq 0$, nous avons $(kx \ ky \ k)\mathbf{l} = 0$ si et seulement si $(x \ y \ 1)\mathbf{l} = 0$. Ainsi, nous pouvons considérer l'ensemble des vecteurs $(kx \ ky \ k)^T$ comme une représentation du point \mathbf{p} dans \mathbb{R}^3 . Ainsi, comme pour les droites, les points sont représentés par des vecteurs homogènes.

Maintenant que nous avons rappelé cette notion, nous pouvons détailler les différentes transformations nécessaires pour exprimer le passage des coordonnées d'un point dans la scène vers ses coordonnées dans l'image.

1.3 Transformation repère scène vers repère caméra

1.3.1 Introduction de la transformation

Il s'agit d'une transformation rigide, c'est-à-dire faisant intervenir une rotation, représentée par la matrice de rotation \mathbf{R} , et une translation, représentée par le vecteur de translation \mathbf{t} . Nous souhaitons

2. Si vous pensez ne pas avoir besoin de ce rappel vous pouvez passer directement à la section 1.3.

3. Pour comprendre en détails ce qu'est un espace projectif, il faudrait dédier un cours entier à cela. Ce que nous pouvons retenir d'important, c'est qu'il s'agit de l'ensemble des droites vectorielles d'un espace vectoriel et que cela permet de prendre en compte la transformation perspective associée au capteur sténopé.

exprimer les coordonnées $(x \ y \ z)$ d'un point \mathbf{P} , dans le repère caméra, en fonction de ses coordonnées $(X \ Y \ Z)$ dans le repère scène. Cela donne :

$$(x \ y \ z)^T = \mathbf{R}(X \ Y \ Z)^T + \mathbf{t}. \quad (1.2)$$

Nous notons :

$$\mathbf{R} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \quad \text{et} \quad \mathbf{t} = \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix}.$$

1.3.2 Matrice des paramètres extrinsèques

En coordonnées homogènes, la relation (1.2) s'écrit :

$$(x \ y \ z \ 1)^T = \mathbf{A}(X \ Y \ Z \ 1)^T, \quad (1.3)$$

avec :

$$\mathbf{A} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (1.4)$$

La matrice \mathbf{A} est ce que nous appelons dans la littérature : la **matrice des paramètres extrinsèques**. Au total, elle possède 12 paramètres à estimer. Toutefois, la matrice de rotation \mathbf{R} contient 9 éléments, mais la relation qui la définit, à savoir $\mathbf{R} \mathbf{R}^T = 1$ réduit le nombre d'éléments indépendants à 3 : les 3 angles polaires. Ainsi, en ajoutant les 3 paramètres de la translation, cette matrice est formée de 6 paramètres indépendants qui décrivent la position relative de la caméra par rapport au repère scène. Nous pouvons donc supposer que 6 correspondances de points suffisent à estimer ces paramètres. En pratique, afin de réduire les effets de bruits ou erreur sur les correspondances, il est conseillé d'utiliser plutôt une centaine de points, voire plusieurs centaines.

1.4 Transformation repère caméra vers repère image

Nous souhaitons à présent définir la transformation qui permet de passer des coordonnées $(x \ y \ z)$ de \mathbf{P} , dans le repère caméra, aux coordonnées image $(u \ v)$. Cette transformation se décompose en deux transformations :

1. une projection perspective ;
2. une transformation du repère caméra au repère image avec un changement d'unités. Cette dernière partie nous permet d'introduire la matrice des paramètres intrinsèques.

1.4.1 Projection perspective

En reprenant les éléments présentés dans la figure 1.1, et en notant $(x_p \ y_p \ z_p)$ les coordonnées du point \mathbf{p} , du plan image, dans le repère caméra, image de \mathbf{P} , nous pouvons écrire :

$$\begin{cases} x_p &= f \frac{x}{z} \\ y_p &= f \frac{y}{z} \\ z_p &= f. \end{cases} \quad (1.5)$$

Sachant que f représente la focale et que pour retrouver ces relations, il suffit d'utiliser le théorème de Thalès.

1.4.2 Matrice des paramètres intrinsèques

Nous rappelons qu'à présent, nous cherchons à exprimer la transformation du repère caméra au repère image avec changement d'unités. Plus précisément, pour obtenir les coordonnées (u, v) , cf. figure 1.1, il faut calculer :

$$\begin{cases} u &= -x_p + u_0 \\ v &= y_p + v_0 \end{cases} \quad (1.6)$$

Alors que dans le repère caméra les distances sont exprimées en millimètre, dans le repère image, elles sont exprimées en pixels. Il est donc nécessaire d'effectuer un changement d'échelle. De plus, les pixels de l'image ne sont pas forcément carrés et nous avons donc besoin de deux facteurs d'échelle : k_u , facteur vertical, et k_v , facteur horizontal. Leur unité est donc le *pixels/mm*. Le système (1.6) devient donc :

$$\begin{cases} u &= -k_u x_p + u_0 \\ v &= k_v y_p + v_0 \end{cases} \quad (1.7)$$

Ainsi, en combinant (1.5) et (1.7), nous obtenons :

$$\begin{cases} u &= -k_u f \frac{x}{z} + u_0 \\ v &= k_v f \frac{y}{z} + v_0 \end{cases} \quad (1.8)$$

En l'écrivant sous forme matricielle, en coordonnées homogènes, ce système devient :

$$\begin{pmatrix} \lambda u \\ \lambda v \\ \lambda \end{pmatrix} = \begin{pmatrix} -k_u f & 0 & u_0 \\ 0 & k_v f & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad (1.9)$$

Comme annoncé au début de ce paragraphe, cette expression inclut la projection perspective ainsi que le changement d'échelle. Ainsi, les coordonnées cartésiennes (u, v) dans le repère image sont obtenues en divisant le résultat obtenu par λ (la troisième coordonnée). L'équation (1.9) peut s'écrire également :

$$\begin{aligned} \lambda(u \ v \ 1)^T &= \mathbf{K} (x \ y \ z)^T \quad \text{avec} \\ \mathbf{K} &= \begin{pmatrix} -k_u f & 0 & u_0 \\ 0 & k_v f & v_0 \\ 0 & 0 & 1 \end{pmatrix} \end{aligned} \quad (1.10)$$

La matrice \mathbf{K} est appelée la **matrice des paramètres intrinsèques**. Elle contient 4 paramètres : $\alpha_u = -k_u f$, $\alpha_v = -k_v f$, u_0 et v_0 .

1.5 Expression de la matrice de projection perspective

À présent nous pouvons reprendre l'équation (1.2) en utilisant (1.8) et ainsi fournir l'expression permettant de passer des coordonnées $(X \ Y \ Z)$ dans le repère caméra aux coordonnées (u, v) dans l'image :

$$\begin{cases} u &= -k_u f \frac{r_{11}X + r_{12}Y + r_{13}Z + t_x}{r_{31}X + r_{32}Y + r_{33}Z + t_z} + u_0 \\ v &= k_v f \frac{r_{21}X + r_{22}Y + r_{23}Z + t_y}{r_{31}X + r_{32}Y + r_{33}Z + t_z} + v_0 \end{cases} \quad (1.11)$$

Il est possible de réduire au même dénominateur commun et ainsi de faire en sorte que le système (1.11) puisse s'écrire ainsi :

$$\begin{cases} u &= \frac{m_{11}X + m_{12}Y + m_{13}Z + m_{14}}{m_{31}X + m_{32}Y + m_{33}Z + m_{34}} \\ v &= \frac{m_{21}X + m_{22}Y + m_{23}Z + m_{24}}{m_{31}X + m_{32}Y + m_{33}Z + m_{34}} \end{cases} \quad (1.12)$$

Et ainsi, nous retrouvons la forme matricielle introduite au début de ce chapitre, cf. équation (1.1), à savoir :

$$\lambda \begin{pmatrix} u & v & 1 \end{pmatrix}^T = \mathbf{M} \begin{pmatrix} X & Y & Z & 1 \end{pmatrix}^T, \lambda \in \mathbb{R}^*. \quad (1.13)$$

Cette matrice de projection perspective possèdent les propriétés suivantes :

1. Elle est définie à un facteur multiplicatif près.
2. Connaissant cette matrice, nous pouvons estimer les paramètres intrinsèques (matrice \mathbf{K} , équation (1.10)) et extrinsèques (matrice \mathbf{A} , équation (1.4)).
3. Inversement, cette matrice peut être calculée à partir des paramètres, de la manière suivante :

$$\mathbf{M} = (\mathbf{K} \mathbf{0}_{3 \times 1}) \mathbf{A}. \quad (1.14)$$

1.6 Expression des paramètres à partir de la matrice de projection perspective

Nous pouvons noter \mathbf{A} de la manière suivante:

$$\mathbf{A} = \begin{pmatrix} \mathbf{r}_1 & t_x \\ \mathbf{r}_2 & t_y \\ \mathbf{r}_3 & t_z \\ \mathbf{0} & 1 \end{pmatrix} \text{ avec } \mathbf{r}_i = (r_{i1} \ r_{i2} \ r_{i3}). \quad (1.15)$$

On peut également utiliser la notation suivante pour \mathbf{M} :

$$\mathbf{M} = \begin{pmatrix} \mathbf{m}_1 & m_{14} \\ \mathbf{m}_2 & m_{24} \\ \mathbf{m}_3 & m_{34} \end{pmatrix} \text{ avec } \mathbf{m}_i = (m_{i1} \ m_{i2} \ m_{i3}). \quad (1.16)$$

En utilisant la relation (1.14), ainsi que les propriétés d'orthogonalité de la matrice de rotation et le fait que $\alpha_u < 0$, nous obtenons les équations suivantes:

$$\begin{aligned} \mathbf{r}_3 &= \mathbf{m}_3 \\ u_0 &= \mathbf{m}_1 \cdot \mathbf{m}_3 \\ v_0 &= \mathbf{m}_2 \cdot \mathbf{m}_3 \\ \alpha_u &= -\|\mathbf{m}_1 \wedge \mathbf{m}_3\| \\ \alpha_v &= \|\mathbf{m}_2 \wedge \mathbf{m}_3\| \\ \mathbf{r}_1 &= \frac{\mathbf{m}_1 - u_0 \mathbf{m}_3}{\alpha_u} \\ \mathbf{r}_2 &= \frac{\mathbf{m}_2 - v_0 \mathbf{m}_3}{\alpha_v} \\ t_x &= \frac{m_{14} - u_0 m_{34}}{\alpha_u} \\ t_y &= \frac{m_{24} - v_0 m_{34}}{\alpha_v} \\ t_z &= m_{34} \end{aligned} \quad (1.17)$$

1.7 Calibrage d'une caméra

Calibrer une caméra consiste à :

1. Estimer la matrice de projection \mathbf{M} déjà introduite dans la section 1.5. Cette matrice possède 12 paramètres, mais, comme elle est définie à un facteur multiplicatif près, elle décrit réellement 11 paramètres. Ainsi, dans la littérature, l'estimation de ces paramètres correspond à ce qui est appelé un problème de resection ou *resection problem*.
2. Estimer, éventuellement, si nécessaire, les paramètres extrinsèques et intrinsèques de la caméra, soit respectivement les matrices \mathbf{A} et \mathbf{K} .

En général, les méthodes de calibrage proposées consistent à estimer la matrice de projection puis à partir de cette matrice d'estimer les paramètres intrinsèques et extrinsèques.

1.7.1 Estimation des paramètres de la matrice de projection perspective

Il existe de nombreuses approches de calibrage et notamment en faisant intervenir ou non des points connus de la scène. Dans ce cours, nous ne faisons référence qu'aux approches utilisant des correspondance de points connus. Ainsi, si nous supposons que nous connaissons N correspondances entre points $\mathbf{P}_i = (X_i \ Y_i \ Z_i)$ de la scène (dans le repère caméra) et points $\mathbf{p}_i = (u_i \ v_i \ 1)$, avec $i = 1 \dots N$, de l'image, nous pouvons construire le système d'équations suivant :

$$\lambda_i \mathbf{p}_i = \mathbf{M} \mathbf{P}_i. \quad (1.18)$$

Les inconnus sont les 11 paramètres de \mathbf{M} et les N λ_i . De plus, ce système possède $3N$ équations. Ainsi, pour le résoudre nous devons avoir N tel que :

$$3N \geq 11 + N$$

Ce qui donne que pour résoudre ce système il faut : $N \geq 6$.

Une méthode simple pour résoudre ce système d'équations est la méthode de la transformation linéaire directe ou *Direct Linear Transformation* (DLT). Si nous notons :

$$\mathbf{M} = \begin{pmatrix} \mathbf{m}_1^T \\ \mathbf{m}_2^T \\ \mathbf{m}_3^T \end{pmatrix}$$

Le système (1.18) peut aussi s'écrire, avec $\mathbf{0}_{1 \times 4} = \mathbf{0}$:

$$\begin{pmatrix} \mathbf{P}_i & \mathbf{0} & \mathbf{0} & -u_i \\ \mathbf{0} & \mathbf{P}_i & \mathbf{0} & -v_i \\ \mathbf{0} & \mathbf{0} & \mathbf{P}_i & -1 \end{pmatrix} \begin{pmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \\ \mathbf{m}_3 \\ \lambda_i \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \quad (1.19)$$

Et en développant l'équation (1.19), pour les N correspondances, nous obtenons :

$$\underbrace{\begin{pmatrix} \mathbf{P}_1 & \mathbf{0} & \mathbf{0} & -u_1 & \dots & \dots & \dots \\ \mathbf{0} & \mathbf{P}_1 & \mathbf{0} & -u_1 & \dots & \dots & \dots \\ \mathbf{0} & \mathbf{0} & \mathbf{P}_1 & -1 & \dots & \dots & \dots \\ \mathbf{P}_2 & \mathbf{0} & \mathbf{0} & 0 & -u_2 & \dots & \dots \\ \mathbf{0} & \mathbf{P}_2 & \mathbf{0} & 0 & -u_2 & \dots & \dots \\ \mathbf{0} & \mathbf{0} & \mathbf{P}_2 & 0 & -1 & \dots & \dots \\ \mathbf{P}_3 & \mathbf{0} & \mathbf{0} & 0 & 0 & -u_3 & \dots \\ \mathbf{0} & \mathbf{P}_3 & \mathbf{0} & 0 & 0 & -u_3 & \dots \\ \mathbf{0} & \mathbf{0} & \mathbf{P}_3 & 0 & 0 & -1 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}}_{=\mathbf{D}} \underbrace{\begin{pmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \\ \mathbf{m}_3 \\ \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \vdots \end{pmatrix}}_{=1} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \end{pmatrix}. \quad (1.20)$$

Nous cherchons donc à estimer un vecteur non nul du noyau de la matrice \mathbf{D} . De plus, étant donné que l'échelle peut être choisie arbitrairement, nous pouvons ajouter la contrainte suivant : $\|\mathbf{l}\|^2 = 1$. Enfin, étant donné qu'il n'y a pas une solution exacte à ce système, pour le résoudre, il est adapté de résoudre :

$$\min_{\|\mathbf{l}\|^2=0} \|\mathbf{D}\mathbf{l}\|^2. \quad (1.21)$$

Il s'agit donc d'effectuer une estimation au sens des moindres carrés. À ce stade, il est important de noter que l'équation (1.21) a deux solutions au moins puisque $\|\mathbf{D}\mathbf{l}\| = \|\mathbf{D}(-\mathbf{l})\|$ et $\|\mathbf{l}\| = \|-\mathbf{l}\|$. Toutefois, il s'agit de la même projection mais une des solutions placent la caméra derrière la scène,

ce qui engendre des profondeurs négatives. Pour éviter ce problème, il faut alors choisir la solution où tous les λ_i sont positifs.

La solution peut être ainsi retrouvée en utilisant les valeurs propres. Plus précisément, il suffit de prendre le vecteur propre ayant la plus petite valeur propre de la matrice $\mathbf{D}^T \mathbf{D}$. Étant donné la forme de cette matrice, il est possible d'utiliser l'algorithme de décomposition en valeurs singulières, *Singular Value Decomposition* (SVD) [Abdi 07].

1.7.2 Estimation des paramètres intrinsèques

Une façon d'extraire les paramètres de la matrice \mathbf{K} est d'utiliser une factorisation QR ou *QR factorization*. Nous avons :

$$\mathbf{M} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$$

avec \mathbf{K} une matrice triangulaire et \mathbf{R} une matrice de rotation. Il est alors possible d'appliquer le théorème de factorisation QR. Ce théorème est le suivant : Si \mathbf{B} est une matrice carrée $n \times n$ alors il existe une matrice orthogonale \mathbf{Q} et une matrice triangulaire droite \mathbf{R} . Ainsi, dans notre contexte de calibrage, nous posons $\mathbf{Q} = \mathbf{R}$ et $\mathbf{R} = \mathbf{K}$. En conséquence, nous obtenons : $\mathbf{A} = \mathbf{R}\mathbf{K}$.

1.7.3 En pratique

Nous rappelons que tout le processus d'estimation que nous venons d'expliquer ne fonctionne que si nous possédons des correspondances entre points de la scène et points de l'image. Ainsi, pour effectuer le calibrage, il faut établir ces correspondances fiables. Pour cela, il est possible d'utiliser une mire ou des marqueurs particuliers ou d'utiliser des détections de points d'intérêt ou d'autres primitives très fiables pour réaliser le calibrage. En Travaux Pratiques (TP), dans la partie réalité augmentée, vous allez suivre les étapes de calibrage s'appuyant sur une mire en suivant cette chaîne de traitement :

- Utilisation de photos acquises en présence d'une mire : ici un damier (mais il y a plein d'autres possibilités) ;
- Détection de points d'intérêt de cette mire : les coins du damier, car ils sont bien contrastés ;
- À partir des correspondances de points, estimation de la matrice \mathbf{M} , cf. section 1.7.1 ;
- À partir de \mathbf{M} , estimer \mathbf{A} et \mathbf{K} , cf. section 1.7.2.

Chapitre 2

Modèle du capteur stéréoscopique

2.1 Présentation du modèle

Dans le cas particulier de la stéréovision binoculaire, deux images sont prises depuis deux points de vue différents. Pour cela, soit deux caméras sont utilisées, soit la caméra est déplacée entre les deux prises de vue (dans ce cas, les objets doivent être immobiles). La position relative des deux prises de vue doit être choisie de telle sorte qu'une grande partie de la scène soit visible dans les deux images. La position de la deuxième prise de vue ne doit pas être obtenue par rotation autour du centre optique de la caméra, car, dans ce cas, la reconstruction 3D est impossible. Les paramètres du modèle géométrique du capteur stéréoscopique binoculaire, cf. figure 2.1, peuvent être représentés par les deux matrices de projection perspectives associées aux deux images.

2.2 Géométrie épipolaire

Cette partie n'est pas un cours sur la géométrie épipolaire mais simplement l'introduction de la contrainte dont nous reparlerons dans le chapitre 7. Et ensuite, nous introduisons la notion de rectification épipolaire.

2.2.1 Contrainte épipolaire

La figure 2.2 résume la géométrie épipolaire. Le point \mathbf{P} de la scène et les centres de projection des deux caméras, \mathbf{O}_1 et \mathbf{O}_2 , définissent le plan épipolaire. Les intersections de ce plan avec les deux plans images sont deux droites appelées épipolaires conjuguées.

La contrainte épipolaire est donc définie par :

Le correspondant \mathbf{p}_2 d'un point \mathbf{p}_1 , qui se trouve sur la droite épipolaire gauche \mathcal{D}_1 , se trouve nécessairement sur la droite \mathcal{D}_2 , image de \mathcal{D}_1 dans le plan droit. La droite \mathcal{D}_2 est appelée la droite épipolaire droite associée au point \mathbf{p}_1 .

Les droites épipolaires possèdent la propriété suivante :

Toutes les droites épipolaires concourent en un point appelé l'épipôle, noté \mathbf{e}_l . Cette épipôle est également l'image du centre de projection de l'autre image.

Les deux épipoles sont simplement l'intersection de la droite formée par les deux centres des caméras et les plans de formation des images de chaque caméra.

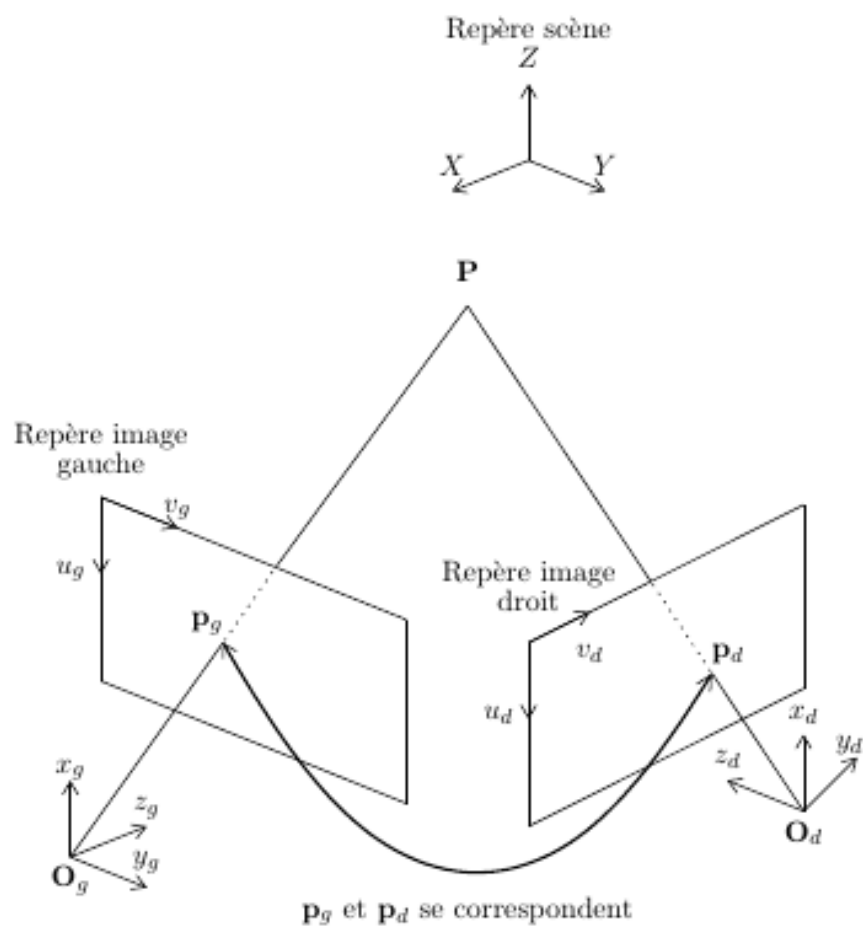


FIGURE 2.1 – Modèle géométrique du capteur stéréoscopique binoculaire.

La contrainte épipolaire permet d'obtenir une relation entre les points de l'image 1 et les points de l'image 2 qui ne dépend que des paramètres des caméras et des coordonnées image et qui est indépendante des coordonnées du point de la scène. La matrice qui représente cette relation est appelée la matrice fondamentale, cf. cours de vision par ordinateur.

2.2.2 Rectification épipolaire

La rectification épipolaire consiste à se ramener à une configuration particulière du capteur, appelée configuration parallèle, dans laquelle :

les droites épipolaires sont toutes parallèles et horizontales dans les deux images et deux pixels qui se correspondent se trouvent sur la même ligne. Ainsi le déplacement entre un pixel dans l'image gauche et son correspondant dans l'image droite correspond à une simple translation.

Ainsi, la rectification épipolaire est une transformation géométrique à appliquer aux deux images. La difficulté réside dans l'estimation des paramètres de cette transformation. Deux techniques sont habituellement utilisées pour rectifier des images stéréoscopiques, selon l'information dont on dispose :

- Les paramètres de calibrage pour les deux images sont disponibles, alors les matrices de projection perspective associées à chaque image peuvent être estimées et les auteurs peuvent en déduire les transformations à appliquer pour n'avoir qu'une translation entre les deux images.
- Lorsque les informations complètes de calibrage ne sont pas connues, le calcul des paramètres de la transformation à appliquer sur les images passe par l'estimation de la matrice fondamentale à partir de correspondance de points.

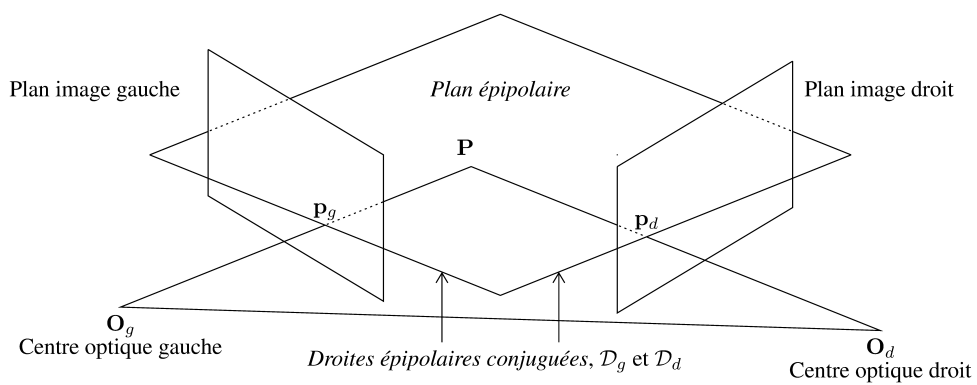


FIGURE 2.2 – Géométrie épipolaire.

Troisième partie

Détection de points d'intérêt

Chapitre 3

Introduction

3.1 Définition des primitives d'intérêt

Qu'est ce que c'est ?

Nous appelons une **primitive** un élément pouvant être extrait d'une image afin d'obtenir une information caractéristique de la scène comme la présence d'un objet, une distance, un relief (c'est-à-dire une altitude ou une profondeur). Ces primitives peuvent être des pixels, des régions, des contours, des polygones, tout ensemble de points permettant d'obtenir l'information recherchée. Dans la suite, nous étudierons particulièrement la notion de **primitive d'intérêt**, pour ensuite se consacrer à la notion de **points d'intérêt**.

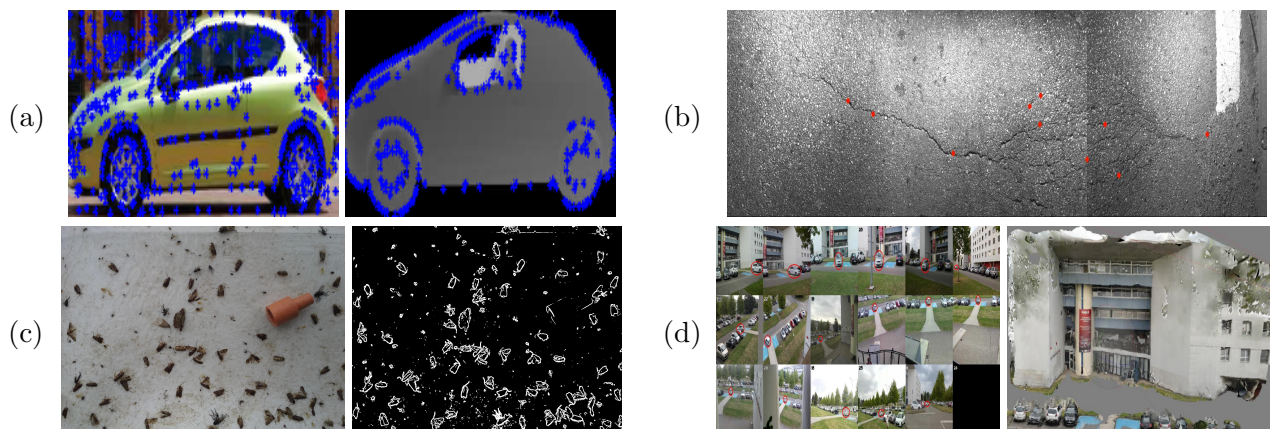


FIGURE 3.1 – Exemples de travaux de recherche sur les primitives d'intérêt en 2D – En (a), nous cherchons à obtenir des points d'intérêt répétables entre une image et une carte de profondeur. Pour cela nous exploitons les maximums de courbure dans de multiples échelles [Rashwan 16]. En (b), nous recherchons des points qui possèdent une corrélation forte dans une direction donnée afin de mettre en évidence des points d'intérêt (en rouge) appartenant à une fissure de la chaussée [Chambon 11]. En (c), nous caractérisons des points d'intérêt sur des contours (en blanc) afin de détecter des insectes [Bakkay 18]. En (d), nous avons étudié la possibilité d'effectuer une reconstruction 3D à partir de multiples caméras sur un jeu de données publié dans [Malon 18].

Une **primitive d'intérêt**, cf. exemples des figures 3.1 et 3.2, est donc un ensemble de points sur une image qui correspond à la projection de points/zones particulières d'une scène, d'un objet étudié : coins, jonctions, variation d'intensité, de texture, forme particulière, objet particulier. Il s'agit donc de **primitives ayant des caractéristiques particulières** : fort contraste, texture forte, couleurs non

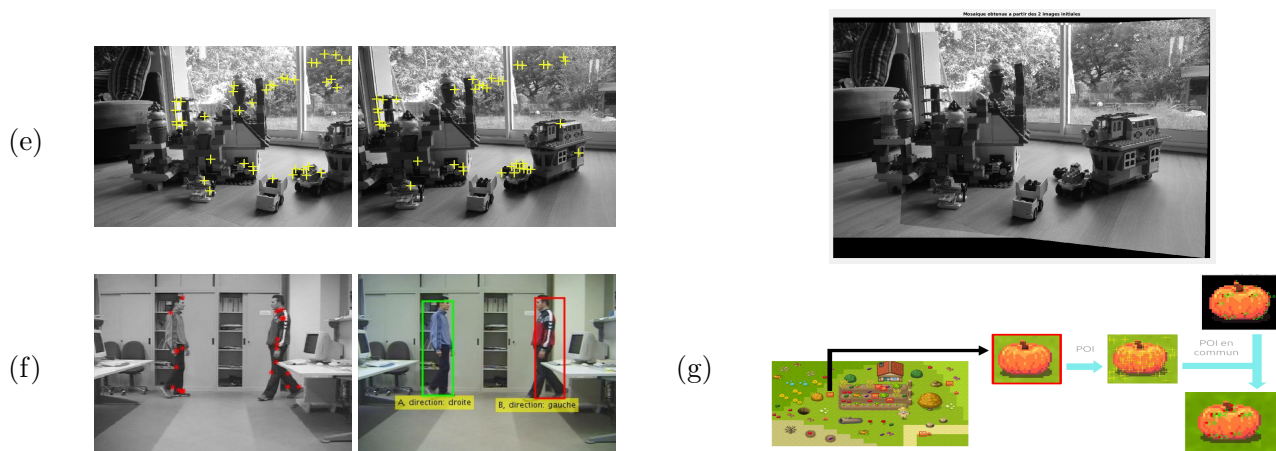


FIGURE 3.2 – Exemples de projets d'enseignement sur les primitives d'intérêt en 2D – En (e), dans le cadre de travaux pratiques proposés à l'ENSEEIH, nous détectons des points d'intérêt dans un couple d'images (croix jaunes) afin de construire un panorama de la scène (à droite). En (f), pour un bureau d'étude proposé à l'ENSEEIH, une solution consiste à détecter des points d'intérêt (en rouge) afin de suivre les personnes en mouvement. Le résultat est présenté sous la forme de boîtes englobantes indiquant la direction du déplacement (à droite). En (g), au cours d'un Apprentissage par Problème et Projet (APP), les étudiants sont amenés à détecter des points d'intérêt afin de reconnaître des éléments dans un jeu vidéo.

corrélées au reste de la scène. Ces caractéristiques particulières doivent permettre la reconnaissance, et donc le suivi, non ambigu par comparaison avec les autres entités de la scène.

Pourquoi ?

Détecter des points d'intérêt puis caractériser des points, d'intérêt ou non, est une tâche que nous retrouvons dans de très nombreuses applications en vision par ordinateur. En effet, utiliser des points caractéristiques d'un objet que nous visualisons permet de le reconnaître dans un premier temps, mais également, de caractériser, dans un second temps, ce qui le distingue des autres points de la scène. Par exemple, il peut être nécessaire de comprendre comment cet objet est différent des objets de la même catégorie ou quelles sont les qualités intéressantes qu'il possède pour extraire des informations, ou au contraire, de quelle manière il est dégradé par rapport à d'autres objets de la même classe. Ces analyses nécessitent la plupart du temps une étape de caractérisation suivie d'une étape de mise en correspondance. Nous pouvons citer de nombreuses applications existantes, illustrées dans les figures 3.1 et 3.2, comme :

- La reconnaissance de formes (reconnaître des points de fortes courbures pour détecter des objets particuliers, comme un visage [Wang 10], ou de manière plus générale, une personne, ou encore des objets [Rashwan 17]), cf. figure (a) ;
- La reconnaissance de défauts sur des objets (reconnaître des points particuliers pour identifier un défaut [Amhaz 16]), cf. figure (b) ;
- La segmentation (reconnaître des points contours pour délimiter des régions, comme dans [Bakkay 18]), cf. figure (c) ;
- La mise en correspondance [Gales 10] et plus généralement la reconstruction 3D (estimer un vecteur de déplacement grâce à la connaissance des correspondances de points, sous certaines hypothèses et calculs, et être capable de reconstruire le relief d'une scène) [Agarwal 11], cf. figure (d) ;
- La construction de mosaïque d'images ou plus simplement de panorama (mise en correspondance

de points caractéristiques qui possèdent des caractéristiques discriminantes par rapport aux autres points de la scène et sont donc plus faciles et fiables à mettre en correspondance) [Shum 98, Capel 04], cf. figure (e) ;

- Le suivi d'objets (reconnaître des points caractéristiques d'un objet particulier, comme un piéton, pour le suivre) [Gauglitz 11], cf. figure (f) ;
- L'indexation (reconnaître des régions avec des caractéristiques particulières de couleur et de forme pour indexer) [Bres 99], cf. figure (g).

Dans l'énoncé même de ces applications, nous comprenons que la détection ne peut se passer de la caractérisation, afin d'être par la suite exploitée, par exemple pour de la reconnaissance. C'est la raison pour laquelle, nous présentons dans ce chapitre l'aspect de détection alors que dans la partie IV, nous présentons la façon de caractériser ces primitives (et surtout de les apparier en s'appuyant sur ces caractéristiques).

Caractériser une primitive, c'est trouver des caractéristiques qui permettront de la reconnaître, de l'apparier, de la suivre et cette étape est aussi cruciale que la détection. Traditionnellement, détection et caractérisation sont distinguées l'une de l'autre, c'est le cas pour le détecteur de Harris [Harris 88], très largement présent dans la littérature. Mais, depuis ces deux dernières décennies, et notamment depuis la célèbre publication de [Lowe 04], la littérature abonde de méthodes permettant à la fois de détecter et de caractériser.

Comment ?

En général, un détecteur de points d'intérêt suit les étapes suivantes, cf. figure 3.3, pour chaque point \mathbf{p} de coordonnées (x, y) :

- (1) **Calcul de la réponse R d'un détecteur, cf. figure 3.3.(b)** – Plus précisément, R possède le profil suivant :

$$R : \mathbb{N}^2 \rightarrow \mathbb{R}^n.$$

Nous utilisons n car certains détecteurs donnent une réponse en dimension n , notamment les détecteurs multi-échelle.

- (2) **Suppression des non-maxima locaux, cf. figure 3.3.(c)** – Cette étape est réalisée en prenant en compte un certain voisinage \mathcal{V} à déterminer. La fonction réponse est modifiée de la manière suivante :

$$R_{\text{modifie}}(x, y) = \begin{cases} R_{\max}(x, y) & \text{si } R(x, y) = R_{\max}(x, y) \\ 0 & \text{sinon.} \end{cases} \quad (3.1)$$

La réponse $R_{\max}(x, y)$ est la réponse maximale, suivant un critère à définir, dans le voisinage $\mathcal{V}(\mathbf{p})$. Plus de détails sur les possibilités de choix sur ce critère sont donnés dans [Harzallah 11, page 18]. Cette étape est importante pour tenter d'éviter les amas de points d'intérêt mais elle est optionnelle et n'est donc pas toujours réalisée.

- (3) **Sélection des Nb points maximaux, cf. figure 3.3.(d)** – Les points \mathbf{p} pour lesquels $T(R(x, y))$ est vraie sont conservés. La plupart du temps $T(R(x, y))$ correspond au test suivant : $R(x, y) > S$ avec S , un seuil à fixer. Fixer ce seuil, quelle que soit l'approche choisie, reste toujours délicat. Plutôt que d'utiliser un seuil sur la réponse qui n'est pas toujours évident à fixer intuitivement, dans la littérature, un seuil sur le nombre de points souhaités est habituellement utilisé.
- (4) **Affinement de la détection** – Il peut s'agir d'une détection plus fine (sous-pixelique, c'est le cas pour [Lowe 04]) ou d'une suppression des points erronés (des points qui ne sont plus considérés comme des points d'intérêt). Cette étape est également optionnelle.

Les étapes (2) et (3) ne sont pas obligatoirement réalisées toutes les deux. En effet, nous pouvons imaginer les stratégies suivantes :

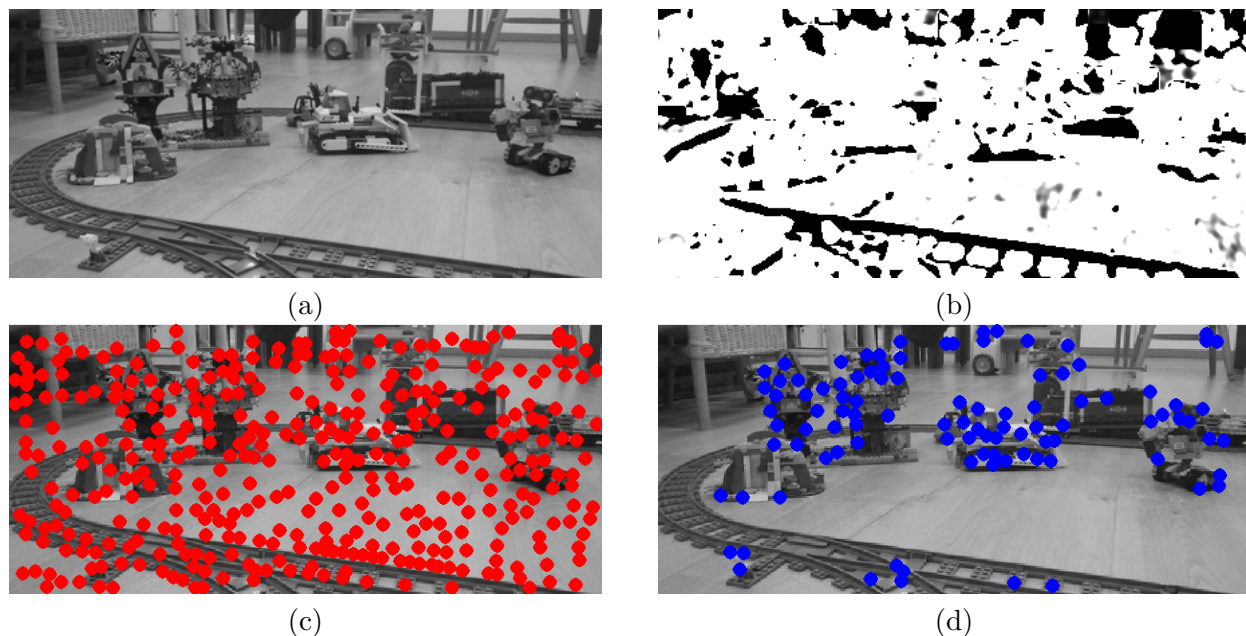


FIGURE 3.3 – Étapes classiques du fonctionnement d'un détecteur – À partir de l'image originale, en (a), nous calculons une réponse, en (b). Plus le pixel est sombre, plus la réponse est élevée. En (c), nous montrons la réponse obtenue (pastilles rouges) après suppression des non-maxima locaux. Ici, nous avons choisi un voisinage de 9×9 . Enfin, en (d), il s'agit des points sélectionnés (pastilles bleues) après un seuillage. Dans cet exemple, nous avons choisi de prendre les 100 premiers points.

- Nous appliquons uniquement l'étape (2) :
 - L'avantage est qu'il y aura des points bien répartis sur toute l'image.
 - L'inconvénient est que certains points n'auront pas des caractéristiques suffisamment saillantes pour être suivis correctement et il y a un risque élevé d'erreurs de suivi.
- Nous appliquons uniquement l'étape (3) :
 - L'avantage est qu'il y aura des points plutôt faciles à suivre car avec des caractéristiques très discriminantes.
 - L'inconvénient est que certaines zones de l'image ne posséderont aucun point suivi.

Qualité du résultat ?

Dans la littérature, de nombreuses approches ont été mises en œuvre et il est évident qu'il est nécessaire d'avoir des critères de comparaison pour évaluer la qualité des résultats obtenus. Ainsi, un premier aspect important lorsque il est nécessaire d'apparier des primitives, et en particulier des primitives d'intérêt, est la propriété de *répétabilité*. La définition que nous donnons ici, similaire à celle de [Szeliski 10], est la suivante :

La répétabilité d'une primitive correspond à sa capacité à être détectée quelle que soit l'image ou la donnée dans laquelle elle apparaît. Plus précisément, si une primitive est détectée dans une représentation donnée, alors, elle doit également être détectée dans une autre représentation.

Voici une manière précise de définir la répétabilité : si nous notons \mathbf{p} , un point dans une image I_1 , \mathbf{p}' son correspondant théorique¹ dans une autre image, I_2 alors ce point est répétable s'il existe un point

1. Cela suppose de connaître le déplacement du point \mathbf{p} et donc d'avoir une vérité terrain.

\mathbf{q} , détecté dans I_2 et tel que :

$$\|\mathbf{p}' - \mathbf{q}\| \leq \epsilon. \quad (3.2)$$

Nous notons $\mathcal{P}_{D,i}$ l'ensemble des points détectés dans l'image I_i par le détecteur D . Ainsi le taux de répétabilité entre les deux images, noté $R(I_1, I_2)$, est donné par :

$$R(\mathcal{P}_{D,1}, \mathcal{P}_{D,2}) = \frac{R(\mathcal{P}_{D,1} \rightarrow \mathcal{P}_{D,2}) + R(\mathcal{P}_{D,2} \rightarrow \mathcal{P}_{D,1})}{2}. \quad (3.3)$$

avec

$$R(\mathcal{P}_{D,i} \rightarrow \mathcal{P}_{D,j}) = \frac{\#\{\text{Points répétables de } \mathcal{P}_{D,i} \text{ vers } \mathcal{P}_{D,j}\}}{\#\{\mathcal{P}_{D,i}\}}. \quad (3.4)$$

Il est également souhaité que la détection soit exacte, précise. En effet, souvent, la localisation est imprécise, cela est dû aux différents opérateurs de lissage ou d'approximation utilisés. C'est pour cela que de nombreuses méthodes proposent une étape de **relocalisation**.

Il peut être nécessaire d'être robuste à certaines difficultés (cette nécessité est souvent liée à la notion de suivi). La plupart du temps ces propriétés de robustesse sont à rapprocher de l'étape d'appariement abordée au chapitre IV. Dans [Tuytelaars 06], les auteurs résument ainsi parfaitement les propriétés attendues pour un détecteur « idéal »:

- Sa définition doit être locale afin de lui permettre d'être robuste aux occultations.
- Il doit être invariant, ou co-variant, aux transformations d'images, ce qui signifie qu'une transformation appliquée à l'image n'affectera pas le résultat du détecteur, ou l'affectera de manière consistante avec la transformation appliquée.
- Il doit être robuste à tous types de dégradations de l'image : bruits, flou, discrétisation ou encore compression.
- Une grande quantité de primitives d'intérêt doit être trouvée quelque soit l'objet, même s'il est de petite taille.
- La détection doit être rapide, c'est-à-dire proche du traitement en temps réel.

Bien évidemment, les auteurs ajoutent à ces caractéristiques celles que nous avons déjà énoncées : précision et répétabilité. Cette dernière notion est plutôt définie comme la capacité à être discriminant, c'est-à-dire la capacité du détecteur à proposer des primitives qui peuvent être appariées avec le moins d'ambiguïté possible à travers tout type de représentation de cette primitive dans une large base de données. Il est intéressant de souligner la contradiction entre le fait d'être répétable et discriminant, tout en étant robuste à des transformations, cela nous permet de mettre en évidence la difficulté de cette tâche de détection.

Pour terminer sur cet aspect, des publications ont tenté d'évaluer et comparer plus finement les performances des différents détecteurs, comme les articles de référence suivants : [Mikolajczyk 04, Aanaes 12].

Et le descripteur ?

Entre 1980 et 2000, ce sont essentiellement des détecteurs qui ont été proposés et très peu de descripteurs. Le plus connu est le descripteur associé au détecteur SIFT, *Scale Invariant Feature Transform*, mais nous effectuons une présentation des descripteurs principaux dans ce cours, cf. 4.6. De plus, la plupart du temps, la manière de calculer la différence entre deux descripteurs est une simple distance euclidienne ou la corrélation croisée centrée normalisée ou encore une distance de Mahalanobis. Il est possible d'utiliser bien d'autres distances, cf. partie IV, mais cet aspect n'a pas été beaucoup investi dans la littérature.

3.2 Approches de détection et de caractérisation des points d'intérêt en 2D

La littérature est très abondante sur le sujet de la détection de primitives d'intérêt en 2D et il est délicat d'établir une classification des approches. En effet, parmi toutes les approches existantes, il est possible d'effectuer un tri en distinguant les approches locales (prise en compte d'un voisinage), des approches globales (comme les approches utilisant les outils de transformation de Fourier). Nous pouvons également distinguer les primitives photométriques, qui s'appuient sur l'étude de l'intensité ou de la texture, des primitives géométriques, qui prennent en compte des notions plus proches de la géométrie, comme les notions de segment ou d'intersection. Enfin, un critère important pour caractériser les différentes approches existantes est de considérer l'aspect multi-résolution ou multi-échelle, c'est-à-dire, est-ce que l'approche proposée fait intervenir différentes résolutions des images ou des objets manipulés, ou est-ce que l'approche fait intervenir différentes échelles dans les outils utilisés, comme différentes tailles de filtrage. Enfin, les acquisitions actuelles, aussi bien en 2D qu'en 3D permettent la collecte de nombreux paramètres liés aux conditions d'acquisition, ainsi que de nombreuses informations extraites en pré-traitement, c'est ce que nous appelons de manière générale des métadonnées. Par exemple, il est possible d'obtenir la date, l'heure, les dimensions, la focale, la vitesse d'obturation, les coordonnées GPS ou encore le modèle de l'appareil. Toutes ces informations sont surtout utiles pour positionner et calibrer. Nous pouvons également citer toutes les applications ajoutées qui permettent, par exemple, la détection automatique des visages, des yeux, des zones de flou, une première indexation de l'image (comme intérieur, extérieur, jour, nuit) [Hu 15]. Parmi toutes ces possibilités, nous avons choisi d'extraire les aspects que nous avons jugés importants, c'est-à-dire les plus exprimés et les plus utilisés dans la littérature. C'est ce que nous détaillons dans les paragraphes suivants et que nous résumons dans la figure 3.4.

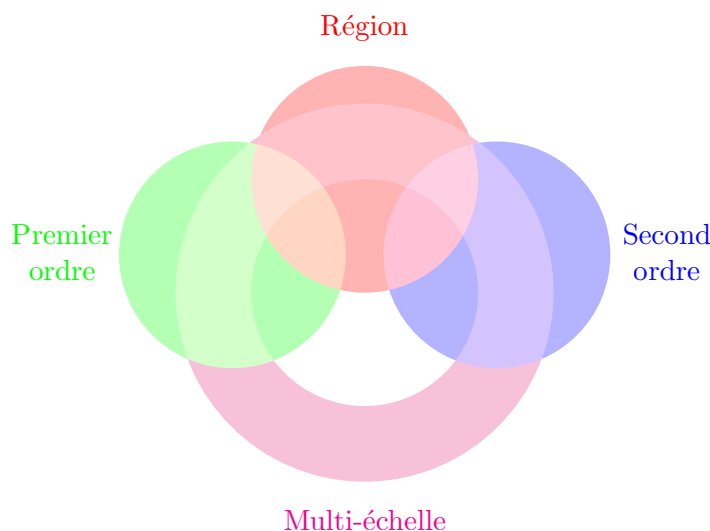


FIGURE 3.4 – Classement des approches de détection de points d'intérêt en 2D – Ce graphique nous permet d'illustrer les interactions entre les différents éléments que nous avons identifiés. Ainsi, nous choisissons de distinguer trois familles (les trois cercles) mais ce qui relie ces familles c'est la généralisation de certains opérateurs à la multirésolution (l'anneau). Les intersections mettent en évidence qu'il existe bien des approches qui peuvent être classées dans plusieurs familles et que, dans la table 3.1, nous avons fait un choix. À notre connaissance, aucune approche ne mélange approche du premier et du second ordre.

3.2.1 Approches basées premier ordre

Les approches s'appuyant sur une détection de contours [Canny 86], c'est-à-dire exploitant les dérivées premières de l'image, sont les approches fondatrices de cette discipline. Nous les appellerons les approches de premier ordre. L'approche la plus célèbre consiste à exploiter le détecteur de [Harris 88], cf. § 4.3.1, qui permet de détecter plus particulièrement des coins. Le principe est d'estimer à quel point un point étudié est corrélé à ses voisins, en s'appuyant sur la publication initiale de [Moravec 80], cf. page 41.

3.2.2 Approches basées région

Ce genre de technique propose de définir une région caractéristique autour du point étudié et, ensuite, d'étudier le comportement de cette région, afin de déterminer le caractère unique du point étudié. Un point d'intérêt ainsi obtenu est très différent car il correspond au centre d'une région d'intérêt et non à un point caractéristique sur le contour d'une forme donnée. Dans [Tuytelaars 06], les auteurs insistent sur le fait que ce type de détecteur correspond à une nouvelle manière de décrire un point d'intérêt : il ne s'agit plus de juste sélectionner des lieux caractéristiques dans une image pour en extraire l'information pertinente pour faire des calculs ou tout simplement pour réduire les temps de calcul mais il s'agit de trouver une nouvelle façon de représenter l'image permettant d'en décrire les objets, sans pour autant s'appuyer sur une segmentation. Ainsi, de nombreuses versions de détecteurs locaux s'appuyant sur l'étude de région d'intérêt autour d'un point, parfois appelées *blob*, ont été introduites. L'approche la plus connue est celle de SUSAN, *Smallest Univalued Segment Assimilating Nucleus* [Smith 97], cf. § 4.4.1, qui étudie la proportion de pixels photométriquement proches du pixel étudié dans une région circulaire autour de lui. Une approche similaire mais plus simple à implémenter et permettant une accélération des calculs est FAST [Rosten 06], cf. § 4.4.2.

Dans [Matas 02], les auteurs considèrent simplement les composantes connexes d'une image seuillée. Ils appellent ce détecteur MSER, *Maximally Stable Extremal Regions*, cf. § 4.4.3 pour plus de détails. Pour le détecteur IBR, *Intensity Based Regions* [Tuytelaars 04], les auteurs introduisent la notion de blob. Les régions d'intérêt détectées autour des points prennent en compte les maxima le long de droites partant du point étudié. L'approche se termine en calculant une approximation par une ellipse de la région formée par tous ces maxima locaux, cf. § 4.4.4. Enfin, une dernière approche est celle de [Deng 07], PCBR, *Principal Curvature-Based Regions*. Elle est introduite comme une approche permettant d'étudier des régions où des contours sont détectés par ligne de partage des eaux. Cette approche s'appuie également sur le calcul de la courbure principale, c'est pourquoi, pour nous, elle appartient également à la dernière catégorie d'approches que nous distinguons.

3.2.3 Approches basées second ordre

Les approches par détection de la courbure sont les approches de second ordre puisqu'elles exploitent les dérivées secondes. Un des détecteurs les plus anciens est celui de Beaudet, appelé aussi le détecteur Hessien [Beaudet 78], décrit dans le § 4.5.1. Ce détecteur estime la courbure de la surface alors que le détecteur de Kitchen et Rosenfeld [Kitchen 82], cf. § 4.5.3, utilise la notion de courbure du contour passant par le point considéré. De plus, chaque point utilisé est pondéré par la norme du gradient car les auteurs souhaitent prendre en compte les cas où la norme du gradient est faible (donc un contour peu contrasté, qui peut correspondre à un bruit). Ce détecteur s'appuie sur les dérivées secondes et donc sur la notion de courbure de la surface. Plus récemment, les auteurs de [Fischer 14] ont proposé un détecteur s'appuyant sur la courbure, notée κ . Cette courbure est exprimée comme le changement, noté q , de gradient image le long de la tangente. Ce terme q est donc une approximation de κ . Dans la publication de [Deng 07], les auteurs s'appuient sur la valeur propre minimale (qui met en valeur une frontière sombre sur un fond clair) ou maximale (qui met en valeur une frontière claire sur un

fond sombre) de la matrice Hessienne afin d'estimer la courbure principale dans un contexte de multi-résolution, cf. § 4.4.7.

3.2.4 Introduction des notions d'analyse multi-échelle et multi-résolution



FIGURE 3.5 – Illustration de l'introduction d'une analyse multi-échelle dans une pyramide d'images (multi-résolution) – Ces illustrations sont extraites de la thèse de Guillaume GALÈS [Gales 11]. De haut en bas, la résolution des images est diminuée, formant ainsi une pyramide d'images. De gauche à droite un filtrage gaussien est appliqué avec un filtre de plus en plus grand, pour fournir une analyse multi-échelle.

Plus récemment, de nombreuses approches ont été introduites afin d'avoir un détecteur multi-échelle ou multi-résolution. Ici, nous faisons le choix d'appeler l'**échelle le niveau de détails utilisé pour observer/analyser** une image, une scène, un ensemble de données, alors que la **résolution est liée à la taille de l'image utilisée**, cf. figure 3.5. Suivant l'échelle d'analyse choisie, les difficultés suivantes peuvent apparaître : avec une échelle trop faible, des détails non significatifs (un motif de surface) peuvent être pris en compte et produire des détections non pertinentes alors qu'avec une échelle trop importante, des détails pertinents de la scène mais de petite taille peuvent être ignorés. De plus, dans le cas du suivi de primitives, nous pouvons observer des changements de résolution entre images/objets comparés. Pour toutes ces raisons, le célèbre détecteur SIFT, Scale Invariant Features Transform [Lowe 04], a été introduit. Par la suite, le raisonnement multi-échelle a été généralisé à d'autres détecteurs connus, comme celui de Harris [Mikolajczyk 04]. Dans de nombreux cas, cette capacité à s'adapter aux différentes échelles va se traduire par le fait de faire varier un (ou plusieurs) paramètre(s) utilisés par le détecteur. Par exemple, dans le cas du détecteur SIFT, c'est la taille du filtre gaussien utilisé pour le lissage et pour le calcul du laplacien qui va varier. Cette variation permet de mettre en évidence les détails plus ou moins fins des données étudiées. L'inconvénient principal de ces approches étant le temps de calcul, des approches mettant en œuvre des astuces algorithmiques ont été introduites, comme celle de [Bay 08] avec le détecteur SURF, *Speeded Up Robust Features*. Encore une fois, ces approches s'appuient essentiellement sur la détection des variations de texture et d'intensité.

Les détecteurs mono-échelle utilisent une réponse associée à un lissage dépendant d'un paramètre :

l'échelle. Ils détectent les points ou régions à une échelle unique. Cette échelle étant *a priori* inconnue, les approches multi-échelles cherchent à s'affranchir de cette difficulté en effectuant une analyse sur de multiples échelles et à fusionner les résultats obtenus pour fournir une réponse multi-échelle. Dans la littérature, il a été démontré que le noyau gaussien est le seul lissage linéaire permettant une analyse multi-échelle correcte [Yu 11]. Ainsi, les détecteurs actuels construisent une pyramide gaussienne, calculent une mesure d'intérêt spatiale normalisée pour chaque pixel et pour chaque niveau de la pyramide, puis détectent les extrema localisés en x , y , σ dans cet espace-échelle 3D.

Ainsi, l'opérateur de premier ordre que nous avons déjà introduit a été généralisé au multi-échelle par [Mikolajczyk 04] pour donner l'approche Harris-Laplace, cf. § 4.3.2.

De même, en ce qui concerne les détecteurs basés régions, [Tuytelaars 04] a proposé une version multi-échelle avec l'opérateur EBR, *Edge-Based Regions*, qui étudie les familles de parallélogrammes définis à partir du point étudié et des points de contours passant par le point étudié, cf. § 4.4.6.

Encore une fois, les auteurs de [Mikolajczyk 04] ont généralisé l'opérateur Hessien, du second ordre, au multi-échelle, Hessien-Laplace, cf. § 4.5.2.

La méthode SIFT, *Scale Invariant Feature Transform* [Lowe 04], cf. § 4.5.5, s'appuie sur la détection des extrema en échelle et en espace du laplacien. Les auteurs utilisent des différences de gaussiennes pour approximer le laplacien et ils effectuent les calculs avec différentes tailles (échelles) de filtre dans différentes résolutions (utilisation d'une pyramide d'images). La particularité de ce détecteur est que les auteurs ont proposé l'utilisation d'un descripteur associé qui considère l'orientation principale qui correspond au gradient dominant dans le voisinage du point considéré. Le descripteur associé à ce détecteur prend ainsi en compte un histogramme local de ces orientations. Il existe une variante permettant de diminuer les temps de calcul : SURF, *Speeded Up Robust Features* [Bay 08], cf. § 4.5.6. Enfin, le dernier détecteur de second ordre multi-échelle est CSS, *Curvature Scale Space* [Mokhtarian 98], cf. § 4.5.4, qui recherche les extrema de la courbure le long des contours en utilisant une approche multi-échelle.

3.2.5 Classement des publications significatives et invariance des détecteurs

La table 3.1 nous permet de reprendre les différents détecteurs considérés en fonction de leur catégorie. De plus, la table 3.2 permet de résumer toutes les invariances que nous avons répertoriées dans la littérature sur tous les détecteurs étudiés. Toutes ces approches sont détaillées dans le chapitre 4. contributions.

	Premier ordre	Région	Second ordre
MONO-ÉCHELLE	HA [Moravec 80, Harris 88]	SUSAN [Smith 97] FAST [Rosten 06] MSER [Matas 02] IBR [Tuytelaars 04]	BE [Beaudet 78] KR [Kitchen 82]
MULTI-ÉCHELLE	Harris-laplace [Mikolajczyk 04]	KA [Kadir 04] EBR [Tuytelaars 04]	Hessien-Laplace [Mikolajczyk 04] SIFT [Lowe 04] SURF [Bay 08] CSS [Mokhtarian 98] MFC [Rashwan 17]
			PCBR [Deng 07]

TABLE 3.1 – Publications significatives sur la détection de points d'intérêt en 2D.

Dans la figure 3.6, nous présentons le résultat de détection de points d'intérêt avec quatre détecteurs différents. Nous pouvons observer que les points obtenus sont assez complémentaires les uns des autres.

Nous comprenons ainsi la difficulté pour choisir un détecteur et également l'intérêt qu'il peut y avoir à combiner différentes catégories de détecteurs.

Ainsi, nous avons présenté notre manière de classer et de comparer les détecteurs en fonction de leur définition et de leurs invariances. À présent, nous allons présenter les détails sur les détecteurs présentés, dans le chapitre 4 mais vous pouvez trouver plus de détails encore dans la thèse de Guillaume GALÈS, en particulier une page dédiée à la comparaison des paramètres utilisés [Gales 11, pages 100-101].

Détecteur	Affines photo- métriques	Géométriques non affine	Affines géométriques	Changement d'échelle
[Moravec 80, Harris 88]				
Harris-laplace [Mikolajczyk 04]				×
SUSAN [Smith 97] FAST [Rosten 06] MSER [Matas 02] IBR [Tuytelaars 04]	×	×	×	
KA [Kadir 04] EBR [Tuytelaars 04] PCBR [Deng 07]	×	×	×	×
[Beaudet 78] [Kitchen 82]				
Hessien-Laplace [Mikolajczyk 04] SIFT [Lowe 04] SURF [Bay 08] CSS [Mokhtarian 98] <i>MFC</i> [Rashwan 17]	×	×	×	×

TABLE 3.2 – Invariance des différents détecteurs présentés – Dans ce tableau, nous répertorions les invariances annoncées par les auteurs des différentes publications. Le détecteur MCF [Rashwan 17] n'est pas décrit dans ce cours, il propose une méthodologie complète de mise en correspondance 2D/3D.

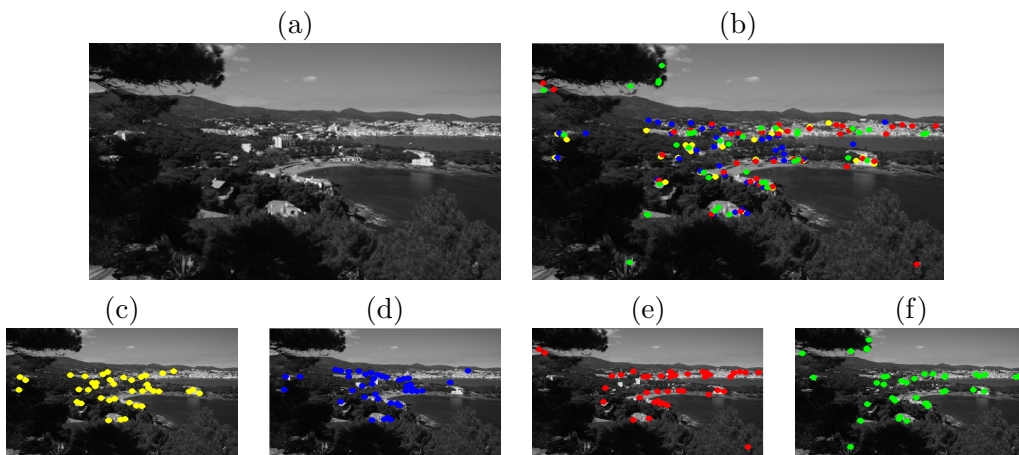


FIGURE 3.6 – Résultats obtenus avec quatre détecteurs de points d'intérêt en 2D – En (a), nous présentons l'image originale étudiée et en (b), l'ensemble des points d'intérêt détectés. Nous pouvons observer plus distinctement les différences avec les résultats présentés séparément, en (c) avec Harris (pastilles jaunes), en (d), avec le détecteur de Beaudet (pastilles bleues), en (e) avec le détecteur de Kitchen et Rosenfeld (pastilles rouges) et en (f), le détecteur SURF (pastilles vertes). Il s'agit du même code couleur utilisé en (b). Il faut noter que les résultats présentés sont ceux obtenus pendant les travaux pratiques et le projet réalisés aux cours des enseignements de spécialité à l'ENSEEIH (les étudiants codent les 3 premiers détecteurs et réutilisent celui codé par `Matlab`, pour le dernier).

Chapitre 4

Détecteurs de points d'intérêt

4.1 Notations

- I : une image qui à chaque point/pixel \mathbf{p} associe un niveau de gris noté $I(x, y)$;
- $I_x, I_y, I_{xx}, I_{yy}, I_{xy}$: images des dérivées premières et secondes de l'image ;
- $I_x(x, y), \dots, I_{xy}(x, y)$: valeurs des dérivées premières et secondes au point \mathbf{p} ;
- Pour simplifier, on notera I_x pour $I_x(x, y)$...

Nous avons choisi de présenter les approches en les séparant en trois catégories : opérateur du premier ordre, opérateur basé région et opérateur du second ordre. De plus, pour chaque catégorie, nous indiquons les approches multi-échelle. C'est pourquoi, dans un premier temps, nous présentons la manière la plus classique de construire une pyramide d'images.

4.2 Construction d'une pyramide d'images

Le problème consiste à produire à partir d'une image, un ensemble d'images de résolution de plus en plus faibles. Plusieurs techniques ont été développées pour calculer des pyramides d'images, principalement, dans le domaine de la compression d'images [Burt 83]. Il existe différents types de pyramides :

- Les **pyramides régulières** – Ce sont les plus couramment employées, Le même voisinage est considéré dans la phase de réduction de la résolution des images.
- Les **pyramides irrégulières** – Elles sont utilisées de manière beaucoup plus marginale [Montanvert 91]. Nous pouvons distinguer les pyramides stochastiques exposées dans [Jolion 03], des pyramides centrées proposées dans [Brigger 99].

Les pyramides irrégulières ne sont pas utilisées dans le cadre du suivi. Elles ont souvent été proposées pour effectuer une segmentation en régions, cf. cours 2A. Ces pyramides semblent, en effet, mieux respecter les différentes structures de l'image et ne pas supprimer les détails pertinents. Dans la suite, nous ne détaillerons donc que les pyramides régulières.

Si nous notons n la résolution courante à calculer (qui est plus faible que la résolution $n - 1$), la construction d'une pyramide régulière d'images en chaque point (i, j) est définie par :

$$I^n(i, j) = \text{reduction}_{(i', j') \in ZV(i, j)} f_r(I^{n-1}(i', j')), \quad (4.1)$$

sachant que $ZV(i, j)$ correspond au voisinage de réduction (c'est une fonction qui permet de retourner l'ensemble des pixels considérés au niveau $n - 1$ pour évaluer la valeur d'un pixel au niveau n), f_r est la fonction qu'il faut appliquer aux pixels pris en compte (le plus souvent il s'agit d'une pondération) et l'opérateur reduction est une fusion des valeurs des pixels voisins du niveau $n - 1$ pour obtenir le niveau n .

Les pyramides régulières les plus courantes et les plus utilisées dans le cadre de la mise en correspondance sont :

- Les **pyramides gaussiennes** – Il s’agit d’effectuer un lissage par un filtre gaussien, puis un sous-échantillonnage, en récupérant un pixel sur quatre.
- Les **pyramides moyennes** – Une moyenne sur un voisinage 2×2 est calculée.

4.3 Détecteurs du premier ordre

Il y a peu de détecteurs dans cette famille et leur principe est simple : il s’agit d’estimer à quel point **un point étudié est corrélé à ses voisins**. Cette famille est liée au détecteur le plus ancien et le plus célèbre : le détecteur de [Harris 88]. Par la suite, une variante multi-échelle a été introduite et nous la présentons également.

4.3.1 Auto-corrélation [Harris 88]

Principe d’un détecteur basé auto-corrélation [Moravec 80]

Le principe est d’estimer à quel point **un point étudié est corrélé à ses voisins**. Plus précisément, si on pose (u, v) un déplacement quelconque alors le détecteur de Moravec revient à évaluer l’impact de ce déplacement en (x, y) de la façon suivante :

$$E_{(u,v)}(x, y) = \sum_{x,y} w(x, y) (I(x + u, y + v) - I(x, y))^2 \quad (4.2)$$

où :

- w correspond à un poids. Plus précisément, si le point est pris en compte dans le calcul, le poids associé vaut 1, sinon, c’est 0. Cela signifie que le voisinage correspond à toute l’image.
- Ainsi, nous pouvons constater que $E_{(u,v)}(x, y)$ évalue la **moyenne du changement** lorsque la fenêtre est déplacée.

Ainsi, en utilisant cette réponse E , on peut considérer les trois cas suivants, cf. figure 4.1 :

- (1) Dans une zone homogène, la réponse donnée par E sera faible dans toutes les directions.
- (2) Dans une zone sur un contour rectiligne, les termes de E , le long du contour seront faibles, alors que perpendiculairement au contour ce sera élevé.
- (3) Dans une zone de coins, voire d’un point isolé, la plupart des termes de E seront élevés et donneront donc une réponse globale élevée.

En conséquence, le principe du détecteur de Moravec est de donner comme réponse la **valeur minimale obtenue par Mor parmi toutes les directions testées**:

$$Mor(x, y) = \min_{(u,v)} E_{(u,v)}(x, y)$$

Améliorations proposées par Harris et Stephen

Le détecteur de Moravec présente plusieurs défauts. En particulier, **Harris** et **Stephen** ont identifié des erreurs/des problèmes et ils ont également proposé de les corriger :

- (1) La **réponse est bruitée en raison des poids binaires**.

Harris Utilisation des poids Gaussiens, de moyennes nulles, donc :

$$w(x, y) = e^{(-\frac{x^2+y^2}{2\sigma^2})} \quad (4.3)$$

- (2) La **réponse du détecteur est anisotrope (dépendant de la direction)** du fait que l’on utilise un pas discret de 45 degrés.

ET

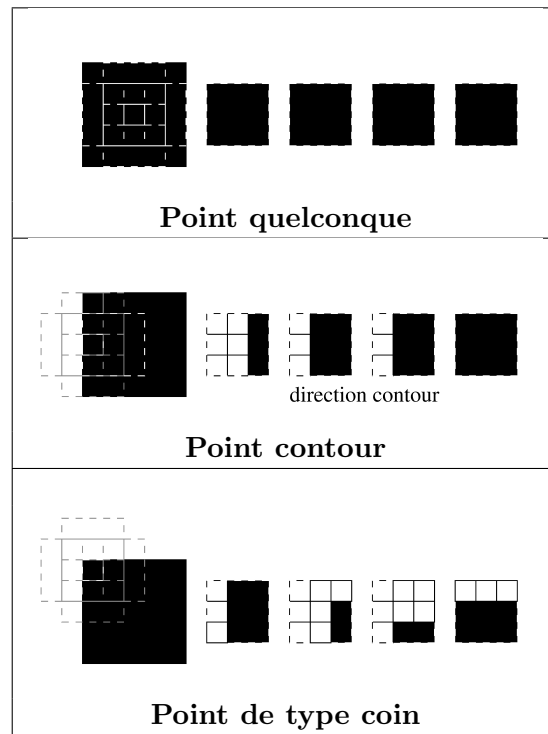


FIGURE 4.1 – Comportement de l'auto-corrélation suivant le type de points – Pour chaque cas, on présente à gauche un point et son voisinage (en binaire), puis à droite nous développons les voisinages de ces voisins 4-connexes.

- (3) La réponse est trop forte/ambiguë au niveau des contours puisque finalement, une seule valeur est conservée.

Harris Pour répondre à ces deux problèmes/difficultés, les auteurs ont proposé d'utiliser un **développement de Taylor à l'ordre 2** au voisinage de (x, y) :

On peut écrire :

$$I(x + u, y + v) = I(x, y) + u \frac{\partial I}{\partial u} + v \frac{\partial I}{\partial v} + o(u^2, v^2). \quad (4.4)$$

D'où, si on remplace dans l'équation (4.2) :

$$E_{(u,v)}^{\text{modifiée}}(x, y) = \sum_{x,y} w(x, y) \left(u \frac{\partial I}{\partial u} + v \frac{\partial I}{\partial v} + o(u^2, v^2) \right)^2. \quad (4.5)$$

En négligeant $o(u^2, v^2)$, ce qui est valide dans le cadre de **petits déplacements**, on obtient :

$$E_{\text{modifiée}} = Au^2 + 2Cuv + Bv^2, \quad (4.6)$$

avec $A = \frac{\partial^2 I}{\partial u^2} \otimes w$, $B = \frac{\partial^2 I}{\partial v^2} \otimes w$ et $C = \frac{\partial I}{\partial u} \frac{\partial I}{\partial v} \otimes w$ où \otimes représente la convolution.

Harris Ainsi, grâce à cette nouvelle façon d'écrire, on peut proposer de prendre en compte tout le comportement local à partir de E modifiée. En repartant de l'équation (4.5) ré-écrite de la manière suivante :

$$E_{(u,v)}^{\text{modifiée}}(x, y) = (u, v) \dot{M}(u, v)^T, \quad (4.7)$$

avec $M = \begin{pmatrix} A & C \\ C & B \end{pmatrix}$. Les **valeurs propres de la matrice M correspondent aux courbures principales associées à E** , ainsi :

- Si les deux courbures sont faibles, la région a une intensité constante.
- Si une des courbures est forte alors que l'autre est faible, alors la région se trouve sur un contour.
- Si les deux courbures sont fortes alors l'intensité varie fortement dans toutes les directions et on a caractérisé un coin.

Définition de la réponse de Harris et Stephen

Ainsi, la réponse du détecteur est définie par :

$$R(x, y) = \text{Det}(M) - k\text{Trace}(M)^2, \quad (4.8)$$

avec $\text{Det}(M) = AB - C^2$ et $\text{Trace}(M) = A + B$. Avec cette formulation, **on s'est affranchi du choix de la direction**. On a pour habitude de prendre comme valeur pour k une valeur entre 0.04 et 0.06.

Ainsi, l'analyse de R permet de conclure que les valeurs de R sont :

- positives au voisinage d'un coin,
- négatives au voisinage d'un contour et
- faibles dans une région d'intensité constante.

4.3.2 Vers le multi-échelle : Harris-Laplace [Mikolajczyk 04]

La version multi-échelle de Harris proposé dans [Mikolajczyk 04] s'appuie sur les deux étapes suivantes :

- (1) **Calcul de la réponse de Harris à plusieurs échelles** – Les échelles sont définies par :

$$\sigma_k = c^k \sigma_0,$$

avec c le facteur d'échelle. La manière de fixer k dépend de l'échelle d'intégration (lissage d'un niveau d'échelle à un autre) et de l'échelle de différentiation (lissage pour calculer les dérivées). **Les extrema locaux de la réponse sont estimés en (i, j, σ) . Ils vont être utilisés à l'étape suivante.**

- (2) **Affinement de la sélection des points d'intérêt de coordonnées (i, j) dans l'espace échelle du laplacien** – Pour cette étape, un algorithme itératif est appliqué **pour chaque candidat retenu à la première étape**. Soient (i_0, j_0, σ_0) , les coordonnées et l'échelle initiale du candidat considéré :
 - (a) Recherche d'un extremum pour le candidat dans l'espace échelle du laplacien pour les échelles comprises dans $[c^{-1}\sigma_k; c\sigma_k]$, avec intervalle inter-échelle déterminé par $c_2 < c$. Si un extremum n'est pas trouvé alors ce candidat est rejeté sinon σ_u , l'échelle à laquelle cet extremum a été trouvé, est conservée.
 - (b) Détection de la position (i_{k+1}, j_{k+1}) du maximum de la réponse de Harris le plus proche de (i_k, j_k) à l'échelle σ_u .
 - (c) Nouvelle itération si $\sigma_k \neq \sigma_{k+1}$ ou $(i_k, j_k) \neq (i_{k+1}, j_{k+1})$.

La sortie de cet algorithme sera donc les points d'intérêt détectés en prenant en compte l'échelle. Une version moins coûteuse est possible en étant plus sélectif à la fin de la première étape en appliquant un seuillage.

4.4 Détecteurs de régions

Pour ce type de détecteur, le principe commun est de considérer non plus un point d'intérêt comme un point qui se distingue des autres, mais plutôt, comme le centre d'une région qui se distingue des

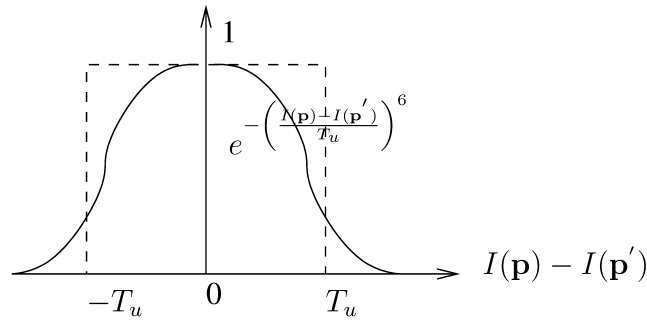


FIGURE 4.3 – SUSAN : Comportement de la fonction de similarité.

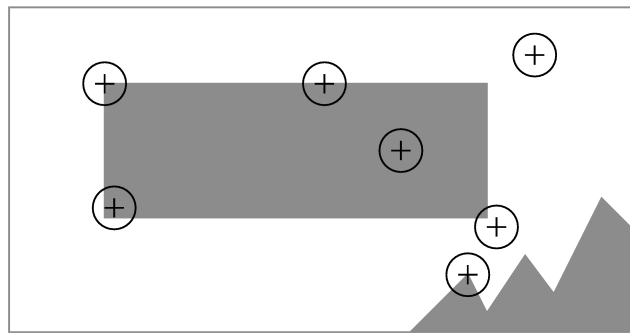


FIGURE 4.4 – Comportement du détecteur de SUSAN – Les cercles correspondent au voisinage pris en compte.

- Si la surface de l'USAN est proche de la surface du voisinage alors le pixel est situé dans une zone homogène.
- Si la surface de l'USAN est proche de la moitié de la surface du voisinage alors le pixel est situé sur un contour.
- Si la surface de l'USAN est plus petite que la moitié de la surface du voisinage alors le pixel est un coin et c'est un point d'intérêt candidat.

On voit clairement qu'en jouant sur ce seuil on peut **moduler le type de points (contours/coins) détectés**.

Habituellement, SUSAN est utilisé pour détecter des points d'intérêt et on prend généralement $T_g = \frac{s_{max}}{2}$ où s_{max} est la surface maximale de l'USAN étant donnée la taille du voisinage considéré.

Post-traitement ou affinement spécifique Le détecteur peut également faire intervenir des post-traitements afin d'améliorer la qualité des résultats. Certains candidats peuvent être rejetés à l'aide de ces **deux contraintes** :

- (1) **Contrainte du centre** – Le centre de l'USAN doit être éloigné d'une distance d (en pixels) du noyau, cf. figure 4.5. Les coordonnées du centre de l'USAN sont données par le barycentre de l'USAN :

$$x_c/y_c = \frac{\sum_{\mathbf{p}' \in \text{USAN}(\mathbf{p})} (x/y' - x/y) f(\mathbf{p}')}{\sum_{\mathbf{p}' \in \text{USAN}(\mathbf{p})} f(\mathbf{p}')}$$

Et il faut donc vérifier que $\sqrt{(x_c - x)^2 + (y_c - y)^2} > d$.

- (2) **Contrainte de continuité** – Tous les points sur le segment reliant le noyau et le centre doivent appartenir à l'USAN, cf. figure 4.6.

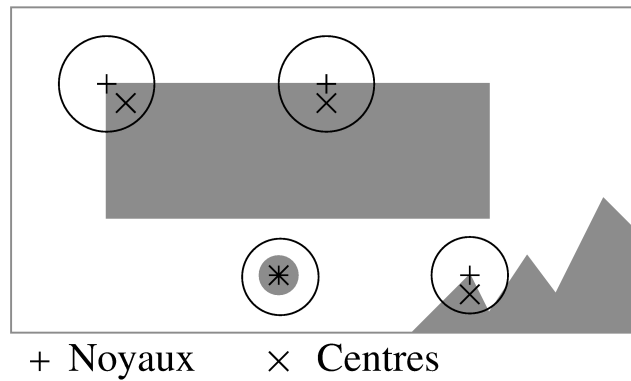


FIGURE 4.5 – Contrainte du centre.

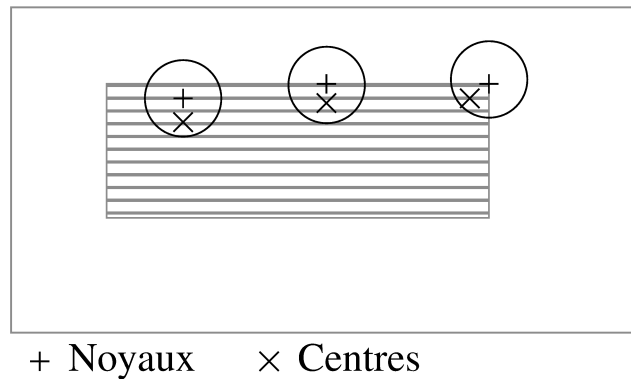


FIGURE 4.6 – Contrainte de continuité.

Plus on ajoute de contraintes, plus on élimine de “mauvais” points d’intérêt mais plus on risque d’éliminer de “bons” points d’intérêt.

4.4.2 FAST, *Features from Accelerated Segment Test* [Rosten 06]

Ce détecteur possède des éléments en commun avec celui de SUSAN présenté au paragraphe précédent. En effet, initialement introduit par [Trajković 98], le principe de ce détecteur est d’étudier une zone circulaire et d’évaluer les ensembles de pixels de même intensité qui forment une région connexe. Plus précisément, les auteurs utilisent une zone circulaire respectant l’algorithme du cercle de Bresenham de rayon 3. Parmi les 16 points sur le périmètre de ce cercle, ils vérifient si une de ces hypothèses est vraie:

1. Il existe $N_p = 12$ pixels connexes tels que $I(x', y') < I(x, y) - T$.
2. Il existe $N_p = 12$ pixels connexes tels que $I(x', y') > I(x, y) + T$.

Ce test implique le choix d’un paramètre, le seuil T . Pour simplifier l’approche et la rendre encore plus rapide, il est proposé de simplifier encore plus ce test en ne considérant que 4 des 16 pixels qui composent le contour, les pixels au position 1, 5, 9 et 13. Si 3 des 4 pixels vérifient une de ces deux hypothèses, alors, le pixel à la position (x, y) est considéré comme un point d’intérêt.

Cette approche assez simple possède les inconvénients suivants : il est difficile d’adapter cet algorithme lorsque $N_p < 12$, le choix des N_p pixels connexes implique des hypothèses fortes sur l’apparence des points d’intérêt, de nombreuses primitives sont détectées de manière adjacente. Pour toutes ces raisons, les auteurs ont proposé d’ajouter une phase d’apprentissage.

FIGURE 4.7 – Illustration du cercle utilisé pour le détecteur FAST.

4.4.3 MSER, *Maximally Stable Extremal Regions* [Matas 02]

Ce détecteur repose sur l'idée que si nous seuillons progressivement une image, nous allons voir apparaître des points puis des régions qui vont ensuite se rejoindre pour fusionner ou déterminer les frontières entre régions d'intérêt. Ce principe est comparable à celui de la ligne de partage des eaux qui est d'ailleurs l'algorithme utilisé. L'ensemble de toutes les composantes connexes obtenues, à chaque itération, forme l'ensemble des régions maximales. Parmi ces régions, certaines sont considérées comme des maxima/minima car tous les points à la frontière de la région, mais n'appartenant pas à la région, ont une intensité plus grande/petite que les points appartenant à la région. De plus, parmi toutes les régions maximales, certaines vont rester stables sur plusieurs itérations. Elles constituent alors ce que les auteurs appellent les régions maximales stables. Les régions d'intérêt, dans ce contexte, sont donc, les régions maximales extrema et stables.

Ces régions sont invariantes aux transformations monotones d'intensité (et non uniquement aux transformations affines photométriques) et aux transformations géométriques homographiques ou non linéaires mais continues. L'inconvénient de ce détecteur est sa sensibilité au flou, car dans ce cas l'intensité sur les frontières est mal définie et les contours des régions sont également mal localisés.

4.4.4 IBR, *Intensity Based Regions* [Tuytelaars 04]

Voici les étapes suivies par ce détecteur :

1. Détection des extremums locaux d'intensité de l'image I ;
2. Analyse des profils d'intensité le long des rayons partants de ces extremums locaux ;
3. Sélection des extremums sur chaque rayon ;
4. Connexion de tous les extremums sur chaque rayon pour former une région fermée autour du point étudié ;
5. Approximation de cette région par une ellipse dont la taille est doublée pour définir la région d'intérêt finale.

Pour l'étape (2), plus précisément, les auteurs souhaitent explorer/analyser la région autour de chacun des extremums en étudiant une fonction de l'intensité le long des rayons issus de l'extremum local étudié. Pour chaque rayon ¹, la fonction suivante est évaluée :

$$f_I(t) = \frac{|I(t) - I(0)|}{\max(d, \frac{1}{t} \int_0^t |I(x) - I(0)| dx)}, \quad (4.11)$$

où t est l'abscisse curviligne le long du rayon et d est un nombre de valeur faible permettant d'éviter les divisions par 0. Le dénominateur permet, d'après les auteurs, d'assurer une bonne localisation des extremum.

Cet opérateur détecte des *blobs*. Dans la littérature, il s'agit d'une zone de l'image autour d'un point (d'intérêt) où les valeurs des pixels se distinguent des valeurs des pixels du voisinage. Par exemple, une zone d'une image qui représente une tache claire située dans une zone sombre constitue un *blob*. **Cette notion est assez proche de la notion de noyau abordée pour le détecteur de SUSAN, cf. § 4.4.1.** Cet opérateur est invariant par transformations affines, géométriques et photométriques.

1. À partir du point étudié, la zone circulaire autour de lui est découpée en secteur angulaire et nous prenons les rayons correspondant aux frontières de ces secteurs angulaires.

4.4.5 Vers le multi-échelle : Détecteur de [Kadir 04]

Cette approche s'appuie sur l'entropie de la distribution des valeurs des pixels au voisinage du point considéré, c'est-à-dire sur la « quantité » de désordre dans le voisinage du pixel considéré. Cela signifie que nous supposons que plus le voisinage est désordonné plus il est discriminant par rapport aux autres points de l'image et, au contraire, plus la distribution est uniforme, plus le point étudié est probablement situé dans une zone homogène et a donc peu d'intérêt. L'entropie est donnée par :

$$\mathcal{H} : \mathbb{N}^2 \rightarrow \mathbb{R}$$

$$\mathbf{p} \mapsto - \sum_{k,l} p(\mathbf{p}) \log(p(\mathbf{p})) \quad (4.12)$$

avec $(k, l) \in \mathbb{N}^2$ et $(k - i)^2 + (l - j)^2 < r^2$ où r est le rayon du voisinage considéré et où la probabilité $p(\mathbf{p})$ est donnée par la distribution des valeurs des pixels du voisinage. En pratique, un histogramme des valeurs des pixels composé de w intervalles est utilisé pour estimer p et la réponse suivante est calculée pour chaque candidat :

$$\text{KA}(\mathbf{p}) = \mathcal{H}(\mathbf{p}) \frac{\sigma^2}{2\sigma - 1} \sum_{k,l} \left| \frac{\partial p(\mathbf{p})}{\partial \sigma} \right|. \quad (4.13)$$

Dans [Kadir 04], il est montré comment cette formulation leur permet d'obtenir un détecteur invariant aux transformations photométriques (biais), aux transformations géométriques affines, ce qui rend ce détecteur indépendant du point de vue.

4.4.6 Vers le multi-échelle : EBR, *Edge-Based Regions* [Tuytelaars 04]

Les contours sont des objets assez stables par rapport aux changements de point de vue et de conditions d'éclairage. Ces détecteurs exploitent ainsi la géométrie locale : les droites et les parallélogrammes pour les *Edge-Based Regions* ou encore la courbure pour les *Curvature Scale Space*, abordé dans la partie multi-échelle, cf. § 4.5.4. Voici les quatre étapes suivies pour réaliser cette détection :

1. Coins localisés grâce au détecteur de Harris ;
2. Contours détectés avec l'approche de Canny-Deriche [Canny 86, Deriche 87] ;
3. Estimation des paramètres affines invariant le long des contours avec construction d'une famille de parallélogrammes ;
4. Sélection des parallélogrammes extrema suivant une fonction.

La principale contribution réside dans les deux dernières étapes. Plus précisément, voici le principe suivi pour l'étape (3) : Si un coin p est considéré, ainsi que deux points $p_1(l_1)$ et $p_2(l_2)$ se déplaçant le long des contours partant de ce coin, alors, la distance curviligne $l_{1,2}$ de chaque point $p_{1,2}$ au coin p est définie par la surface entre le contour et la droite joignant le coin et le point $p_{1,2}$. Cette distance est invariante aux transformations isométriques. Ces deux points et le coin définissent alors une famille de parallélogrammes, cf. figure 4.8.

Enfin, pour la dernière étape, parmi tous les parallélogrammes qui peuvent définir le coin, celui retenu est celui qui présente une valeur extremum invariante d'une fonction photométrique basée sur les moments du premier ordre de la fonction intensité. Plus précisément, les auteurs cherchent à minimiser la distance entre le centre de gravité et les diagonales du parallélogramme, comme illustré dans la figure 4.9. Quelque part, cette façon d'agir est déjà un premier pas vers une réponse invariante à l'échelle.

Cet opérateur permet de retrouver des formes parallélépipédiques, invariantes aux transformations affines géométriques et photométriques. Pour ces détecteurs, le type des primitives recherchées les destinent plutôt à des scènes structurées, comportant des éléments artificiels et bien contrastés où les frontières sont bien identifiables.

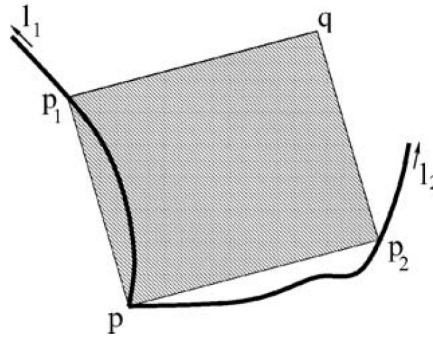


FIGURE 4.8 – Illustration du parallélogramme utilisé dans le détecteur EBR.

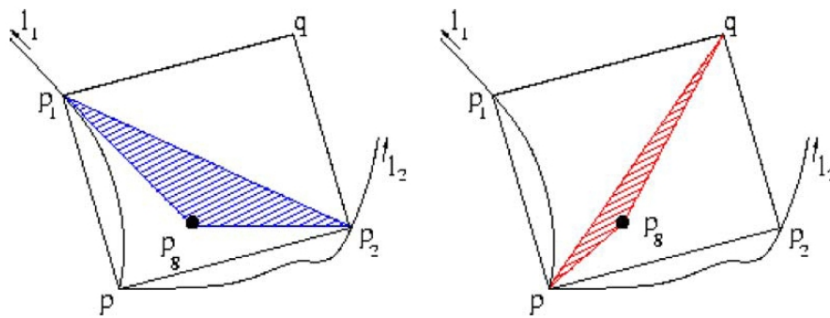


FIGURE 4.9 – Illustration des critères utilisés dans EBR.

4.4.7 Vers le multi-échelle : PCBR, *Principal Curvature-Based Regions* [Deng 07]

Ce détecteur a été introduit afin de proposer un opérateur pour la reconnaissance de classe d'objets. Son principe est de détecter des lignes de partages des eaux, à partir des courbures principales estimées à différentes échelles. Nous avons placé ce détecteur à la fin de cette section sur les détecteurs de régions, car, il est à la limite du classement que nous avons proposé, qui distingue les approches régions des approches du second ordre. Une propriété attendue par les auteurs pour ce détecteur est d'extraire des lignes de partage des eaux fiables. Pour cela, les auteurs proposent de post-traiter l'estimation des courbures principales en utilisant une fermeture morphologique combinée à un seuillage par hystérésis sur les vecteurs propres. De plus, les courbures principales ne sont conservées que si elles sont maximales sur plusieurs échelles consécutives. La façon de calculer la courbure principale, commune à de nombreuses publications, est expliquée dans la section 4.5.

Ainsi, nous avons introduit des approches prenant en compte le voisinage afin de déterminer les points les moins corrélés à leurs voisins, cf. section 4.3, puis, dans cette section, les approches qui à l'inverse, caractérise un point comme le centre d'une région cohérente. À présent, nous abordons la dernière famille de détecteurs qui prend en compte de opérateurs du second ordre. L'objectif est de prendre en considération la notion de courbure.

4.5 Détecteurs du second ordre

La plupart des détecteurs présentés dans cette dernière section s'appuie sur la notion de courbure, à l'exception de SIFT, *Scale Invariant Feature Transform*, que nous présentons en dernier. La première

catégorie repose sur le fait que nous cherchons à déterminer les points saillants en terme de courbure, c'est-à-dire qui présentent un brusque changement de courbure, en supposant que cette caractéristique les rend discriminants par rapport aux autres points de l'image. Cette notion de courbure est très commune dans la littérature, notamment dans la recherche de primitives en 3D. L'intérêt de ce type d'approches est notamment de faire le lien avec la notion de points saillants d'une forme. Nous nous attachons donc plus ici à la notion de forme que de texture, comme c'est le cas avec les détecteurs basés auto-corrélation. La seconde catégorie, essentiellement liée au détecteur SIFT, est l'utilisation du gradient du second ordre à différentes échelles. Cela veut dire que nous nous intéressons aux points d'intérêt le long des contours (plutôt de type toit).

4.5.1 Beaudet ou Hessien [Beaudet 78]

Ce détecteur s'appuie sur les dérivées secondes et donc sur la notion de courbure de la surface. Le principe est de minimiser le déterminant de la matrice hessienne, soit :

$$\text{Beaudet}(x, y) = \det \begin{pmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{pmatrix} = I_{xx}I_{yy} - I_{xy}^2. \quad (4.14)$$

En effet, il représente la courbure de la surface constituée par les niveaux de gris (ou couleurs) de l'image pour chaque pixel étudié.

4.5.2 Vers le multi-échelle : Hessien-Laplace [Mikolajczyk 04]

La version multi-échelle du détecteur de Beaudet a également été mise en œuvre en utilisant exactement le même principe que celui exposé au § 4.3.2 mais en l'appliquant au détecteur de Beaudet. Il s'agit du détecteur Hessien-Laplace.

4.5.3 Kitchen et Rosenfeld [Kitchen 82]

La réponse dépend également de la courbure, mais, cette fois, c'est la courbure du contour passant par le point considéré. De plus, chaque point utilisé est pondéré par la norme du gradient car les auteurs souhaitent prendre en compte les cas où la norme du gradient est faible (donc un contour peu contrasté, qui peut correspondre à un bruit). Ainsi, la formulation suivante est obtenue :

$$\text{KR}(x, y) = \frac{I_{xx}I_y^2 - 2I_{xy}I_xI_y + I_{yy}I_x^2}{I_x^2 + I_y^2}. \quad (4.15)$$

4.5.4 CSS, *Curvature Scale Space* [Mokhtarian 98]

Le détecteur en courbure et espace d'échelle, *Curvature Scale Space (CSS)* [Mokhtarian 98] recherche les extrema de la courbure le long des contours en utilisant une approche multi-échelle. Il détecte d'abord les contours en utilisant l'opérateur de Canny, fusionne les contours très proches, puis calcule la courbure $K(t, \sigma)$ de ce contour lissé par une gaussienne. Ceci définit l'espace d'échelle de courbure dont les coins sont assimilés à des extrema. Les candidats dont la réponse est trop faible ou trop proches les uns des autres ne sont pas retenus. Une variante de ce détecteur différencie le traitement des contours courts et des contours longs en déterminant automatiquement le seuil de décision. Cet opérateur est invariant par transformations affines géométriques et photométriques.

Le principal inconvénient des détecteurs basés sur les contours est la définition des paramètres utilisés par le détecteur de contours. Bien que tous les détecteurs soient paramétrés, la non-détection des contours s'avère être un handicap important.

4.5.5 Vers le multi-échelle : SIFT, Scale Invariant Feature Transform [Lowe 04]

Principe – La méthode SIFT, *Scale Invariant Feature Transform*, s'appuie sur la **détection des extremums en échelle et en espace**, notés (x, y, σ) , du laplacien en (x, y, σ) . Une fois ces extremums extraits, ils sont **caractérisés par une orientation principale** qui correspond au **gradient dominant dans le voisinage du point considéré avec un taux d'échantillonnage proportionnel à $\sqrt{\sigma^2 + c^2}$** où c correspond au taux de bruit supposé présent dans l'image. Les auteurs font l'hypothèse que pour chaque image traitée, $c = 0.5$. Le descripteur associé à ce détecteur prend en compte un **histogramme local de ces orientations**.

Propriétés – Les points d'intérêt obtenus sont **bien localisés aussi bien dans le domaine spatial que dans le domaine fréquentiel**, on peut dire qu'ils sont relativement **robustes aux occultations et aux bruits**. Étant donné que chaque point détecté est caractérisé par l'orientation du gradient qui est invariante aux changements de contraste, cela permet une certaine **invariance aux changements de luminosité**. De plus, l'utilisation de l'histogramme local permet une certaine **robustesse aux changements de points de vue**.

Enfin, ce détecteur est complet dans le sens où les auteurs lui ont associé un descripteur permettant de décrire chacun des points et ainsi de les mettre en correspondance de manière robuste.

Étapes – Le détecteur proposé s'appuie sur les 4 grandes étapes suivantes :

- (1) *Détection des extremums en échelle et en espace* : l'objectif est d'estimer un critère, ici le gradient de l'image, dans toutes les échelles et en tous points.
- (2) *Localisation des points* : en s'appuyant sur un modèle, il s'agit de déterminer les points stables suivant ce modèle et ainsi d'en déduire les points d'intérêt.
- (3) *Affectation d'une orientation* : une orientation principale est attribuée à chaque point.
- (4) *Estimation des descripteurs* : cela est réalisé en prenant en compte un voisinage.

Les deux premières étapes correspondent à la détection des points d'intérêt alors que les deux dernières permettent le calcul d'un descripteur associé au point d'intérêt afin de réaliser la mise en correspondance ou le suivi, ils seront donc détaillés dans la partie concernée sur la mise en correspondance, cf. chapitre IV.

Détection des extremums en échelle et en espace – Il s'agit d'une approche par **filtres « en cascade »**. On cherche à détecter les formes/points stables en échelles. Pour cela, on s'appuie sur un modèle/un noyau Gaussien et on suit les étapes suivantes :

- (1) Si on note $L(x, y, \sigma)$ les images filtrées à l'échelle σ , on a :

$$L(x, y, \sigma) = G(x, y, \sigma) \otimes I(x, y) \quad (4.16)$$

où $G(x, y, \sigma)$ est la fonction gaussienne.

- (2) Pour détecter les points stables, les auteurs estiment par la suite des différences de gaussiennes de la manière suivante :

$$DoG(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) = L(x, y, k\sigma) - L(x, y, \sigma). \quad (4.17)$$

Cette différence de gaussienne n'est autre qu'une **approximation du laplacien**. En l'effectuant pour un ensemble de valeurs successives de σ , on obtient donc le laplacien normalisé en échelle. Un ensemble d'échelles correspond à une octave. Pour passer d'une octave à l'autre, on suppose que l'on double la valeur de σ . Ainsi, si on souhaite n échelles par octave, alors il faut fixer $k = 2^{1/n}$.

- (3) Enfin, les maxima/minima locaux pour chaque octave sont sélectionnés en prenant en compte un voisinage 3×3 dans l'image ainsi que celui de l'échelle inférieure et de l'échelle supérieure.

Expérimentalement, les auteurs ont montré qu'il suffisait d'avoir 3 échelles par octaves. En effet, il s'agit du meilleur compromis : coût/détection d'extremums locaux. De plus, il a été démontré que plus σ est élevée, plus les extremums retenus seront **répétables, c'est-à-dire qu'on détectera les mêmes points quel que soit le changement entre les deux points de vue et de la même manière, empiriquement, les auteurs ont choisi $\sigma = 1.6$.**

Pour résumer, si l'on souhaite n échelles pour sélectionner les minima/maxima locaux, il est nécessaire de calculer $n+1$ différences de gaussiennes, soit $n+2$ images filtrées. Sachant que la dernière image d'une octave et la première image de l'octave suivante ne peuvent pas être les mêmes, il est donc nécessaire de calculer $n+3$ images filtrées, cf. figure 4.10.

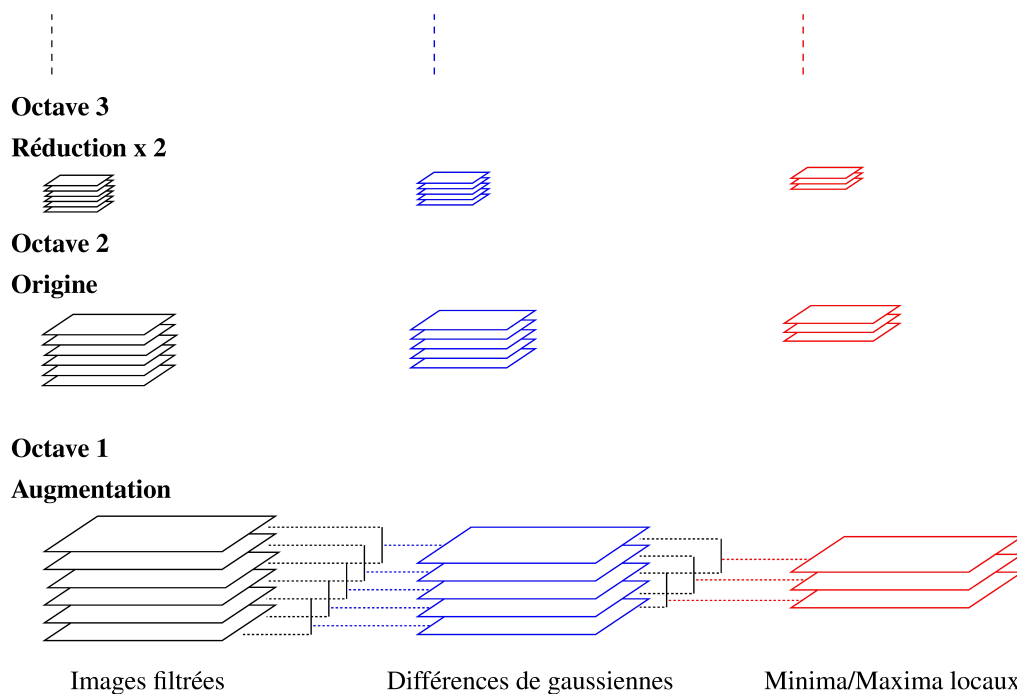


FIGURE 4.10 – Octaves et échelles pour le détecteur SIFT – Pour cette exemple, on suppose qu'il y a 3 échelles ainsi 6 images filtrées seront évaluées par octave.

Pour terminer, l'algorithme de calcul des minima/maxima locaux à travers les octaves et échelles est donc celui exposé dans la figure 4.11.

Il est important de noter que les auteurs supposent la présence d'un bruit gaussien avec un $\sigma = 0.5$, ce σ n'a rien à voir avec celui utilisé pour calculer les octaves. Ainsi, pour le calcul de la pyramide d'images, pour contrecarrer les effets d'un éventuel bruit gaussien au départ avec un $\sigma = 0.5$ (tendance à supprimer les petits détails), les auteurs suggèrent de **doubler la résolution de l'image au départ**. Ensuite seulement, la résolution de l'image est réduite par 2 à chaque octave. Là aussi, le fait d'ajouter cette étape **permet de renforcer la répétabilité du détecteur**.

Position sub-pixellique ou relocalisation – Pour contrecarrer les effets des lissages et le fait que l'on perd de la précision sur la localisation des points, les points extraits sont « relocalisés », *i.e.* leur position est calculée au sous-pixel. Pour cela, on utilise une approche s'appuyant sur le

- (1) Calculer une pyramide gaussienne d'images
- (2) Pour chaque octave o (chaque image de la pyramide) faire
 Pour chaque valeur de k faire
 Calculer en chaque point (x, y) , $G(x, y, k\sigma)$
- (3) Pour chaque octave o faire
 Pour chaque échelle $s \in [2, n]$ faire
 Calculer en chaque point (x, y) , $DoG(x, y, \sigma)$
- (4) Pour chaque octave o faire
 Pour chaque échelle $s \in [2, n - 1]$ faire
 Extraire les minima/maxima locaux

FIGURE 4.11 – Algorithme de détection des points SIFT.

développement de Taylor à l'ordre 2, soit :

$$D(\hat{\mathbf{p}}) = D(\mathbf{p}) + \frac{\partial D(\mathbf{p})^T}{\partial \mathbf{p}} \mathbf{p} + \frac{1}{2} \mathbf{p}^T \frac{\partial^2 D(\mathbf{p})}{\partial \mathbf{p}^2} \mathbf{p}, \quad (4.18)$$

La localisation précise de l'extremum est donnée par le passage par zéro de la dérivée suivant \mathbf{p} , soit :

$$\hat{\mathbf{p}} = -\frac{\partial^2 D(\mathbf{p})^{-1} \partial D(\mathbf{p})}{\partial \mathbf{p}^2 \partial \mathbf{p}}. \quad (4.19)$$

Si le décalage est plus grand que 0.5 dans chaque direction, alors, on suppose qu'il est nécessaire de réaliser la relocalisation sinon on garde le point initial. Dans le cas où il faut relocaliser, cela signifie qu'il faut d'abord prendre le point de l'image, en coordonnées entières, le plus proche de $\hat{\mathbf{p}}$, que nous notons \mathbf{p}' . Enfin, il faudra appliquer le processus de relocalisation à \mathbf{p}' , en suivant l'équation (4.19) pour obtenir le point d'intérêt final, noté $\hat{\mathbf{p}}'$.

Filtrage des minima/maxima locaux – Afin, les auteurs ont ajouté deux contraintes pour la sélection des minima/maxima locaux :

- (1) Une fois les points re-localisés, on élimine ceux dont le contraste est trop faible. Plus précisément, si on substitue l'équation (4.19) dans l'équation (4.18), on obtient :

$$D(\hat{\mathbf{p}}) = D(\mathbf{p}) + \frac{1}{2} \frac{\partial D(\mathbf{p})^T}{\partial \mathbf{p}} \hat{\mathbf{p}}. \quad (4.20)$$

Et dans leur expérimentations, les auteurs ont éliminé tous les points pour lesquels $|D(\hat{\mathbf{p}})| < 0.03$, en supposant une variation entre 0 et 1.

- (2) Les points sur les contours sont rejetés. Pour cela, les auteurs utilisent la matrice hessienne, comme pour l'approche de Harris. La courbure principale dans la direction normale à un contour est forte et celle dans la direction du contour est faible. Les courbures principales étant proportionnelles aux valeurs propres de la matrice hessienne, seuls les points où le rapport entre la plus grande et la plus petite valeur propre est inférieur à un seuil r sont conservés. Plus formellement, il faut vérifier :

$$\frac{\text{Trace}(M)^2}{\text{Det}(M)} < \frac{(r+1)^2}{r}. \quad (4.21)$$

Dans leurs travaux, les auteurs ont pris $r = 10$.

4.5.6 Vers une variante accélérée de SIFT : SURF, Speeded Up Robust Features [Bay 08]

Ce détecteur utilise comme réponse une approximation du déterminant de la matrice hessienne à plusieurs échelles. L'intérêt de ce nouveau détecteur réside surtout dans le fait que les auteurs proposent une implémentation astucieuse. Plus précisément, c'est le calcul de cette approximation du Hessien qui le rend beaucoup plus rapide que les autres détecteurs de la littérature. Par exemple, SURF est moins coûteux en temps de calculs que SIFT. Le lissage gaussien est réalisé par des *box filter* et une image intégrale est alors utilisée pour calculer la convolution.

Les auteurs ont remarqué qu'en pratique, le calcul des dérivées avec un masque de convolution constitue déjà une approximation pour les autres approches de la littérature. Toutefois, ils remarquent également que cette perte de précision n'empêche pas les détecteurs basés laplacien de donner de bons résultats en terme de répétabilité (les mêmes points d'intérêt sont retrouvés d'un point de vue à l'autre) et proposent d'utiliser directement des approximations moins coûteuses en temps de calculs. Pour cela, ils introduisent l'utilisation de masques de dérivation au second ordre lissés par une gaussienne et tronqués de telle sorte que les calculs puissent être réalisés à l'aide d'images intégrales.

Image intégrale – Chaque point de l'image est la somme des niveaux de gris des points sur les lignes et colonnes précédentes. L'intérêt de ces images est de pouvoir faire par la suite des calculs très rapides. Plus formellement, ces images sont définies par :

$$I_{int}(i, j) = \sum_{k=0}^{i-1} \sum_{l=0}^{j-1} I(k, l). \quad (4.22)$$

Cette image permet de calculer facilement la somme de toutes les valeurs de I situées dans un rectangle délimité par trois opérations arithmétiques, cf. illustration dans la figure 4.12 :

$$\sum_{i=a}^b \sum_{j=c}^d I(i, j) = I_{int}(b, d) - I_{int}(a, d) - I_{int}(b, c) + I_{int}(a, c). \quad (4.23)$$

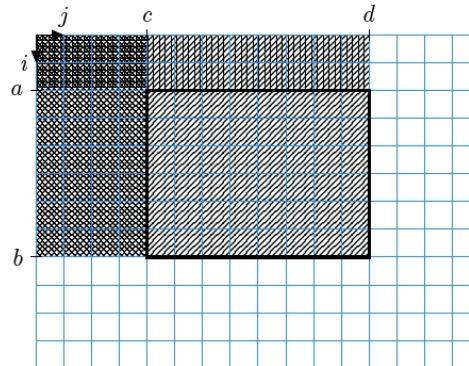


FIGURE 4.12 – Utilisation de l'image intégrale.

Box filtering – Les masques utilisés pour calculer les dérivées sont construits de telle sorte qu'ils ne correspondent qu'à des combinaisons de sommes de valeurs de pixels situés dans différents rectangles, *i.e.*, les *box*. Ainsi, l'image intégrale peut être utilisée pour réduire le nombre d'opérations arithmétiques, cf. figure 4.13.

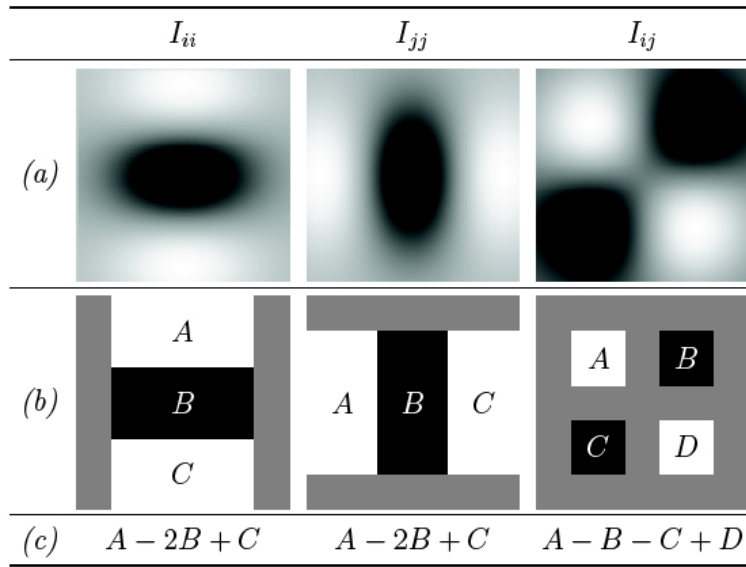


FIGURE 4.13 – Illustration du *Box filtering* – La ligne (a) montre les images des masques de convolution pour calculer I_{ii} , I_{jj} et I_{ij} en un point donné. Plus le niveau de gris est élevé, plus la valeur du masque est élevée (ces valeurs pouvant être négatives, le gris représente 0). Les valeurs de ces masques sont approchées et tronquées pour obtenir les masques de la ligne (b). Ainsi, le calcul de la convolution peut être remplacé par l'opération indiquée en (c). A est la somme des valeurs des pixels de l'image inclus dans le rectangle A lorsque le masque est centré sur p. Cette somme peut être calculée rapidement à l'aide de l'image intégrale.

Il existe une autre variante de ce type de détecteurs, permettant d'améliorer encore le temps de calculs, ils s'agit du détecteur BRISK, *Binary Robust Invariant Scalable Keypoints*, qui s'appuie sur des motifs binaires. Le lecteur intéressé pourra trouver les détails dans la publication suivante [Leutenegger 11].

4.6 Descripteurs de points d'intérêt en 2D

Jusqu'à présent, parmi tous les détecteurs que nous avons présentés, seulement deux correspondent à ce que nous appelons des détecteurs complets. Un détecteur complet fournit un opérateur de détection ainsi qu'un descripteur du point étudié, c'est le cas pour les détecteurs SIFT, cf. § 4.5.5, et SURF, cf. § 4.5.6. Dans cette section, nous présentons, d'une part, les descripteurs associés à ces détecteurs, et, d'autre part, d'autres outils qui permettent de décrire un point, sans fournir un détecteur.

4.6.1 Descripteur associé à SIFT et SURF

L'objectif du descripteur proposé pour effectuer la mise en correspondance est d'être suffisamment discriminant/unique pour permettre une mise en correspondance la plus fiable possible. C'est la raison pour laquelle les auteurs introduisent un descripteur de dimension 128 alors que la plupart des autres approches s'appuient sur des données 1D (niveaux de gris), 3D (couleurs) voire 5D (couleurs + position).

Affectation d'une orientation – Une fois les extremums détectés, les auteurs attribuent des orientations principales à chaque extremum en suivant les points suivants :

- (1) *Estimation de la norme et de l'orientation du vecteur gradient* – Cela est calculé simplement avec :

$$m(x, y) = \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2}$$

$$\theta(x, y) = \tan^{-1} \left(\frac{L(x, y+1) - L(x, y-1)}{L(x+1, y) - L(x-1, y)} \right) \quad (4.24)$$

- (2) *Construction d'un histogramme **local** des orientations θ* – Pour le calcul de cet histogramme, la somme des éléments de même orientation est pondérée par la norme m et par une gaussienne, centrée sur le point étudié avec $\sigma = 1.5$. L'objectif de cette pondération est de minimiser l'influence des voisins les plus éloignés ainsi que l'influence des voisins dont le gradient est faible. En pratique, l'histogramme est discrétisé (les auteurs utilisent 36 orientations pour couvrir les 360 degrés), dans la figure 8.1.(a), nous donnons un exemple d'histogramme sur 8 orientations uniquement. De plus, le voisinage pris en compte est de taille 8×8 .
- (3) *Sélection des orientations* – La plus grande valeur de cet histogramme, l'orientation principale, ainsi que toutes celles au dessus de 80% de ce maximum, les orientations secondaires sont considérées comme les orientations du point. Ainsi, nous pouvons avoir plusieurs orientations pour le même point. En pratique, les auteurs signalent que seuls 15% au maximum des points, se voient attribués plusieurs orientations.
- (4) *Interpolation des orientations* – Pour mieux localiser le maximum, une interpolation est réalisée au voisinage de chaque maximum, en prenant en compte les 3 valeurs les plus proches de ce maximum, cf. figure 8.1.(b).

Calcul du descripteur – Le descripteur proposé s'inspire du modèle visuel primaire humain qui est plus sensible aux changements d'orientation du gradient qu'aux déplacements (pour permettre la reconnaissance d'un objet quelque soit le point de vue). Ainsi, le descripteur utilisé est construit de la manière suivante :

- (1) *Utilisation de l'orientation et de l'histogramme déjà estimés* – Les mêmes données, cf. figure 8.1.(a), sont utilisés, à trois différences près :
- (a) L'histogramme ne possède plus 36 mais seulement 8 orientations.
 - (b) Une rotation, relative à l'orientation du point d'intérêt, est appliquée aux coordonnées ainsi qu'à l'orientation de chaque point (ce qui est estimé comme dans la figure 8.1.(b)). Ceci, dans le but de rendre la description invariante aux rotations.
 - (c) La participation de chaque point dans le calcul de l'histogramme local est pondérée par sa distance au point d'intérêt caractérisé (et non plus de simple poids gaussiens), c'est-à-dire par $1 - d$ dans chaque dimension où d est la distance du point au point central.
- (2) *Première normalisation du vecteur-descripteur* – Les valeurs sont recadrées entre 0 et 1 afin que le descripteur soit invariant aux changements affines d'illumination.
- (3) *Seuillage des forts gradients* – Cette étape est réalisée afin de prendre en compte des changements d'illumination non-linéaires. Les auteurs ont expérimentalement choisi un seuil très restrictif à 0.15. Ceci implique la normalisation suivante.
- (4) *Deuxième normalisation du vecteur-descripteur* – De la même manière que la première normalisation.

Les auteurs proposent un descripteur de taille 128 en utilisant un voisinage 4×4 et 8 orientations.

Appariement – Pour la mise en correspondance, une simple distance peut être utilisée. Dans la littérature, nous pouvons citer l'utilisation d'un produit scalaire (entre les deux descripteurs de 128), de la mesure ZNCC ou d'une distance L_2 .

4.6.2 HoG, *Histogram of Oriented Gradient* [Dalal 05]

Cette approche comporte de grandes similitudes avec le descripteur SIFT et c'est pourquoi nous commencerons par ce descripteur. En effet, le principe est de caractériser un point par la distribution locale des orientations des gradients dans son voisinage. Donc, de la même manière que SIFT, il est nécessaire de calculer des histogrammes locaux des orientations des gradients. Plus précisément les étapes de construction de ce détecteur sont :

1. Division du voisinage du point en sous-imagettes régulières, appelées cellules ;
2. Calcul des histogrammes des orientations des gradients pour chaque cellule ;
3. Normalisation des histogrammes pour les rendre robustes aux changements d'illumination ;
4. Combinaison des histogrammes pour former le descripteur.

L'étape (3) est optionnelle. L'avantage de cette approche est de faire intervenir des outils simples de la vision. En effet :

- Les calculs se font sur les images en couleur, avec le système *RGB*.
- Les dérivées sont calculées par simple différence (avec les filtres $[-1, 0, 1]$ et $[-1, 0, 1]^T$ et il n'est pas nécessaire de filtrer les images au préalable).
- Les histogrammes sont calculés sur 12 bins et en pondérant par l'intensité du gradient (comme SIFT).
- La normalisation est réalisée sur des ensembles de cellule, que les auteurs appellent des blocs.
- Il est conseillé de prendre des cellules de taille 6×6 (c'est plus petit que pour SIFT) et des blocs de 3×3 .

Ainsi, au final, un descripteur HoG classique sera de taille 12 (le nombre de bins par cellule) $\times 3 \times 3$ (le nombre de cellule par bloc) = 108.

4.6.3 BRIEF, *Binary Robust Independent Elementary Features* [Calonder 10]

Le principe de ce descripteur est de construire une chaîne de bits, relativement courte à l'opposé des descripteurs SIFT ou HoG, permettant de résumer la région autour du point pour ensuite utiliser une distance de Hamming pour effectuer les comparaisons. Un descripteur court pouvant être simplement comparé à un autre, doit, d'après les auteurs, permettre d'être plus rapide et plus efficace que les descripteurs traditionnellement utilisés (SIFT, HoG) et classiquement comparé en utilisant la norme L_2 ou la mesure ZNCC.

Le principe utilisé pour construire la chaîne de bits repose sur la simple comparaison des intensités le long de lignes. Plus précisément, les auteurs proposent le codage suivant, avec (x_i, y_i) et (x'_i, y'_i) les coordonnées de deux pixels dans le voisinage du pixel (x, y) étudié :

$$\text{BRIEF}(x, y) = \sum_{1 \leq i \leq N_b} 2^{i-1} \tau(x_i, y_i, x'_i, y'_i), \quad (4.25)$$

avec N_b le nombre de bits formant le descripteur. De plus la fonction τ est définie par :

$$\tau(x, y, x_i, y_i) = \begin{cases} 1 & \text{si } I(x, y) < I(x_i, y_i) \\ 0 & \text{sinon.} \end{cases} \quad (4.26)$$

Les deux éléments à fixer sont : N_b le nombre de bits (les auteurs ont testé 128, 256 et 512) et surtout le sous-ensemble de points à prendre en compte dans le voisinage. Pour ce dernier aspect, les auteurs ont testés 5 stratégies différents : un tirage suivant une loi uniforme, suivant une loi gaussienne en (x, y) , une loi gaussienne séparément sur chaque canal (en x puis en y), un tirage aléatoire suivant une grille du cercle polaire pour introduire une forme de quantification, une répartition uniforme sur une grille du cercle polaire. Avant de calculer ce descripteur et afin d'être robuste aux bruits dans l'image, les auteurs proposent de filtrer l'image au préalable (avec un filtre gaussien par exemple).

Il est important de noter qu'actuellement, une approche utilisant BRIEF a été développée. Il s'agit du détecteur et descripteur ORB, *Oriented FAST and Rotated BRIEF* [Ruble 11]. Cette approche combine le détecteur FAST que nous avons présenté au § 4.4.2 avec le descripteur BRIEF. Les auteurs font remarquer que ce détecteur et ce descripteur ne sont pas robustes au changement d'orientation des primitives suivies, comme peuvent l'être SIFT ou SURF. Ils ajoutent donc une façon de prendre en compte l'orientation aussi bien pour le détecteur FAST que pour le descripteur BRIEF. Pour cela, voici les étapes suivies pour l'amélioration des points détectés par FAST :

1. Calcul de N points d'intérêt en utilisant FAST avec un rayon de 9 ;
2. Ordonnancement de ces points en utilisant la mesure de Harris ;
3. Calcul de l'orientation de chaque point en utilisant les moments associés à chaque région circulaire de chaque point sélectionné.

Afin d'être robuste au changement d'échelle, tous ces calculs sont réalisés pour toutes les images d'une pyramide d'images. Enfin, le descripteur BRIEF peut être amélioré en prenant en compte l'orientation associée à chaque point d'intérêt. De plus, les auteurs ont ajouté une seconde phase d'apprentissage pour réduire la présence de points d'intérêt avec une grande corrélation entre eux et ainsi augmenter la variance entre points d'intérêt détectés.

4.6.4 LBP, *Local Binary Pattern* [Ojala 02]

Les motifs locaux binaires, ou *Local Binary Pattern*, LBP, ont été introduits dans le contexte de la classification de texture. Le principe consiste à prendre en considération la distribution des pixels dans un voisinage circulaire, plus précisément, les niveaux de gris des pixels se situant sur un cercle, dont il faut définir le rayon et le pas d'échantillonnage. Les niveaux de gris de cette distribution sont centrés par rapport au niveau de gris du pixel central, en remplaçant toutes les valeurs négatives par 0 et toutes valeurs positives par 1. Ce centrage doit permettre une première invariance aux changements de niveaux de gris. Enfin, ces valeurs sont accumulées en utilisant un facteur binomial associé à chaque niveau, afin de former un code binaire, et ainsi, si nous considérons \mathbf{p} le point pour lequel nous calculons le descripteur, nous définissons :

$$\text{LBP}(\mathbf{p}, P, R) = \sum_{p=0}^{P-1} s(I(\mathbf{p}') - I(\mathbf{p}))2^p, \quad (4.27)$$

avec P le nombre de pixels considérés pour le motif local, placés sur un cercle de rayon R . Afin d'être invariant aux rotations, les auteurs effectuent $(P - 1)$ décalage du code LBP de l'équation (4.27) afin de sélectionner celui de plus faible valeur. De plus, les auteurs identifient le nombre de transitions dans le code (c'est-à-dire le nombre de fois où la valeur de deux bits consécutifs est différente) et considèrent qu'au delà de deux transitions, tous les motifs sont égaux à $P = 1$. Au final, le descripteur associé à une région est l'histogramme de tous les codes obtenus dans son voisinage.

4.6.5 Détecteur s'appuyant sur la covariance [Tuzel 06]

Tous les descripteurs présentés jusque là possèdent de nombreux éléments et sont donc de taille importante, ce qui rend leur utilisation assez coûteuse. Une autre façon de rendre leur estimation et leur utilisation moins coûteuse est de, d'une part de proposer des traitements rapides ou accélérés, et, d'autre part, de proposer un descripteur contenant moins d'éléments. C'est le cas du détecteur proposé dans [Tuzel 06] où les auteurs proposent d'utiliser seulement les matrices de covariance des régions d'intérêt étudiés en prenant en compte neuf éléments : position, couleur, dérivées premières et secondes suivant les lignes et les colonnes. De plus, des techniques d'accélération des calculs sont utilisées, notamment en s'appuyant sur les images intégrales, cf. section 4.5.6.

Quatrième partie

Mise en correspondance

Chapitre 5

Introduction

5.1 Définition

La mise en correspondance ou appariement a pour objectif de trouver les entités homologues, c'est-à-dire, les éléments qui se correspondent entre deux images [Scharstein 02a].

De cette définition découle des propriétés que l'on cherche généralement à retrouver. En effet, on souhaite qu'un point d'intérêt :

- (1) Permette une mise en correspondance fiable – Par exemple, des points de **motifs répétitifs** ne sont pas évidents à mettre en correspondance. Ce cas est plutôt facile à éviter. En revanche ce qui est plus difficile c'est d'éviter de détecter des **points occultés**. Mais ceci peut être corrigé *a posteriori*.
- (2) Respecte un critère de répétabilité – C'est-à-dire qu'un point particulier doit être détecté dans toutes les images de la scène.

Ainsi, nous nous plaçons dans le cas où la mise en correspondance est effectuée de l'image gauche vers l'image droite mais toutes les explications et les formules peuvent être utilisées dans le cas d'une mise en correspondance de la droite vers la gauche. Une manière de représenter le résultat d'une mise en correspondance consiste à associer à chaque pixel $\mathbf{p}_g^{i,j} = (i \ j)^T$ de l'image gauche un vecteur appelé disparité, défini par ses composantes $(u - i \ v - j)^T$ où $(u \ v)^T$ sont les coordonnées dans l'image droite du point correspondant au pixel $\mathbf{p}_g^{i,j}$.

Nous pouvons assimiler le problème de la mise en correspondance à la recherche d'une fonction de disparité d qui attribue une disparité à chaque pixel $\mathbf{p}_g^{i,j}$. Dans le cas général, nous avons :

$$\begin{aligned} d : \mathbb{N}^2 &\longrightarrow \mathbb{R}^2 \\ \mathbf{p}_g^{i,j} &\longmapsto d(\mathbf{p}_g^{i,j}) = (u - i \ v - j)^T. \end{aligned} \tag{5.1}$$

Lorsqu'une disparité réelle est calculée, il s'agit d'un appariement au sous-pixel. La plupart des méthodes effectuent une mise en correspondance de pixel à pixel, c'est-à-dire qu'elles calculent des disparités entières, et dans ce cas, nous avons :

$$d : \mathbb{N}^2 \rightarrow \mathbb{Z}^2.$$

5.2 Éléments constitutants de la mise en correspondance

Pour caractériser les méthodes de mise en correspondance, nous avons identifié les éléments constitutants suivants :

- (1) Les **primitives à appairer et leurs attributs ou descripteurs** – Nous distinguons deux catégories :

- Les **pixels de l'image** – Nous pouvons, par exemple, considérer tous les pixels de l'image ou seulement une partie, comme des points d'intérêt. Leurs attributs sont la plupart du temps les niveaux de gris ou les composantes couleur.
 - Les **primitives structurées pour le *feature-based matching*** – Nous pouvons distinguer : les segments (Les attributs généralement utilisés sont, entre autres, la position, l'orientation, la longueur, les niveaux de gris du segment, le contraste sur le segment et les relations avec les segments adjacents), les contours (niveaux de gris, composantes couleur ou gradients des pixels) et les régions (niveaux de gris ou les couleurs de la région, la variance des niveaux de gris ou des moments).
- (2) Le **coût global de mise en correspondance** – Lorsqu'un ensemble de correspondances entre primitives est établi, il est nécessaire de **déterminer à quel point cet ensemble de correspondances est en accord avec le modèle choisi**, c'est-à-dire à quel point les hypothèses sont vérifiées et les contraintes sont satisfaites. Pour cela, un coût global de mise en correspondance est associé à l'ensemble des correspondances. Ce coût se compose de deux termes :
- (a) Le **coût de correspondance** – Chaque coût local permet de déterminer la dissimilarité entre deux primitives correspondantes.
 - (b) Le **coût des contraintes** – Il quantifie le degré de respect des contraintes prises en compte pour l'ensemble des correspondances.

Par exemple, dans le cas où les primitives sont des pixels et où les niveaux de gris sont leurs attributs, le coût local peut consister en une simple fonction de la différence de niveaux de gris et le coût de voisinage en la distance qui sépare le correspondant et le correspondant de l'un des pixels voisins. Plus précisément, si nous notons \mathcal{S} l'ensemble des pixels de I_1 pour lesquels nous cherchons un correspondant, alors nous souhaitons minimiser une énergie entre les images I_1 et I_2 qui a la forme suivante :

$$E_{\text{global}}(d) = (1 - \lambda)E_{\text{correspondance}}(d) + \lambda E_{\text{contrainte}}(d), \quad (5.2)$$

où $\lambda \in [0; 1]$ permet d'ajuster l'influence des poids entre les deux termes :

- $E_{\text{correspondance}}$ qui est le coût de correspondance ou terme d'attache aux données.
 - $E_{\text{contrainte}}$ qui est le coût des contraintes modélisant les interactions entre les pixels considérés.
- Résoudre le problème de la mise en correspondance, formulé de cette manière, consiste à trouver la fonction de disparité d qui minimise ce coût E_{global} .
- (3) Les **zones d'agrégation** – Elles concernent :
- Le **coût local** – Des méthodes prennent en compte des primitives voisines aux primitives étudiées pour calculer le coût local. L'ensemble des primitives prises en compte constitue la zone d'agrégation.
 - Le **coût de voisinage** – Dans la plupart des cas, la zone d'agrégation associée au coût de voisinage est différente de celle associée au coût local.
- (4) La **méthode d'optimisation** – Il s'agit de la méthode utilisée pour trouver la valeur minimale du coût global, dans le cas des méthodes globales, ou des coûts locaux, dans le cas des méthodes locales, et déterminer ainsi les couples de primitives retenues comme correspondantes.
- (5) L'**affinement des résultats** – Après avoir établi les correspondances, les résultats peuvent aussi être traités, notamment pour tenter de corriger *a posteriori* d'éventuelles erreurs.

Nous allons à présent reprendre en détails les éléments du coût global.

5.3 Coût de correspondance

Ce terme évalue à quel point un ensemble de correspondances est fiable. Il est déterminé par la somme sur le support \mathcal{S} des coûts locaux et il est de la forme :

$$E_{\text{correspondance}}(d) = \sum_{\mathbf{p}_g^{i,j} \in \mathcal{S}} E_{\text{local}}(\mathbf{p}_g^{i,j}, \mathbf{p}_d^{u,v}) \quad (5.3)$$

avec $\mathbf{p}_d^{u,v} = \mathbf{p}_g^{i,j} + d(\mathbf{p}_g^{i,j})$.

Le coût local correspond au coût d'une correspondance et il est défini par :

$$E_{\text{local}}(\mathbf{p}_g^{i,j}, \mathbf{p}_d^{u,v}) = \sum_{\substack{\mathbf{p}_g^{i',j'} \in \text{ZA}(\mathbf{p}_g^{i,j}) \\ \mathbf{p}_d^{u',v'} \in \text{ZA}(\mathbf{p}_d^{u,v})}} E_{\text{dissimilarité}}(\mathbf{p}_g^{i',j'}, \mathbf{p}_d^{u',v'}), \quad (5.4)$$

où $\text{ZA}(\mathbf{p}_g^{i,j})$ correspond à la zone d'agrégation considérée (pixels pris en compte pour le calcul du coût local). La zone d'agrégation d'un pixel $\mathbf{p}_g^{i,j}$ est le plus souvent un voisinage de $\mathbf{p}_g^{i,j}$, c'est-à-dire qu'elle correspond à un ensemble de pixels connexes qui contient $\mathbf{p}_g^{i,j}$. Un exemple classique de voisinage est une fenêtre carrée de taille $(2t+1) \times (2t+1)$ centrée sur $\mathbf{p}_g^{i,j}$. Le terme $E_{\text{dissimilarité}}$ est le coût de dissimilarité : il évalue à quel point deux pixels ne se ressemblent pas.

5.3.1 Coût des contraintes

Il modélise les contraintes entre les pixels du support \mathcal{S} et les pixels de la zone d'agrégation. Il y a N_c contraintes et ainsi N_c zones d'agrégation associées que nous notons ZAC car cette zone est différente de la zone d'agrégation associée au coût local, ZA, et elle correspond la plupart du temps à un voisinage. Le coût des contraintes est donné par :

$$E_{\text{contrainte}}(d) = \sum_{c=0}^{N_c-1} E_{\text{contrainte}}^c(d) = \sum_{c=0}^{N_c-1} \sum_{\mathbf{p}_g^{i,j} \in \mathcal{S}} E_{\text{voisinage}}^c(\mathbf{p}_g^{i,j}), \quad (5.5)$$

Le coût du voisinage permet d'évaluer l'effet de la contrainte sur le pixel étudié et ses voisins et il est défini par :

$$E_{\text{voisinage}}^c(\mathbf{p}_g^{i,j}) = \sum_{\mathbf{p}_g^{i',j'} \in \text{ZAC}^c(\mathbf{p}_g^{i,j})} E_{\text{lissage}}^c(\mathbf{p}_g^{i,j}, \mathbf{p}_g^{i',j'}). \quad (5.6)$$

Le terme E_{lissage}^c permet de comparer les disparités associées aux deux pixels considérés et il est appelé coût de lissage. La zone d'agrégation associée à la $c^{\text{ème}}$ contrainte est notée ZAC^c .

5.3.2 Coût global développé

Nous avons présenté les deux termes du coût global de mise en correspondance, nous pouvons à présent reprendre l'équation (5.2) et la détailler :

$$E_{\text{global}}(d) = \sum_{\mathbf{p}_g^{i,j} \in \mathcal{S}} \left((1 - \lambda) \underbrace{\sum_{\substack{\mathbf{p}_g^{i',j'} \in \text{ZA}(\mathbf{p}_g^{i,j}) \\ \mathbf{p}_d^{u',v'} \in \text{ZA}(\mathbf{p}_d^{u,v})}} E_{\text{dissimilarité}}(\mathbf{p}_g^{i',j'}, \mathbf{p}_d^{u',v'})}_{\text{coût local}} + \lambda \sum_{c=0}^{N_c-1} \underbrace{\sum_{\mathbf{p}_g^{i',j'} \in \text{ZAC}^c(\mathbf{p}_g^{i,j})} E_{\text{lissage}}^c(\mathbf{p}_g^{i,j}, \mathbf{p}_g^{i',j'})}_{\text{coût du voisinage}} \right) \quad (5.7)$$

avec $\mathbf{p}_d^{u,v} = \mathbf{p}_g^{i,j} + d(\mathbf{p}_g^{i,j})$.

Les choix à faire pour ce coût global de mise en correspondance sont :

- λ – Le plus souvent, les publications n'abordent pas cette pondération.
- $E_{\text{dissimilarité}}$ – Ce coût s'appuie souvent sur les niveaux de gris des pixels. Cet aspect sera détaillé au chapitre 6.
- E_{lissage}^c – Ce coût, dans la majorité des cas, est lié à la disparité associée à chaque pixel ou au fait que le pixel est considéré comme occulté ou non.
- ZA – Nous distinguons les méthodes qui utilisent une zone d'agrégation réduite à un seul pixel (le pixel étudié), ce sont les méthodes globales, des méthodes qui utilisent une zone d'agrégation plus importante, ce sont les méthodes locales et les méthodes mixtes.
- ZAC^c – Les zones d'agrégation des contraintes ne sont, en général, pas identiques aux zones d'agrégation du coût local. Dans de nombreux cas, la zone d'agrégation pour le coût local est limitée au pixel étudié alors que la zone d'agrégation des contraintes est un 4-voisinage.
- \mathcal{S} – Le support peut être le pixel considéré ou l'ensemble des N_g pixels considérés dans l'image ou les N_g^{col} pixels qui se trouvent sur la même ligne (dans le cas des images rectifiées, nous reviendrons sur cette définition au § 2.2.2).

Les méthodes locales effectuent N_g minimisations des coûts locaux et la zone d'agrégation est toujours strictement supérieure à un pixel (au pixel étudié), alors que les méthodes globales effectuent une minimisation du coût global qui est une fonction des coûts locaux et la zone d'agrégation est souvent réduite au pixel considéré.

5.4 Algorithmes de mise en correspondance

Il existe de très nombreuses approches de mise en correspondance. Une manière classique de les distinguer est de différencier les **approches locales** des **approches globales**. En effet, si on reprend le coût défini par l'équation (5.2), certaines méthodes réduisent ce coût à un simple coût local de mise en correspondance et sont appelées les méthodes locales. Les autres méthodes sont appelées les méthodes globales. Ainsi, les **approches locales cherchent à minimiser localement un coût de mise en correspondance alors que les approches globales minimisent ce coût globalement**. Plus précisément, les **méthodes locales** effectuent N_g minimisations des coûts locaux et la zone d'agrégation est toujours strictement supérieure à un pixel (au pixel étudié), alors que les **méthodes**

globales effectuent une minimisation du coût global qui est une fonction des coûts locaux et la zone d'agrégation est souvent réduite au pixel considéré.

5.4.1 Approche globale classique

Les étapes d'une approche globale sont donc les suivantes :

- (1) Estimer un coût de correspondance initiale liée à une première approximation des disparités ;
- (2) Mettre à jour C grâce à la méthode d'optimisation choisie.

Pour la première étape, l'estimation des correspondances initiales et du coût associé, deux solutions sont envisageables :

- Effectuer une mise en correspondance dense et l'étape d'optimisation consistera à mettre à jour des affectations ;
- Effectuer une mise en correspondance éparses mais fiable et l'étape d'optimisation consistera à propager les correspondances trouvées.

Dans le premier cas, si la solution initiale est très bruitée, l'optimisation sera coûteuse alors que dans le deuxième cas, on peut espérer que ce sera plus rapide. Cependant, une erreur sur les correspondances initiales aura plus d'impact sur le résultat final.

En ce qui concerne le coût de similarité, une approche classique consiste à utiliser l'information mutuelle ou *Mutual Information*, MI [Hirschmüller 08] puisqu'elle est robuste aux distorsions radiométriques.

Pour l'étape d'optimisation du coût, il pourrait être envisagé une **recherche exhaustive** mais ceci est désastreux en terme de temps de calculs, il est donc primordial d'utiliser une **méthode d'optimisation** pour minimiser ce coût. Dans la littérature, on peut trouver une utilisation abondante des algorithmes de programmation dynamique, de coupure de graphes, de propagation de croyance, des algorithmes génétiques ...

Une recherche exhaustive est envisageable, en local, et c'est le cas des **approches par corrélation**.

5.4.2 Algorithme d'une approche locale classique

La technique de mise en correspondance par corrélation est très ancienne, très populaire et relativement efficace. Dans les approches basées corrélation, on peut mettre en correspondance des points en utilisant une **stratégie de meilleur d'abord**, cf. figure 5.1 et 5.2. Les algorithmes à base de mesure de corrélation, sont souvent appelés les algorithmes « de région », *block matching* ou *area-based matching*.

Ces méthodes locales sont aussi appelées méthodes par corrélation. Une corrélation indique un degré de ressemblance entre deux ensembles de données. Par abus de langage, dans ce cours, quand nous parlons de méthode par corrélation, la mesure de corrélation utilisée évalue le degré de dissimilarité entre deux ensembles de pixels (score de corrélation). En ce qui concerne le coût de mise en correspondance, pour ces méthodes, nous avons :

- S – Il y a N_g supports, chacun étant réduit à un singleton.
- ZA – Il y a deux possibilités : utiliser une fenêtre carrée centrée sur le pixel considéré ou encore des fenêtres adaptatives [Veksler 02, Yoon 06, Zhang 09]. Les approches les plus sophistiquées s'appuient sur l'utilisation de poids attribués aux pixels de la zone considérée et permettent, entre autres, la gestion des déformations liées à la perspective [Hosni 13]. Nous pouvons également citer les approches prenant en compte un sous-ensemble de pixels non connexe [Muresan 15], en s'appuyant sur les mêmes principes que les *Local Binary Pattern*, LBP, cf. § 4.6.4.
- $E_{\text{dissimilarité}}$ – Une des variantes les plus utilisées est :

$$E_{\text{dissimilarité}}(\mathbf{p}_g^{i',j'}, \mathbf{p}_d^{u',v'}) = f_{\text{dissimilarité}}(I_g^{i',j'} - I_d^{u',v'}) = (I_g^{i',j'} - I_d^{u',v'})^2, \quad (5.8)$$

où $f_{\text{dissimilarité}}$ est une fonction de la différence des deux niveaux de gris.

Les méthodes locales n'ont pas de coût des contraintes et seul le coût local, équation (5.4), est utilisé. Ainsi la fonction de disparité d est déterminée de la façon suivante :

$$\forall \mathbf{p}_g^{i,j} \in \mathcal{P}_g \quad d(\mathbf{p}_g^{i,j}) = (u - i \ v - j)^T \text{ avec } \mathbf{p}_d^{u,v} = \underset{\mathbf{p}_d^{u',v'} \in Z_d(\mathbf{p}_g^{i,j})}{\operatorname{argmin}} E_{\text{local}}(\mathbf{p}_g^{i,j}, \mathbf{p}_d^{u',v'}), \quad (5.9)$$

où \mathcal{P}_g est l'ensemble des pixels considérés dans l'image, $Z_d(\mathbf{p}_g^{i,j})$ correspond à la zone de recherche associée au pixel $\mathbf{p}_g^{i,j}$. La méthode d'optimisation employée est la méthode de recherche exhaustive, *winner take all*, notée WTA, c'est-à-dire que le pixel qui obtient le meilleur coût est sélectionné.

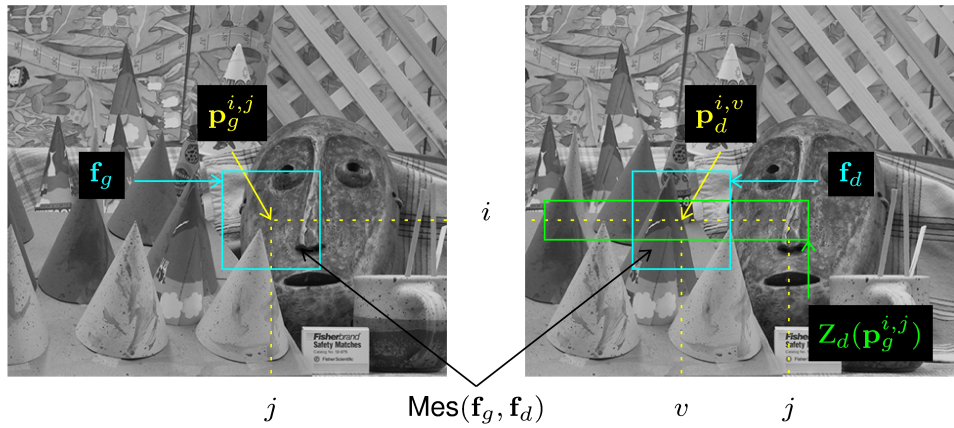


FIGURE 5.1 – Mise en correspondance par corrélation – Pour chaque point de l'image de gauche, montré par la flèche jaune, nous cherchons un correspondant. Pour cela nous déterminons une zone de recherche dans la seconde image (le rectangle vert). Pour chaque point de cette zone (tous les pointillés jaune dans le rectangle vert), nous calculons un score de corrélation, noté Mes. Ce score de corrélation prend en compte les voisinages des points étudiés (les 2 rectangles bleus). Cette figure illustre l'algorithme fourni dans la figure 5.2.

Données
Images à traiter : I_1 et I_2
Point d'intérêt : $\mathbf{p}_1 = (x_1, y_1)$
Zone de recherche du point d'intérêt correspondant de \mathbf{p}_1 : $\mathcal{ROI}(\mathbf{p}_1)$
Mesure de similarité : $m(\mathbf{p}_1, \mathbf{p}_2)$
Algorithme de mise en correspondance
1. <u>Pour</u> chaque $\mathbf{p}_1 \in I_1$ <u>faire</u> Estimer $\mathbf{p}_2 \in I_2$ tel que $\mathbf{p}_2 = \underset{\mathbf{p} \in \mathcal{ROI}(\mathbf{p}_1)}{\operatorname{argmin}} m(\mathbf{p}_1, \mathbf{p})$
2. Ajout de contrainte (seuil, bidirectionnelle)

FIGURE 5.2 – Mise en correspondance locale de primitives – Cet algorithme peut être appliqué entre **2** ou n images. Nous pouvons effectuer une mise en correspondance entre tous les points d'intérêt et tous les points de la deuxième image ou seulement points d'intérêt à points d'intérêt. Expérimentalement, la **première solution est la plus efficace**. Mais, en **termes de temps de calculs**, il est souvent choisi de faire une mise en correspondance points d'intérêt à points d'intérêt (**deuxième solution**). Certains détails de cet algorithme sont donnés dans la figure 5.1.

La mesure m correspond aux mesures que nous allons présenter dans la section 6.

Avantages et inconvénients Les méthodes à base de corrélation possèdent les avantages suivants :

- Leur mise en œuvre est simple et rapide.
- Elles ne sont pas très coûteuses en temps de calcul (elles sont moins coûteuses que les méthodes globales).
- Dans de nombreux travaux, elles ont déjà prouvé leur efficacité [Scharstein 02b].

Les difficultés qui peuvent être liées à ces méthodes sont les suivantes :

- Elles ne prennent pas toujours en compte certaines situations qui rendent difficile la mise en correspondance : les bruits, les raccourcissements, les zones non texturées, les textures répétitives, les changements de luminosité et les occultations.
- Elles ne fonctionnent pas dans le cas de déformation des objets ou de changement d'échelle (nous parlons ici des méthodes classiques qui n'utilisent pas des zones d'agrégation qui s'adaptent à la forme des objets).
- Elles n'autorisent pas des points de vue trop éloignés.
- Comme elles sont locales, les résultats peuvent être erronés car certaines caractéristiques globales de l'image ne sont pas prises en compte. Par exemple, certaines méthodes globales segmentent l'image en régions qui correspondent à des projections d'un même objet de la scène. Cette segmentation permet de réduire l'espace des disparités envisageables, et donc de réduire les erreurs.

Face à ces difficultés, de nombreuses techniques ont été introduites et notamment des approches dites multiples qui combinent plusieurs techniques d'appariement ou une mise en correspondance conjointe. Dans ce contexte, nous appelons mise en correspondance conjointe un algorithme d'appariement qui va combiner les intérêts de multiples approches d'appariement ou de multiples informations extraites. Dans ce cours, nous avons choisi de présenter le principe des approches combinées à une segmentation en régions, voire une sursegmentation.

5.4.3 Appariement en utilisant une segmentation en régions

Le principe des méthodes utilisant des régions repose sur l'hypothèse qu'une région de couleur homogène dans l'image suffisamment petite peut être la projection d'une même surface de la scène et que cette surface peut être approchée par un modèle dans l'espace des disparités, par exemple un plan ou une surface *B-Spline*, [Hong 04, Klaus 06, Yang 09, Bleyer 10]. L'utilisation de ces régions peut aider la mise en correspondance de régions homogènes et permet également d'éviter des artefacts au niveau des discontinuités de profondeur. Le modèle plan est relativement simple et donne de bonnes approximations, en particulier avec les plus petites régions, mais peut se révéler imprécis lorsque les régions ne sont pas les projections de facettes planes. Dans ce dernier cas, le modèle *B-Spline* peut donner de meilleures approximations. Cependant, il est plus complexe à configurer (l'ordre et le vecteur nodal doivent être choisis avec soin) et d'importantes oscillations peuvent apparaître dans le résultat [Lin 03, Bleyer 10].

Ces méthodes comportent plusieurs étapes résumées par la figure 5.3 :

- **Segmentation des images en régions** – Une partition de l'image est calculée en utilisant un algorithme de segmentation en régions de couleur homogène [Gonzalez 04]. La méthode *mean-shift*¹ est la plus utilisée dans ce contexte [Klaus 06, Yang 09, Bleyer 10]. Il est important de noter que pour cette étape, une sur-segmentation est presque toujours calculée. Elle permet d'avoir des petites régions qui sont plus facilement approchées par un modèle plan.
- **Appariement initial classique** – Pour initialiser les disparités, une méthode de mise en correspondance locale est généralement employée comme celles décrites dans le § 5.4.2.
- **Estimation des paramètres du modèle choisi** – Le modèle plan est relativement simple et donne de bonnes approximations des disparités, en particulier pour les petites régions. Il s'agit du modèle le plus couramment utilisé mais le modèle *B-Spline* a également été étudié

1. Méthode étudiée en IMR, en 2A.

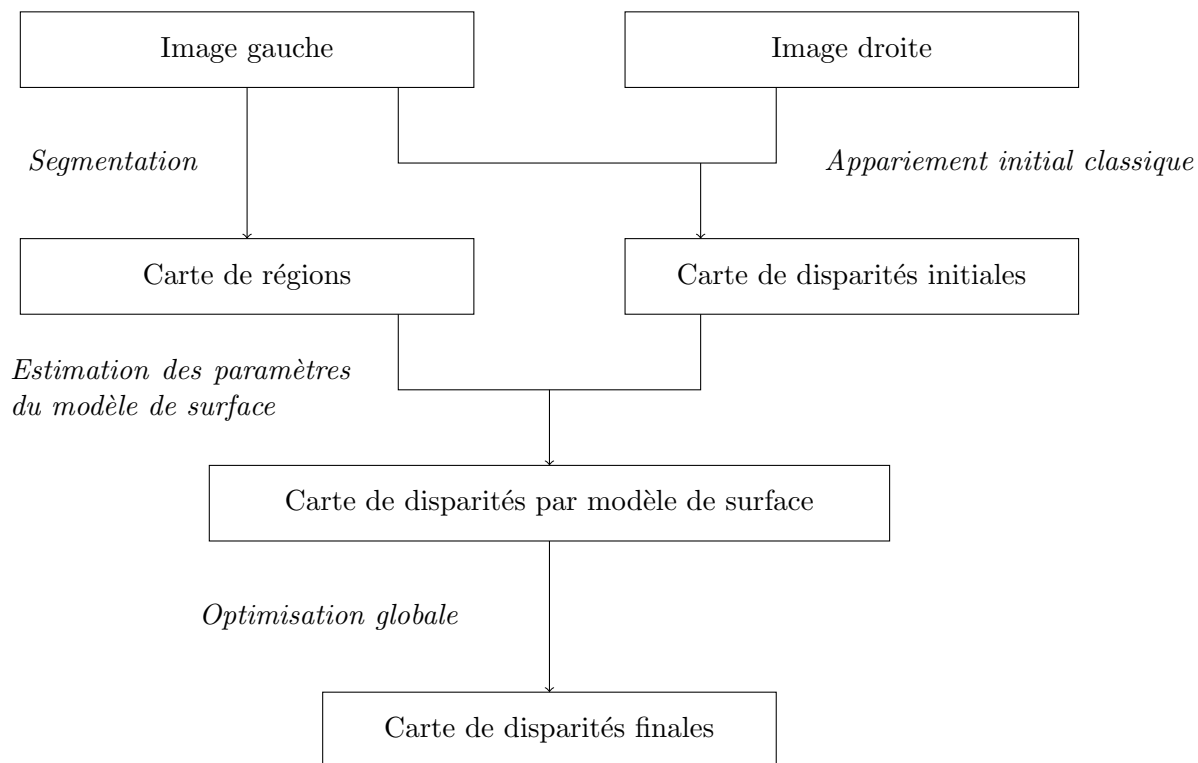


FIGURE 5.3 – Étapes des méthodes de mise en correspondance fondées sur les régions.

dans [Lin 03, Bleyer 10]. Les paramètres du modèle choisi sont habituellement estimés à l'aide d'une méthode d'estimation robuste à partir des disparités initiales : moindres carrés pondérés itératifs [Hong 04], RANSAC, *random sample consensus*, [Yang 09] ou décomposition des paramètres [Wang 08, Klaus 06].

- **Optimisation globale** – Les résultats sont raffinés en appliquant une optimisation globale. Nous pouvons distinguer les méthodes suivantes :
 - *Affectation d'un plan* – Les différents plans de disparité sont obtenus à partir des modèles estimés et chaque pixel est affecté à l'un de ces plans de telle sorte qu'un coût global sur l'image entière soit optimisé. Cela permet de corriger certaines erreurs notamment dues à une mauvaise segmentation. Cette solution est utilisée dans [Hong 04, Bleyer 10] où la technique d'optimisation employée est la coupure de graphe, et également dans [Klaus 06] où la technique d'optimisation employée est la propagation de croyance.
 - *Raffinement des paramètres* – Les paramètres des modèles préalablement estimés peuvent être affinés en minimisant un coût global. Cette solution est notamment utilisée dans [Wang 08].

Ce chapitre nous a permis de détailler les approches les plus utilisées dans la littérature. À présent, nous allons donner des détails sur le critères de similarité et les contraintes que nous pouvons appliquer, avant de détailler des algorithmes proposés dans le cadre de la mise en correspondance de points d'intérêt (SIFT) ou dans le contexte du suivi sans aborder directement la notion de points d'intérêt, KLT.

Chapitre 6

Mesures de similarité utilisées dans les approches locales classiques

6.1 Définitions

Une définition d'une mesure de corrélation est la suivante :

Il s'agit d'une mesure qui évalue à quel point deux ensembles de données se ressemblent.

Par abus de langage, nous employons le terme « mesure de corrélation » aussi bien pour une mesure qui évalue le degré de similarité que pour une mesure qui évalue le degré de dissimilarité.

Le but de la mise en correspondance est de trouver pour chaque pixel $\mathbf{p}_1^{i,j}$ de l'image 1, le pixel $\mathbf{p}_2^{i,v}$ qui lui correspond dans l'image 2. La mise en correspondance par corrélation repose donc sur l'hypothèse suivante :

Les niveaux de gris des pixels correspondants et des pixels de leurs voisinages se ressemblent.

Les mesures de corrélation utilisent donc l'information donnée par les pixels, mais aussi celle qui est fournie par leurs voisinages respectifs. Ainsi, le score de corrélation, qui est la valeur de la mesure de corrélation, est évalué à partir de deux ensembles : le pixel de l'image 1 et son voisinage, représentés par le vecteur \mathbf{f}_1 , et le pixel de l'image 2 et son voisinage, représentés par le vecteur \mathbf{f}_2 .

6.2 Difficultés

La principale limite de la mise en correspondance par corrélation provient de l'hypothèse de départ. En utilisant les pixels du voisinage du pixel considéré pour déterminer le score de corrélation, on suppose que ces derniers possèdent tous la même disparité. Dans la plupart des cas, cette hypothèse est quasiment respectée, mais elle ne l'est plus dans les cas suivants :

- présence de bruits ;
- changements de luminosité ;
- déformations (dues à la projection perspective) des objets de la scène ;
- occultations.

Dans le cas d'images bruitées, des méthodes de préfiltrage peuvent être facilement mises en œuvre. Pour résoudre le problème des changements de luminosité, des mesures centrées et normalisées sont proposées.

Pour les déformations dues à la projection perspective, on peut utiliser une approche de corrélation fine. Le principe de cette méthode est d'adapter localement la fenêtre de corrélation à la courbure et à l'orientation locale de la surface de l'objet considéré. Ainsi, en plus d'estimer la disparité associée à chaque pixel, il faut estimer l'ensemble des paramètres de la transformation locale associée à ce pixel.

Dans le cas des occultations, l'hypothèse de la mise en correspondance par corrélation n'est plus vérifiée à la frontière des objets occultants. En effet, dans le voisinage d'un pixel situé à la frontière d'un changement de profondeur, certains pixels appartiennent au premier niveau de profondeur et d'autres au deuxième niveau. Cette situation va perturber la mise en correspondance par corrélation, car d'une image à l'autre, les pixels qui se correspondent n'auront pas les mêmes niveaux de gris dans leur voisinage et leur score de corrélation ne sera pas maximal.

Plusieurs solutions ont été envisagées pour prendre en compte ce problème. Les deux solutions principales sont :

- Les **fenêtres adaptatives** – La forme de la fenêtre n'est pas fixe. Elle s'adapte en fonction de la zone de l'image qui est parcourue. La méthode de corrélation fine peut être considérée comme une méthode de fenêtres adaptatives.
- Les **mesures robustes aux occultations** – Ces mesures intègrent directement le problème des occultations dans le calcul du score de corrélation.

6.3 Familles de mesures

Nous pouvons distinguer cinq familles de mesures :

- la famille CROISÉE qui regroupe les mesures utilisant la corrélation croisée ;
- la famille CLASSIQUE qui regroupe les mesures utilisant des outils des statistiques classiques de la distribution des différences des niveaux de gris ;
- la famille DÉRIVÉE qui regroupe les mesures utilisant les dérivées des niveaux de gris ;
- la famille NON PARAMÉTRIQUE qui regroupe les mesures utilisant des outils des statistiques non paramétriques ;
- la famille ROBUSTE qui regroupe les mesures utilisant des outils des statistiques robustes.

Nous allons fournir des exemples pour chacune de ces familles.

Notations

Dans la suite, nous parlerons d'invariance au gain et au biais. Cela signifie que nous indiquons si la mesure donne le même résultat s'il y a un biais (ajout d'une constante) ou un gain (facteur multiplicatif). Plus précisément :

- Une mesure M invariante au biais signifie que $M(\mathbf{f}_1 + a, \mathbf{f}_2 + b) = M(\mathbf{f}_1, \mathbf{f}_2)$.
- Une mesure M invariante au gain signifie que $M(a\mathbf{f}_1, b\mathbf{f}_2) = M(\mathbf{f}_1, \mathbf{f}_2)$.

$N_f = (2N_v + 1)(2N_h + 1)$	La taille des fenêtres de corrélation est $(2N_v + 1) \times (2N_h + 1)$ et N_f est donc le nombre de pixels dans la fenêtre de corrélation où N_v est la demi-taille de la fenêtre en verticale (colonne) et N_h en horizontale (ligne).
$\mathbf{f}_l = (\cdots I(i + p, j + q)_l \cdots)^T = (\cdots f_l^k \cdots)^T$ où f_l^k est l'élément k du vecteur \mathbf{f}_l , $p \in [-N_v; N_v]$, $q \in [-N_h; N_h]$ et $k \in [0; N_f - 1]$.	Les niveaux de gris des pixels des fenêtres de corrélation (matrices) sont stockés dans les vecteurs \mathbf{f}_l avec $l \in \{1, 2\}$.
$\nabla I_l^{i,j}$	Il s'agit du vecteur gradient au pixel $(i, j)^T$ de l'image I_l . La norme et l'orientation du gradient sont notées respectivement $\ \nabla I_l^{i,j}\ $ et $\theta_l^{i,j}$.
$\ \mathbf{f}_l\ _P = \left(\sum_{k=0}^{N_f-1} f_l^k ^P \right)^{1/P}$	On notera ainsi les P-normes, ou distances L_P , $P \in \mathbb{N}^*$. Pour la norme euclidienne, nous notons : $\ \mathbf{f}_l\ = \ \mathbf{f}_l\ _2$.
$\mathbf{f}_1 \cdot \mathbf{f}_2 = \sum_{k=0}^{N_f-1} f_1^k f_2^k$	Le produit scalaire.
$\text{moy}(\mathbf{f}_l) = \frac{1}{N_f} \sum_{k=0}^{N_f-1} f_l^k$	Les moyennes. Nous utilisons aussi le vecteur des moyennes défini par : $\bar{\mathbf{f}}_l = (\underbrace{\text{moy}(\mathbf{f}_l) \cdots \text{moy}(\mathbf{f}_l)}_{N_f \text{ colonnes}})^T.$
$\text{sgn}(x) = \begin{cases} -1 & \text{si } x < 0 \\ 0 & \text{si } x = 0 \\ 1 & \text{sinon.} \end{cases}$	La fonction signe.
$D_{\text{Ham}}(\mathbf{f}_1, \mathbf{f}_2) = \sum_{k=0}^{N_f-1} \text{sgn} f_1^k - f_2^k .$	La distance de Hamming.
$(f_l)_{0:N_f-1} \leq \cdots \leq (f_l)_{k:N_f-1} \leq \cdots \leq (f_l)_{N_f-1:N_f-1}.$	Les valeurs ordonnées du vecteur \mathbf{f}_l .

6.4 Corrélation croisée

Cette mesure de similarité évalue grâce à un produit scalaire la différence entre les deux vecteurs de données de la manière suivante :

$$\text{ZNCC}(\mathbf{f}_1, \mathbf{f}_2) = \frac{(\mathbf{f}_1 - \bar{\mathbf{f}}_1) \cdot (\mathbf{f}_2 - \bar{\mathbf{f}}_2)}{\|\mathbf{f}_1 - \bar{\mathbf{f}}_1\| \|\mathbf{f}_2 - \bar{\mathbf{f}}_2\|} \quad (6.1)$$

où $\mathbf{f}_w, w = 1, 2$ correspond aux vecteurs des niveaux de gris dans les voisinages respectifs des points appariés. Il s'agit de la **corrélation centrée normalisée**, *Zero-mean Normalised Cross Correlation*. La normalisation est nécessaire car, sans normalisation, plus les niveaux de gris sont élevés, plus les scores sont élevés. De plus, cela permet une invariance de type gain. Cette mesure correspond au coefficient de corrélation linéaire classique en statistiques. Grâce au centrage (le fait de retrancher la moyenne des niveaux de gris), elle a l'avantage de présenter une invariance de type biais.

6.5 Corrélation classique

Les deux mesures les plus utilisées sont les suivantes :

- **Somme des valeurs absolues des différences** – Il s'agit de la norme L_1 , notée SAD (*Sum of Absolute Differences*) :

$$\text{SAD}(\mathbf{f}_1, \mathbf{f}_2) = D_1(\mathbf{f}_1, \mathbf{f}_2) = \|\mathbf{f}_1 - \mathbf{f}_2\|_1. \quad (6.2)$$

Cette mesure est aussi une des plus populaires et elle est très utilisée dans la littérature.

- **Somme des carrés des différences** – Il s'agit de la norme L_2 , notée SSD (*Sum of Squared Differences*) :

$$\text{SSD}(\mathbf{f}_1, \mathbf{f}_2) = D_2(\mathbf{f}_1, \mathbf{f}_2) = \|\mathbf{f}_1 - \mathbf{f}_2\|^2. \quad (6.3)$$

Cette mesure est aussi très utilisée.

De la même manière que la corrélation croisée, ou plus précisément que la mesure ZNCC présentée au paragraphe 6.4, ces mesures peuvent être centrées et normalisées.

6.6 Corrélation des dérivées : Corrélation des champs de gradients

La plupart des mesures de cette famille font intervenir la direction du gradient. Or cette information, seule, peut entraîner des erreurs, surtout dans le cas de gradients de faible norme.

Une mesure, appelée corrélation des champs de gradients, notée GC (*Gradient field Correlation*) utilise la similarité des vecteurs gradients et est donnée par :

$$\text{GC}(\mathbf{f}_1, \mathbf{f}_2) = \frac{\sum_{p=-N_v}^{N_v} \sum_{q=-N_h}^{N_h} \|\nabla I_1^{i+p, j+q} - \nabla I_2^{i+p, v+q}\|}{\sum_{p=-N_v}^{N_v} \sum_{q=-N_h}^{N_h} (\|\nabla I_1^{i+p, j+q}\| + \|\nabla I_2^{i+p, v+q}\|)}. \quad (6.4)$$

Il s'agit d'une mesure de dissimilarité dont les valeurs appartiennent à l'intervalle $[0; \infty[$. Cette mesure est invariante au biais. Son implémentation nécessite une étape supplémentaire de pré-traitement pour calculer les gradients de l'image, cf. cours 2A.

6.7 Mesures non paramétriques

Les mesures sont fondées sur des transformations qui ne font aucune hypothèse sur les distributions sous-jacentes des niveaux de gris de la fenêtre de corrélation. Deux grands types de mesures ont été distingués : les mesures s'appuyant sur les différences de niveaux de gris entre le point étudié et ses voisins, les mesures exploitant le rang.

6.7.1 ISC, *Increment Sign Correlation*

Parmi les mesures exploitant les différences de niveaux de gris, nous pouvons citer la mesure ISC, *Increment Sign Correlation*. Elle s'appuie sur les vecteurs \mathbf{b}_l suivants :

$$\mathbf{b}_l = \left(\dots b_l^k \dots \right)^T \text{ pour } k = 0 \dots N_f - 2 \text{ avec } b_l^k = \begin{cases} 1 & \text{si } f_l^{k+1} \geq f_l^k \\ 0 & \text{sinon.} \end{cases} \quad (6.5)$$

Si le niveau de gris augmente entre f_l^k et f_l^{k+1} , alors b_l^k vaut 1 ou 0 dans le cas contraire. La mesure ISC compare les vecteurs \mathbf{b}_g et \mathbf{b}_d . Elle détermine si les niveaux de gris varient dans le même sens. Elle est donnée par :

$$\text{ISC}(\mathbf{f}_g, \mathbf{f}_d) = \frac{1}{N_f - 1} (\mathbf{b}_g \cdot \mathbf{b}_d + (\mathbf{1} - \mathbf{b}_g) \cdot (\mathbf{1} - \mathbf{b}_d)). \quad (6.6)$$

C'est une mesure de similarité et ses valeurs appartiennent à l'intervalle $[0; 1]$.

6.7.2 CENSUS, mesure de recensement

La mesure la plus connue est la **mesure de recensement**. Elle s'appuie sur la transformation de recensement qui permet d'obtenir une chaîne de bits représentative des pixels contenus dans la fenêtre de corrélation. Cette chaîne rend compte des pixels dont l'intensité est inférieure à celle du pixel central. La transformation, notée R_τ , est définie par :

$$R_\tau(\mathbf{p}) = \bigotimes_{k \in [0; N_f - 1]} \xi(I^{N_f/2}, I^k), \quad (6.7)$$

où \bigotimes correspond ici à la concaténation. Le terme $\xi(I^{N_f/2}, I^k)$ vaut 1 si $I^{N_f/2} < I^k$ ou 0 dans le cas contraire.

Ainsi, la mesure de corrélation associée, notée CENSUS, utilise la distance de Hamming :

$$\text{CENSUS}(\mathbf{f}_1, \mathbf{f}_2) = \sum_k^{N_f - 1} D_{Ham}(R_\tau(\mathbf{p}_1^k), R_\tau(\mathbf{p}_2^k)). \quad (6.8)$$

où la distance de Hamming est donnée par :

$$D_{Ham}(\mathbf{a}_1, \mathbf{a}_2) = \sum_{k=0}^{N_f - 1} \text{sgn}|a_1^k - a_2^k|. \quad (6.9)$$

La distance de Hamming donne une valeur comprise dans l'intervalle $[0; N_f]$. Ainsi, la mesure CENSUS est une mesure de dissimilarité et ses valeurs appartiennent à l'intervalle $[0; N_f^2]$.

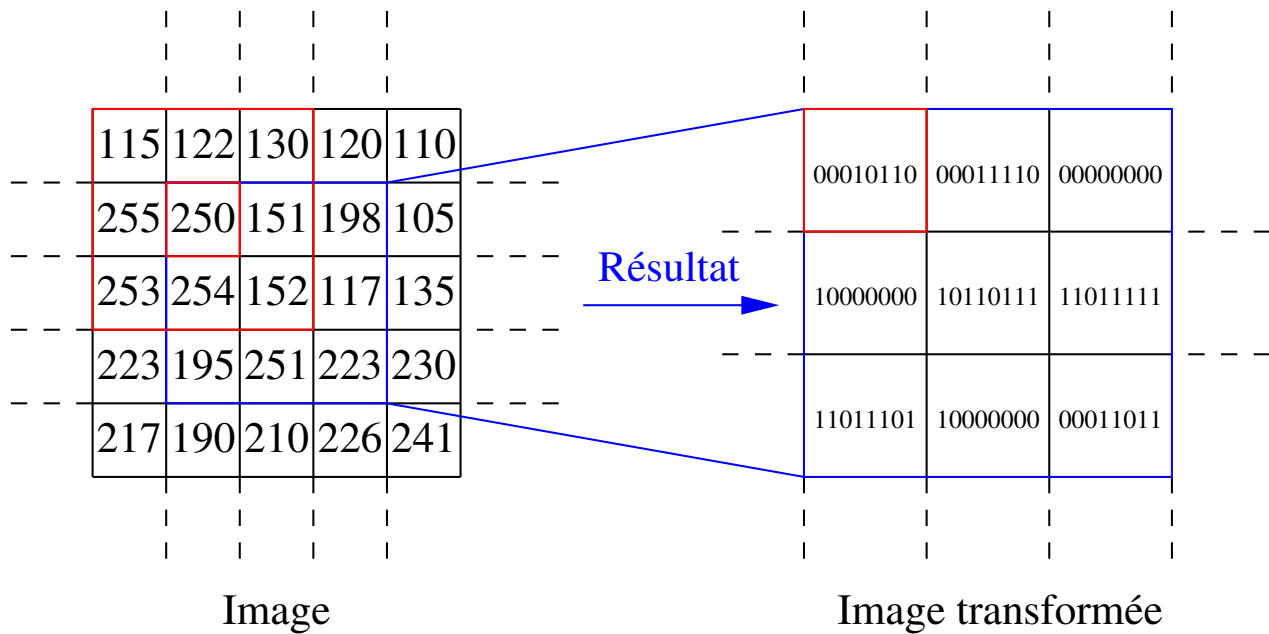


FIGURE 6.1 – Illustrations des calculs réalisés pour évaluer la transformation utilisée par la mesure CENSUS. Ici on suppose que le voisinage pris en compte est de taille 3×3 et que le codage est calculé en effectuant un balayage de gauche à droite et de haut en bas.

6.8 Mesures robustes aux occultations

Le problème des occultations se traduit par le phénomène suivant :

Dans le voisinage d'un pixel situé à la frontière d'un changement de profondeur, certains pixels appartiennent au premier niveau de profondeur et d'autres au deuxième niveau de profondeur. Cette situation peut perturber la mise en correspondance et provoquer des résultats erronés.

Pour prendre en compte ce problème, les mesures robustes s'appuient sur le principe suivant :

Les pixels qui ont une profondeur différente du pixel étudié sont considérés comme des données aberrantes (cf. figure 6.2).

Ainsi, ces mesures utilisent des outils de statistiques robustes qui sont beaucoup moins sensibles aux données aberrantes que les outils des statistiques classiques.

Une des plus simples est la **Smooth Median Powered Deviation**. Cette mesure s'appuie sur l'estimateur SMAD (*Smooth Median Absolute Deviation*), qui correspond à une estimation robuste de la variance définie par :

$$\text{SMAD}(\mathbf{f}_1, \mathbf{f}_2) = \sum_{k=0}^{h-1} \left(|\mathbf{f}_1 - \mathbf{f}_2 - \text{med}(\mathbf{f}_1 - \mathbf{f}_2)|^2 \right)_{k:N_f-1}. \quad (6.10)$$

Le terme h indique le nombre de différence prises en compte. Plus précisément, nous évaluons SMAD avec les h premières puissances des écarts à la médiane. Ainsi, en généralisant le type de distance utilisée, nous obtenons cette mesure, notée SMPD_P (*Smooth Median Powered Deviation*), et donnée par :

$$\text{SMPD}_P(\mathbf{f}_1, \mathbf{f}_2) = \sum_{k=0}^{h-1} \left(|\mathbf{f}_1 - \mathbf{f}_2 - \text{med}(\mathbf{f}_1 - \mathbf{f}_2)|^P \right)_{k:N_f-1}. \quad (6.11)$$

Les valeurs de cette mesure de dissimilarité appartiennent à l'intervalle $[0; \mathbf{I}_{\max}^P h]$.

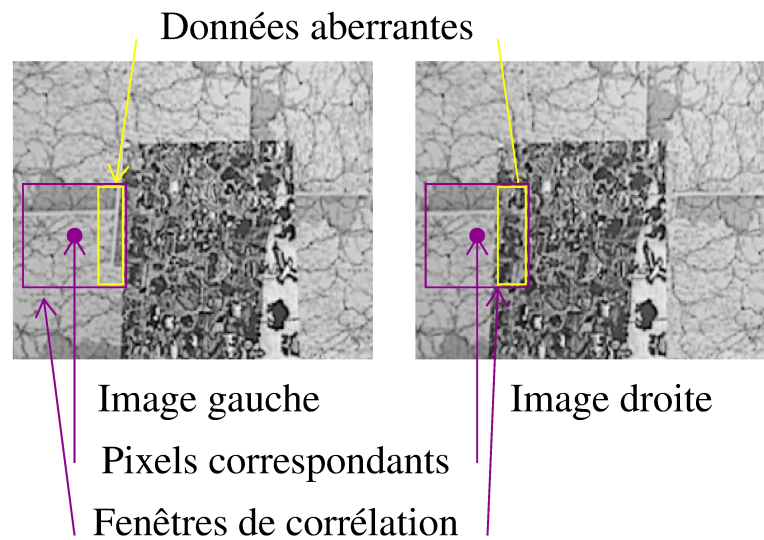


FIGURE 6.2 – Principe des mesures robustes – Les deux disques foncés mettent en évidence deux pixels qui se correspondent parfaitement. Les rectangles clairs entourent les parties droites des fenêtres de corrélation centrées sur les pixels qui se correspondent. Ces rectangles délimitent deux zones très différentes si on considère les niveaux de gris. En effet, dans l'image de gauche cette zone fait partie de l'arrière plan, alors que, dans l'image de droite, elle fait partie du premier plan. La comparaison de ces deux zones n'est donc pas pertinente. Le principe des mesures robustes est de considérer les niveaux de gris des pixels appartenant à cette partie de la fenêtre de corrélation comme des données aberrantes.

6.9 Illustration des résultats sur images de synthèse et images réelles

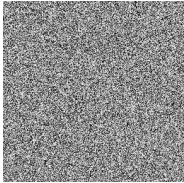
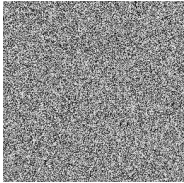
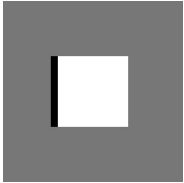
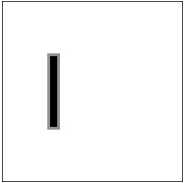
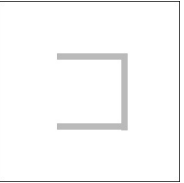
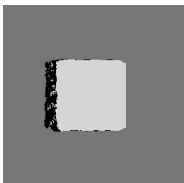

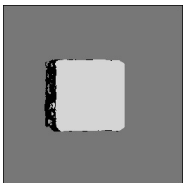
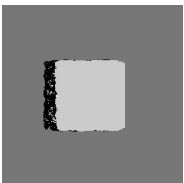
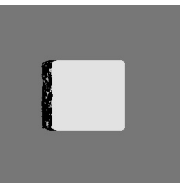
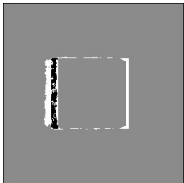
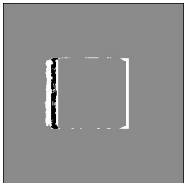
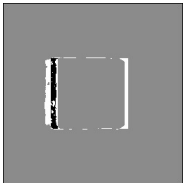
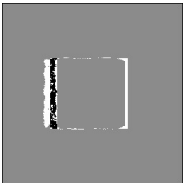
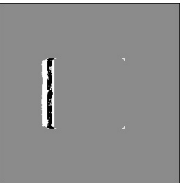


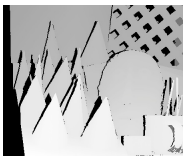
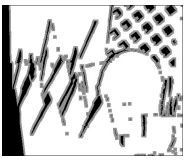

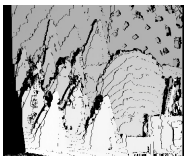
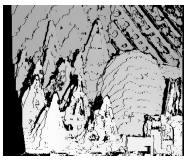
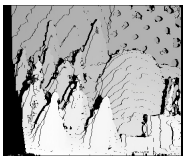
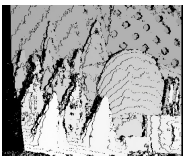
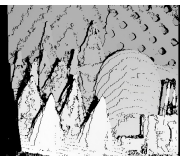
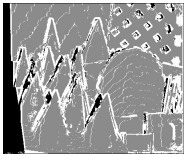
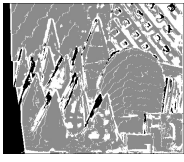
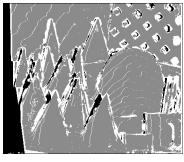
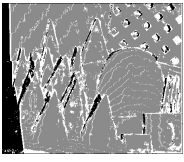
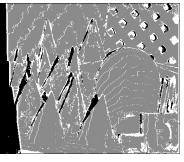
	Gauche	Droite	Disparités	Occultations	Discontinuités
					
	MOR	SAD	GC	ISC	SMPD ₂
Disparités					
Corrects					
	Gauche	Droite	Disparités	Occultations	Discontinuités
					
	MOR	SAD	GC	ISC	SMPD ₂
Disparités					
Corrects					

TABLE 6.1 – Exemples de résultats d'appariement – Si nous observons les frontières des objets, les résultats obtenus sont de meilleure qualité avec les mesures robustes ou non paramétriques puisque ces mesures prennent en compte les difficultés au niveau des discontinuités de profondeur ou des occultations. En revanche, comme ces mesures sont plus permissives, elles obtiennent plus de points mal appariés (les points noirs occultés) dans les zones sans difficulté.

Chapitre 7

Contraintes pour la mise en correspondance

7.1 Introduction

Les contraintes sur la mise en correspondance peuvent intervenir au niveau du deuxième terme de l'équation (5.2) ou au moment de l'affinement des résultats, étape (5). Les buts de l'utilisation de ces contraintes sont :

- limiter le nombre de correspondants potentiels pour chaque pixel ;
- choisir entre plusieurs correspondants potentiels ;
- vérifier et, le cas échéant, supprimer des correspondances.

Il est important de noter que nous pouvons distinguer les **contraintes unaires** qui ne prennent en compte qu'une correspondance, des **contraintes binaires**, qui utilisent les informations relatives à deux correspondances. Ainsi, une contrainte est une propriété liée à une correspondance qui découle d'hypothèses faites sur : la géométrie de la scène, la réflectance de la scène ou la géométrie de la caméra.

Nous rappelons des éléments sur chacun de ces points, sachant que les deux derniers ont été déjà largement abordés dans les autres chapitres et nous développerons ensuite toutes les contraintes liées à la géométrie de la scène.

7.2 Contraintes liées à la géométrie de la scène

Il s'agit des contraintes d'unicité, d'ordre et de symétrie aussi appelées **contraintes de compatibilité**. Elles ne sont pas appliquées seulement à des pixels, mais aussi à d'autres types de primitives, comme les segments. Nous allons décrire, à partir du paragraphe 7.6, ce que nous appelons les contraintes de compatibilité. Il s'agit des plus utilisées, mais cette liste n'est pas exhaustive.

7.3 Contraintes liées à la géométrie du capteur

La principale contrainte utilisée est la contrainte épipolaire. Par abus de langage, nous appelons cette contrainte une contrainte géométrique (sans préciser qu'il s'agit de la géométrie du capteur). La contrainte épipolaire est unaire, et elle peut être utilisée pour faciliter la mise en correspondance en réduisant la zone de recherche. Nous l'avons déjà présentée dans la section 2.2.1. Cette contrainte devient particulièrement intéressante lorsqu'une rectification épipolaire est réalisée, cf. section 2.2.2. En effet, la configuration obtenue (les droites épipolaires conjuguées sont alignées avec les lignes de l'image) est très intéressante car elle simplifie la méthode de recherche des correspondants. Cette contrainte géométrique est utilisée pendant la mise en correspondance alors que les contraintes de

compatibilité peuvent être utilisées pendant ou après l'étape de mise en correspondance. Lorsque les images sont rectifiées, la disparité correspond seulement à la différence de colonnes. Ainsi, si nous étudions le déplacement réalisé entre l'image gauche et l'image droite (ce qui donne comme mouvement apparent dans l'image un déplacement vers la gauche des objets), cette configuration rectifiée implique que la zone de recherche dans l'image droite soit constituée des pixels sur la ligne i et situés sur les colonnes dont l'indice est inférieur ou égal à j . De plus, cette zone de recherche est souvent réduite. En effet, il est possible de fixer des limites respectivement inférieure et supérieure aux disparités qui correspondent aux objets de la scène, respectivement les plus éloignés et les plus proches du capteur.

7.4 Contraintes liées à la réflectance de la surface des objets et les caractéristiques de la source lumineuse

Cela concerne la contrainte de dissimilarité. Nous ne présentons pas la contrainte de dissimilarité dans ce paragraphe car nous considérons qu'elle constitue l'élément essentiel du coût de mise en correspondance que nous avons déjà abordée au paragraphe 6. Toutefois, nous pouvons citer une dernière contrainte très utilisée dans la littérature : le seuillage des scores de corrélation. En effet, nous pouvons considérer qu'au dessous d'une certaine valeur, la correspondance obtenue n'est pas fiable. Nous pouvons donc choisir de l'invalider. Plus précisément, cette contrainte se définit ainsi :

$$\text{Si } \mathbf{p}_1 + d(\mathbf{p}_1) = \mathbf{p}_2 \text{ alors } \text{Mes}(\mathbf{p}_1, \mathbf{p}_2) > T. \quad (7.1)$$

Le seuil T est relatif au type de critère utilisé et à la qualité des résultats de mise en correspondance attendue. En effet, plus le seuil sera restrictif, moins il y aura de correspondances erronées. En revanche, il y a un risque d'éliminer plus de correspondances valides.

Ainsi, dans la suite, nous ne présenterons que les contraintes liées à la géométrie de la scène. Avant de commencer les explications sur les différentes contraintes qui peuvent être utilisées, nous expliquons comment le résultat d'une contrainte peut être exploité.

7.5 Prise en compte d'une contrainte

Nous pouvons distinguer deux manières différentes de prendre en compte ces contraintes :

- (1) Toutes les correspondances ne respectant pas la contrainte sont éliminées : nous obtiendrons donc une disparité éparse.
- (2) Une partie des correspondances est éliminée/l'autre est conservée, voire corrigée.

Dans le deuxième cas, il est nécessaire de définir une stratégie de traitement des correspondances qui posent problème. Là encore, il y a deux possibilités :

- (1) Il est possible de choisir d'éliminer le plus petit sous-ensemble de manière à respecter la contrainte. Par exemple, pour la contrainte figurale, nous allons conserver la correspondance qui se trouve sur le contour principal.
- (2) Une autre solution consiste à vérifier ou compléter cette contrainte par une seconde contrainte pour faire le choix. Par exemple, on peut appliquer une contrainte de seuil sur le critère de similarité pour déterminer quelle correspondance conserver dans le cas où la contrainte d'ordre n'est pas respectée.

À présent, nous allons détailler les contraintes les plus utilisées dans la littérature, en commençant par présenter les notations utilisées.

7.6 Notations utilisées pour les illustrations

À partir de ce paragraphe, nous supposons toujours que nous travaillons avec des images rectifiées. Tous les exemples que nous donnons sont dans le cas où les droites épipolaires conjuguées correspondent à la même ligne dans les deux images étudiées. Nous avons choisi des représentations communes pour toutes les figures illustrant les contraintes de compatibilité ; elles sont résumées dans le tableau 7.1.

REPRÉSENTATION	SIGNIFICATION
■	représente un pixel.
○	indique qu'il y a une correspondance entre deux pixels.
×	indique que cette correspondance est interdite ou qu'un pixel est occulté.
→	représente une correspondance entre deux pixels qui respecte une contrainte.
- - - - - →	représente une correspondance entre deux pixels qui ne respecte pas une contrainte.

TABLE 7.1 – Symboles utilisés dans les figures.

De plus, à partir du paragraphe suivant, nous nous plaçons dans le cas des méthodes de mise en correspondance de pixel à pixel.

À partir de ce paragraphe, nous supposons toujours que nous travaillons avec des **images rectifiées**, c'est-à-dire, tous les exemples que nous donnons sont dans le cas où **les correspondants se trouvent sur la même ligne dans les deux images étudiées**, cf. section 2.2.2. Ceci est réalisé pour simplifier l'exposé et toutes les contraintes présentées se généralisent simplement aux cas de déplacements dans les 2 directions.

Nous noterons $\mathbf{p}_2 = \mathbf{p}_1 + d(\mathbf{p}_1)$ le correspondant \mathbf{p}_2 dans I_2 du point \mathbf{p}_1 de I_1 .

7.7 Contrainte d'unicité

La contrainte d'unicité (cf. figure 7.1.(a)) est définie par :

$$\text{Si } \mathbf{p}_1 + d(\mathbf{p}_1) = \mathbf{p}_2 \text{ alors } \forall \mathbf{p}'_1 \neq \mathbf{p}_1, \mathbf{p}'_1 + d(\mathbf{p}'_1) \neq \mathbf{p}_2. \quad (7.2)$$

Si deux pixels différents ont le même correspondant, alors la contrainte d'unicité n'est pas vérifiée. Cependant, cette contrainte peut ne pas être respectée dans certaines situations : en effet, quand un plan de la scène est très incliné par rapport à l'une des deux caméras, l'effet de raccourcissement peut apparaître et ainsi tous les pixels n'ont pas forcément un correspondant unique. Un exemple est fourni sur la figure 7.1.(b).

7.8 Contrainte d'ordre

La contrainte d'ordre (cf. figure 7.2.(a)) est définie par :

$$\text{Si } \mathbf{p}_1 + d(\mathbf{p}_1) = \mathbf{p}_2 \text{ et } \mathbf{p}'_1 + d(\mathbf{p}'_1) = \mathbf{p}'_2 \text{ alors } (y_1 - y'_1)(y_2 - y'_2) \geq 0. \quad (7.3)$$

Elle signifie que l'ordre des pixels de l'image I_1 doit être le même que celui de leurs correspondants. Cette contrainte peut également ne pas être respectée lorsqu'il y a, dans la scène, des plans transparents fortement inclinés par rapport au plan de l'une des caméras. Un exemple est montré sur la figure 7.2.(b).

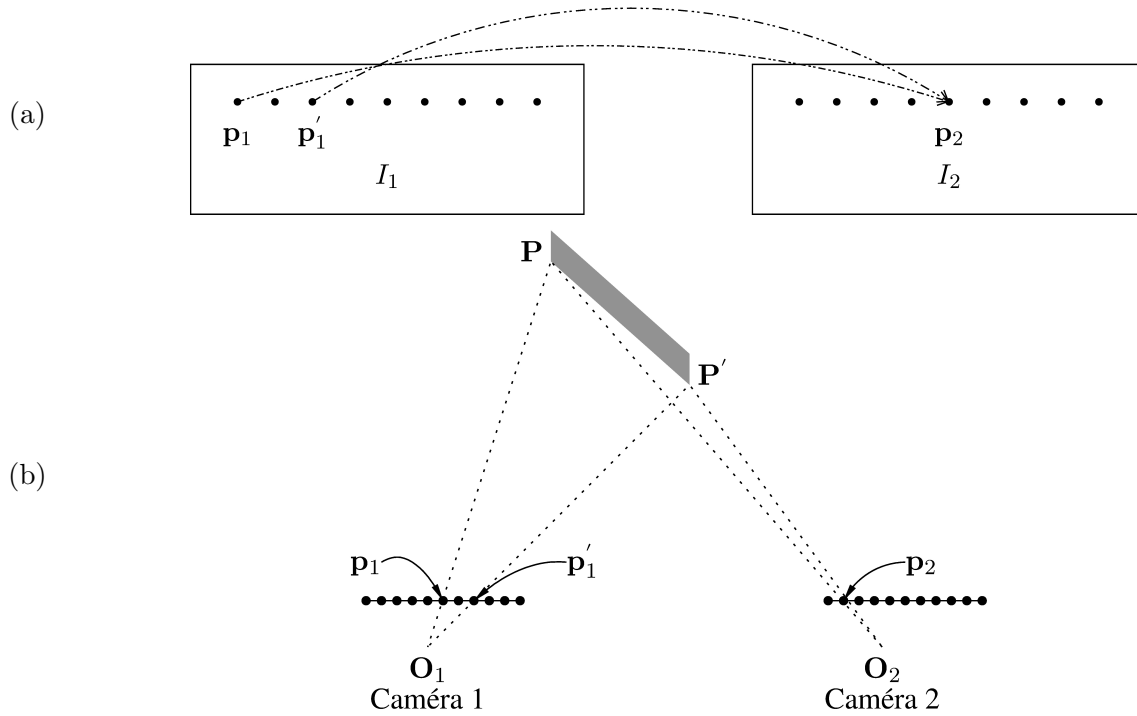


FIGURE 7.1 – Contrainte d’unicité – Dans la figure (a), nous pouvons voir les correspondances entre p_1^{i,j_1} et $p_2^{i,v}$ et entre p_1^{i,j_2} et $p_2^{i,v}$ qui ne respectent pas la contrainte d’unicité. En (b), Il s’agit d’une vue de dessus d’un capteur binoculaire en configuration parallèle et d’une scène. C’est un exemple de violation de la contrainte provoquée par un raccourcissement : tous les points qui se trouvent sur le segment de droite $[P_1P_2]$ se projettent sur le pixel $p_2^{i,v}$ dans l’image I_2 .

7.9 Contrainte de symétrie

Un des affinements des résultats d’une mise en correspondance les plus employés est la vérification bidirectionnelle (cf. figure 7.3.(a)), qui met en œuvre la contrainte de symétrie. Cette contrainte binaire s’écrit :

$$\text{Si } p_1 + d(p_1) = p_2 \text{ alors } p_2 + d(p_2) = p_1. \quad (7.4)$$

Deux mises en correspondance sont effectuées, de I_1 vers I_2 puis de I_2 vers I_1 . Si un pixel p_1 a pour correspondant le pixel p_2 , alors, lors de la seconde mise en correspondance, p_2 doit avoir pour correspondant p_1 . Si ce n’est pas le cas, la contrainte de symétrie n’est pas respectée. Elle est définie comme la conjonction de la **contrainte d’unicité de I_1 vers I_2 puis de I_2 vers I_1** . La **contrainte de symétrie assure la contrainte d’unicité**, ce qui signifie que si un ensemble de correspondances vérifient la contrainte de symétrie, alors elles vérifient aussi la contrainte d’unicité.

Cette contrainte est très forte et, dans certains travaux, les auteurs ont proposé une version qui tolère une erreur de quelques pixels que l’on appelle la contrainte de symétrie faible (cf. figure 7.3.(b)) et qui est définie par :

$$\text{Si } p_1 + d(p_1) = p_2 \text{ et } p_2 + d(p_2) = p_1' \text{ alors } |y_1 - y_1'| < T_j, \quad (7.5)$$

avec T_j est un seuil à fixer. En général, T_j correspond à 1 ou 2 pixel(s).

Les contraintes d’ordre et d’unicité ont été très utilisées. Une variante de ces contraintes, la consistance faible, a même été proposée.

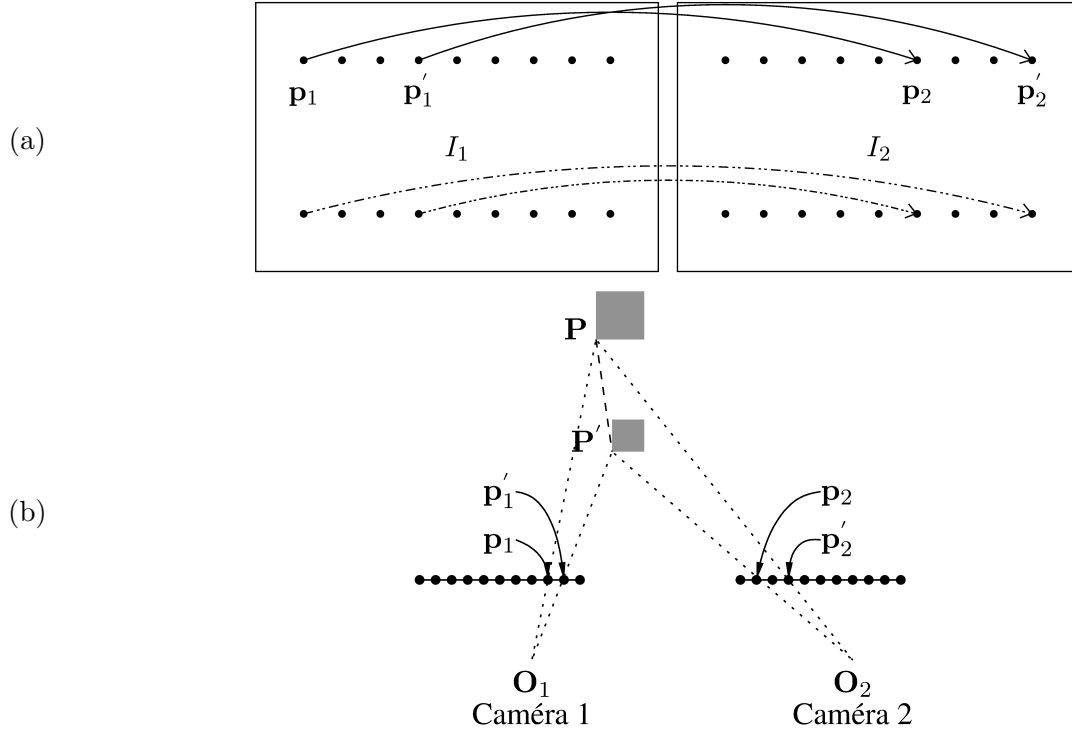


FIGURE 7.2 – Contrainte d'ordre – En (a), nous pouvons voir deux correspondances qui respectent la contrainte d'ordre et deux correspondances qui ne la respectent pas. En (b), une vue de dessus d'un capteur binoculaire en configuration parallèle et d'une scène, montre un exemple de violation de la contrainte d'ordre (le plan transparent vertical passant par les points \mathbf{P}_1 et \mathbf{P}_2 , représenté en tirets, est fortement incliné par rapport au plan des images).

7.10 Consistance faible

La consistance faible s'appuie sur la contrainte d'unicité et la contrainte d'ordre. Elle est donc binaire et elle est définie par :

$$\begin{aligned} &\text{Si } \mathbf{p}_1^{i,j} + d(\mathbf{p}_1^{i,j}) = \mathbf{p}_2^{i,v} \text{ alors} \\ &\forall j' > j \quad \mathbf{p}_1^{i,j'} + d(\mathbf{p}_1^{i,j'}) \neq \mathbf{p}_2^{i,v} \text{ et } \forall v' < v \quad \mathbf{p}_2^{i,v'} + d(\mathbf{p}_2^{i,v'}) \neq \mathbf{p}_1^{i,j}. \end{aligned} \quad (7.6)$$

Cela signifie que si $\mathbf{p}_1^{i,j}$ a pour correspondant $\mathbf{p}_2^{i,v}$ alors la contrainte de symétrie doit être vérifiée pour tous les pixels sur la même ligne que $\mathbf{p}_1^{i,j}$ tels que leurs indices de colonne soient supérieurs à j et pour tous les pixels sur la même ligne que $\mathbf{p}_2^{i,v}$ tels que leurs indices de colonnes soient inférieurs à v . Cette contrainte a été proposée pour tenter de limiter l'effet des contraintes d'unicité et d'ordre, c'est-à-dire pour interdire moins de correspondances.

7.11 Synthèse et comparaison des contraintes d'unicité, de symétrie, d'ordre et de consistance faible

Toutes ces contraintes peuvent être utilisées pour rejeter des appariements ou conditionner la mise en correspondance, c'est-à-dire, diminuer l'espace de recherche ou pénaliser certaines correspondances. Pour illustrer les différences entre toutes ces contraintes (unicité, ordre, symétrie et consistance faible), nous pouvons étudier une correspondance entre deux pixels et analyser l'impact de chaque contrainte, si

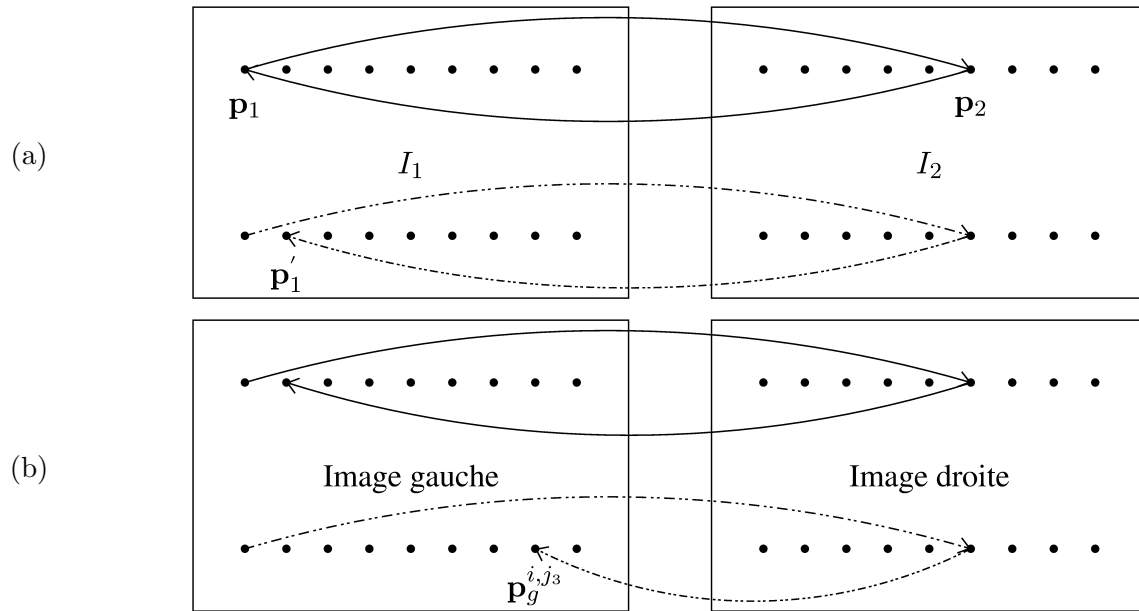


FIGURE 7.3 – Contrainte de symétrie (vérification bidirectionnelle) – La contrainte de symétrie est illustrée en (a) et nous pouvons la comparer à la contrainte de symétrie faible en (b). Nous pouvons remarquer que les correspondances entre $\mathbf{p}_1^{i,j1}$ et $\mathbf{p}_2^{i,v}$ et entre $\mathbf{p}_2^{i,v}$ et $\mathbf{p}_1^{i,j2}$ sont rejetées lorsqu'on applique la contrainte de symétrie alors qu'elles ne le sont pas quand il s'agit de la contrainte de symétrie faible.

elle est appliquée, sur l'appariement des autres pixels de l'image. Sur la figure 7.4, on voit que l'effet de la consistance faible est beaucoup moins important que celui de la contrainte d'ordre et qu'elle permet de tolérer certaines configurations dans lesquelles la contrainte d'unicité et la contrainte d'ordre ne sont pas parfaitement respectées, cf. les exemples fournis dans les sections ??.

7.12 Limite du gradient de disparité

Cette contrainte binaire s'appuie sur l'hypothèse d'une continuité dans les disparités, ce qui implique que le gradient de disparité, noté G_{disp} , ne doit pas dépasser une certaine valeur. Une approximation du gradient de disparité est définie comme la différence des disparités divisée par la **séparation cyclopéenne** (distance entre les milieux des paires formées par les pixels correspondants dans les images I_1 et I_2). Cette contrainte est définie par :

$$\begin{aligned} &\text{Si } \mathbf{p}_1 + d(\mathbf{p}_1) = \mathbf{p}_2 \text{ et } \mathbf{p}'_1 + d(\mathbf{p}'_1) = \mathbf{p}'_2 \text{ alors} \\ &G_{disp} = \frac{\|(\mathbf{p}_1 - \mathbf{p}'_1) - (\mathbf{p}_2 - \mathbf{p}'_2)\|}{\left\| \frac{1}{2} ((\mathbf{p}_1 + \mathbf{p}_2) - (\mathbf{p}'_1 + \mathbf{p}'_2)) \right\|} < T_G, \end{aligned} \quad (7.7)$$

où T_G est un seuil à fixer.

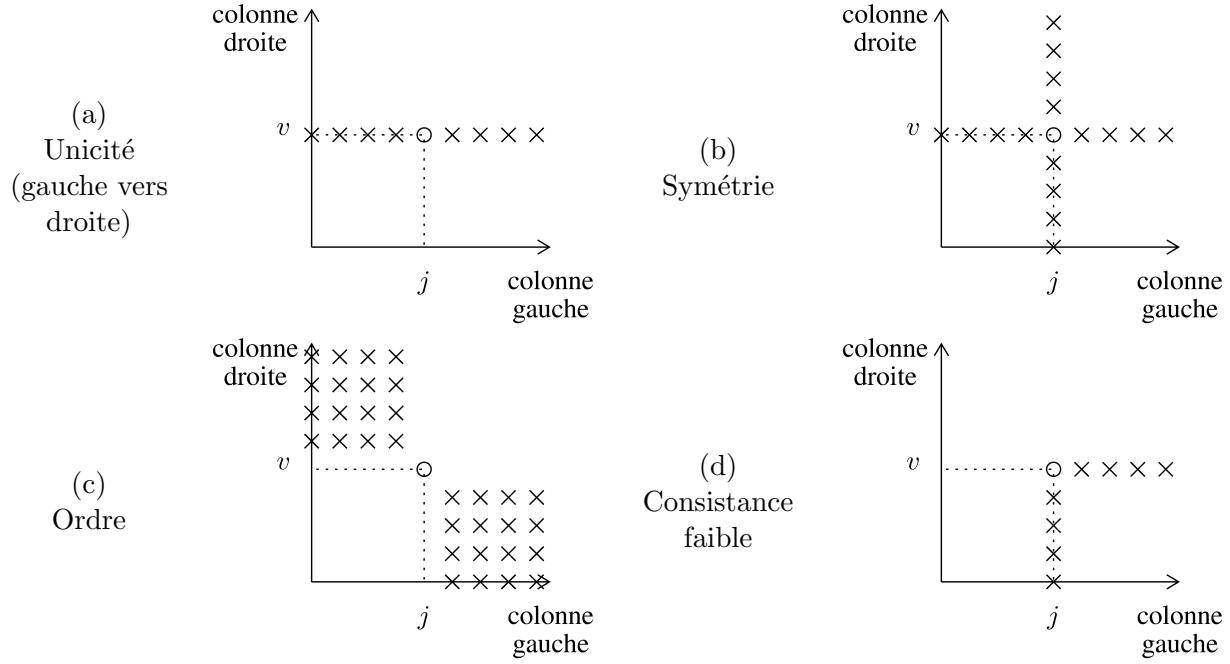


FIGURE 7.4 – Comparaison des contraintes – Ces schémas illustrent l’impact des contraintes sur l’ensemble des appariements possibles, lorsque la correspondance entre un pixel de colonne j , dans l’image gauche, et un pixel de colonne v , dans l’image droite, a été établie. Les axes des abscisses et des ordonnées correspondent respectivement à la colonne des pixels de l’image de gauche et de droite sur leurs droites épipolaires associées. Dans l’exemple de la contrainte d’unicité, en (a), si on établit la correspondance entre les pixels de colonnes j et v , alors toutes les correspondances sur la colonne v sont interdites. Grâce aux figures (a) et (b), on peut illustrer que la contrainte de symétrie est bien équivalente à la contrainte d’unicité appliquée de la gauche vers la droite et de la droite vers la gauche. La contrainte de symétrie implique aussi la contrainte de consistance faible. La contrainte d’ordre (c) est la plus forte, dans le sens où elle permet d’interdire le plus grand nombre de correspondances. Les contraintes d’unicité (a) et de consistance faible (d) sont les moins fortes.

7.13 Contrainte de rang

Elle s’appuie sur la comparaison du niveau de gris du pixel central avec celui des pixels de la zone d’agrégation et se traduit par la règle suivante :

$$\begin{aligned}
 &\text{Si } \mathcal{C}(\mathbf{p}_1) = \mathbf{p}_2 \text{ alors} \\
 &\forall \mathbf{p}'_1 \in \text{ZAC}^c(\mathbf{p}_1), \mathbf{p}'_2 \in \text{ZAC}^c(\mathbf{p}_2) \text{ tels que } y'_1 - y_1 = y'_2 - y_2 \\
 &(I_1(x'_1, y'_1) - I_1(x_1, y_1))(I_2(x'_2, y'_2) - I_2(x_2, y_2)) > 0.
 \end{aligned} \tag{7.8}$$

Le terme $\text{ZAC}^c(\mathbf{p}_l)$ correspond à l’ensemble des pixels qui se trouvent dans la zone d’agrégation de la contrainte du pixel \mathbf{p}_l . Cette contrainte suppose que le signe de la différence entre les niveaux de gris du pixel étudié et de ceux de sa zone d’agrégation doit être le même pour \mathbf{p}_1 et \mathbf{p}_2 .

(a)	(b)	(c)
$\begin{pmatrix} 250 & 200 & 198 \\ 100 & \mathbf{104} & 110 \\ 50 & 48 & 78 \end{pmatrix}$	$\begin{pmatrix} 255 & 205 & 203 \\ 105 & \mathbf{109} & 115 \\ 55 & 53 & 93 \end{pmatrix}$	$\begin{pmatrix} 255 & 205 & \mathbf{100} \\ 105 & \mathbf{109} & \mathbf{105} \\ 55 & 53 & \mathbf{110} \end{pmatrix}$

FIGURE 7.5 – Contrainte de rang – Sur cette figure, chaque matrice représente le niveau de gris du pixel étudié, au centre, en gras, et les niveaux de gris des pixels de la zone d’agrégation (ici, nous avons choisi une fenêtre de taille 3×3 centrée sur le pixel considéré). En (a), il s’agit du pixel dont on cherche le correspondant et de son voisinage 3×3 . En (b) et (c), il s’agit de deux correspondants possibles et de leurs zones d’agrégation respectives. Entre (a) et (b), la contrainte de rang est respectée. Entre (a) et (c), les pixels de la colonne de droite (en italique), ne respectent pas la contrainte.

7.14 Contrainte de continuité figurale

La contrainte de continuité figurale, contrainte binaire, est définie par :

$$\begin{aligned} &\text{Soient } \mathcal{C}_1 \text{ un contour dans l'image } I_1 \text{ et } \mathcal{C}_2 \text{ un contour dans l'image } I_2. \text{ On note :} \\ &\mathcal{P}_{\mathcal{C}_1} = \{\mathbf{p}_1 \in \mathcal{C}_1\} \text{ et } \mathcal{P}'_{\mathcal{C}_1} = \{\mathbf{p}_1 \in \mathcal{P}_{\mathcal{C}_1} \mid \mathcal{C}(\mathbf{p}_1) \in \mathcal{C}_2\}. \\ &\text{Si } \text{card}(\mathcal{P}'_{\mathcal{C}_1}) \geq \frac{1}{2} \text{card}(\mathcal{P}_{\mathcal{C}_1}) \text{ alors } \forall \mathbf{p}'_1 \text{ tels que } \mathbf{p}'_1 \in \mathcal{P}_{\mathcal{C}_1}, \mathcal{C}(\mathbf{p}'_1) \in \mathcal{C}_2. \end{aligned} \quad (7.9)$$

Si un pixel \mathbf{p}_1 appartient à un contour \mathcal{C}_1 dont la majorité des pixels ont été appariés avec des pixels appartenant au même contour \mathcal{C}_2 , dans l’image I_2 , alors le correspondant du pixel \mathbf{p}_1 appartient à \mathcal{C}_2 .

7.15 Notion d’ambiguïté et d’imprécision

Une dernière manière de corriger/contraindre les correspondances de points est d’utiliser l’ambiguïté et l’imprécision. Ces deux critères évalués proviennent d’une analyse de la courbe constituée par les scores de corrélation sur la zone de recherche. Nous précisons que pour décrire ces critères, nos illustrations, cf. figures 7.6 et 7.7, concernent le cas de la mesure ZNCC (dont l’intervalle de variation est $[-1; 1]$).

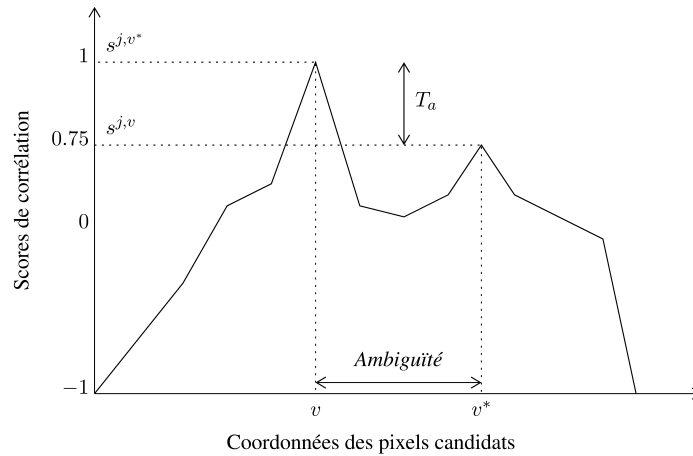


FIGURE 7.6 – Mesure d’ambiguïté – Nous illustrons le calcul de l’ambiguïté, en utilisant la mesure ZNCC et en supposant que l’on cherche un maximum. Nous avons pris $T_a = 0.25$.

L’ambiguïté de la mesure permet de mesurer la gravité d’une grosse erreur que l’on a pu commettre en choisissant un pixel alors qu’il en existe un autre pour lequel le score était proche. Plus

précisément, s'il existe un score de corrélation $s^{j,v}$ proche de s^{j,v^*} , c'est-à-dire, tel que $|s^{j,v} - s^{j,v^*}| \leq T_a$ (T_a est un seuil à fixer) alors l'ambiguïté correspond à $|v^* - v|$.

L'imprécision de la mesure permet de quantifier l'erreur de localisation, c'est-à-dire la gravité d'une petite erreur que l'on a pu commettre. Plus précisément, il y a imprécision s'il existe v_1 et v_2 tels que $v^* \in [v_1; v_2]$ et quel que soit $v \in [v_1; v_2]$ le score $s^{j,v}$ est proche du score s^{j,v^*} , c'est-à-dire, tels que $|s^{j,v} - s^{j,v^*}| < T_i$ (T_i est un seuil à fixer) alors l'imprécision correspond à $|v_1 - v_2|$.

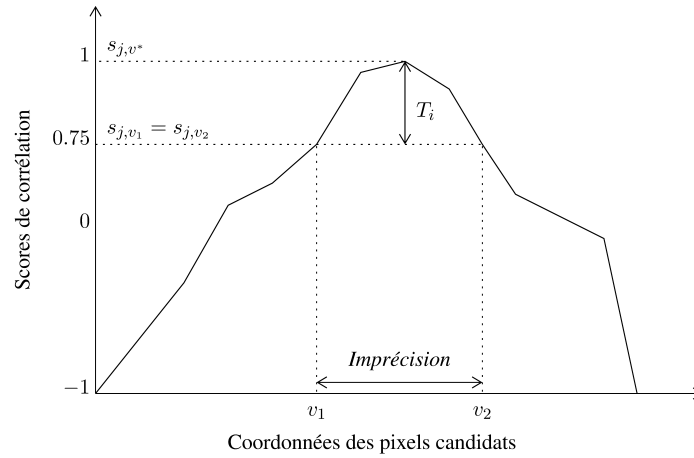


FIGURE 7.7 – Mesure d'imprécision – Nous illustrons le calcul de l'imprécision, en utilisant la mesure ZNCC et en supposant que l'on cherche un maximum. Nous avons pris $T_i = 0.155$.

Chapitre 8

Algorithme de mise en correspondance associé au détecteur SIFT

8.1 Descripteur

L'objectif du descripteur proposé pour effectuer la mise en correspondance est d'être suffisamment discriminant/unique pour permettre une mise en correspondance la plus fiable possible. Ceci est facilement assuré par le fait que les auteurs utilisent un descripteur de dimension 128 alors que la plupart des autres approches s'appuient sur des données 1D (niveaux de gris), 3D (couleurs) voire 5D (couleurs + position).

8.1.1 Affectation d'une orientation

Une fois les extremums détectés, les auteurs attribuent des orientations principales à chaque extremum en suivant les points suivants :

- (1) **Estimation de la norme et de l'orientation du vecteur gradient** – Nous rappelons que L correspond à l'image de départ filtrée (pour un niveau de filtrage donné, cf. équation 4.16, page 51). Ainsi, nous pouvons simplement calculer :

$$\begin{aligned} m(x, y) &= \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2} \\ \theta(x, y) &= \tan^{-1} \left(\frac{L(x, y+1) - L(x, y-1)}{L(x+1, y) - L(x-1, y)} \right) \end{aligned} \quad (8.1)$$

- (2) **Construction d'un histogramme local des orientations θ** – Pour le calcul de cet histogramme, la somme des éléments de même orientation est pondérée par la norme m et par une gaussienne, centrée sur le point étudié avec $\sigma = 1.5$. L'objectif de cette pondération est de minimiser l'influence des voisins les plus éloignés ainsi que l'influence des voisins dont le gradient est faible. En pratique, l'histogramme est discrétisé (les auteurs utilisent 36 orientations pour couvrir les 360 degrés), dans la figure 8.1.(a), on donne exemple d'histogramme sur 8 orientations uniquement. De plus, le voisinage pris en compte est de taille 8×8 .
- (3) **Sélection des orientations** – La plus grande valeur de cet histogramme, l'orientation principale, ainsi que toutes celles au dessus de 80% de ce maximum, les orientations secondaires sont considérées comme les orientations du point. Ainsi, on peut avoir plusieurs orientations pour le même point. En pratique, les auteurs signalent que seuls 15% au maximum des points, se voient attribués plusieurs orientations.
- (4) **Interpolation des orientations** – Pour mieux localiser le maximum, une interpolation est réalisée au voisinage de chaque maximum, en prenant en compte les 3 valeurs les plus proches de ce maximum, cf. figure 8.1.(b).

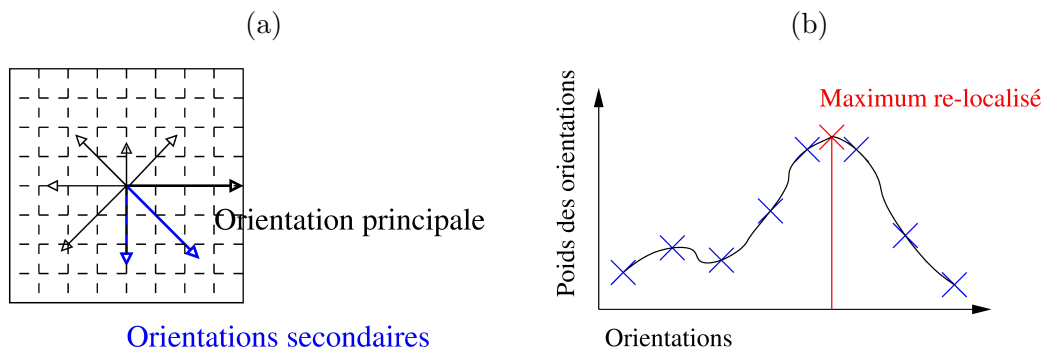


FIGURE 8.1 – Histogramme des orientations pour SIFT – Un exemple sur 8 orientations seulement où il y a une orientation principale ainsi que 2 orientations secondaires (80% de l’orientation principale). Dans cet exemple, on utilise un voisinage 8×8 , mais, dans la pratique le descripteur SIFT s’appuie sur un voisinage 4×4 .

8.1.2 Calcul du descripteur

Le descripteur proposé s’inspire du modèle visuel primaire humain qui est plus sensible aux changements d’orientation du gradient qu’aux déplacements (pour permettre la reconnaissance d’un objet quelque soit le point de vue). Ainsi, le descripteur utilisé est construit de la manière suivante :

- (1) **Utilisation de l’orientation et de l’histogramme déjà estimés** – On exploite les mêmes données, cf. figure 8.1.(a), à trois différences près :
 - (a) L’histogramme ne possède plus 36 mais seulement 8 orientations.
 - (b) On applique une rotation, relative à l’orientation du point d’intérêt, aux coordonnées ainsi qu’à l’orientation de chaque point (ce qui est estimé comme dans la figure 8.1.(b)). Ceci, dans le but de rendre la description invariante aux rotations.
 - (c) La participation de chaque point dans le calcul de l’histogramme local est pondérée par sa distance au point d’intérêt caractérisé (et non plus de simple poids gaussiens), c’est-à-dire par $1 - d$ dans chaque dimension où d est la distance du point au point central.
- (2) **Première normalisation du vecteur-descripteur** – Les valeurs sont recadrées entre 0 et 1 afin que le descripteur soit invariant aux changements affines d’illumination.
- (3) **Seuillage des forts gradients** – Cette étape est réalisée afin de prendre en compte des changements d’illumination non-linéaires. Les auteurs ont expérimentalement choisi un seuil très restrictif à 0.15. Ceci implique la normalisation suivante.
- (4) **Deuxième normalisation du vecteur-descripteur** – De la même manière que la première normalisation.

Les auteurs proposent un descripteur de taille 128 en utilisant un voisinage 4×4 et 8 orientations.

8.2 Appariement

Pour la mise en correspondance, une simple distance peut être utilisée. Dans la littérature, on peut citer l’utilisation d’un produit scalaire (entre les deux descripteurs de 128), de la mesure ZNCC ou d’une distance L_2 .

Chapitre 9

Suivi de zones d'intérêt par KLT (Kanade–Lucas–Tomasi)

9.1 Éléments de bibliographie

Cette partie s'appuie sur les articles suivants :

- Article original de Lucas et Kanade publié en 1981 [Lucas 81] ;
- Rapport complémentaire de Tomasi et Kanade publié en 1991 [Tomasi 91] ;
- Papier qui étend la méthode de Shi et Tomasi aux diverses transformations, publié en 1994 [Shi 94], et en particulier le rapport complémentaire publié préalablement en 1993 [Shi 93] ;
- Mais surtout le site de Birchfield qui regroupe tout l'historique, ainsi qu'une implémentation : <http://www.ces.clemson.edu/stb/klt/> [Birchfield 97].

9.2 Contexte–Introduction

Des modèles basés sur le changement de niveaux de gris ont essentiellement été abordés. Ces modèles se montrent efficaces dans de nombreux cas, mais, ils nécessitent un bon choix des primitives et lorsque l'invariance de niveau de gris de l'objet suivi n'est plus respectée, le suivi devient moins performant. **Le problème majeur** des méthodes basées sur une détection puis un suivi est que le détecteur est souvent proposé sans prendre en compte les contraintes de la méthode de suivi utilisée par la suite.

Pour remédier à ce problème, les auteurs proposent d'introduire la méthode de suivi, puis, de déterminer quelles zones de l'image sont pertinentes pour réaliser le suivi suivant cette méthode. **Il s'agit d'une nouvelle définition de points d'intérêts : ce sont les points/régions qui se distinguent des autres par le fait qu'ils sont adaptés à un modèle de suivi donné.**

Rappelons que le problème du recalage ou suivi, nous l'avons déjà vu, peut se formuler, en **1D**, pour chaque point **p**, de la manière suivante :

$$d = \operatorname{argmin}_E \quad (9.1)$$

avec $E = \operatorname{Dist}(I_t(x + d), I_{t+1}(x))$,

où

- d est le déplacement (ou disparité), ici on suppose une translation de la gauche vers la droite ;
- Dist est une distance à choisir, sachant que la plupart du temps, le choix se limite à L_1 , L_2 **ou la corrélation croisée centrée normalisée.**

On peut également choisir $I_t(x - d)$, si on suppose un déplacement de la droite vers la gauche (ce qui correspond à un mouvement de caméra de la gauche vers la droite).

Voici un ensemble de remarques sur la minimisation de cette expression :

- (1) Il est impossible de résoudre ce problème en effectuant une **recherche exhaustive sur d** car la **complexité serait en $O(M^2N^2)$** (M étant la zone de recherche et N la taille de l'image).
- (2) Il existe des méthodes dites **gloutonnes itératives**, *hill-climbing*, où on part d'une solution et on effectue de petites variations locales. Ces méthodes posent le problème de la nécessité d'une **bonne estimation initiale**. Ces méthodes sont efficaces et rapides mais sont vite peu performantes en cas de changements d'échelles ou de déplacements trop importants. Une fois de plus, elles peuvent également s'avérer coûteuses.
- (3) Une façon de régler les problèmes liés au temps de calcul est d'utiliser une approche **pyramidale ou multi-échelle**.

En observant la figure 9.1, on peut conclure que le déplacement que l'on cherche à estimer est en relation avec la dérivée de l'image.

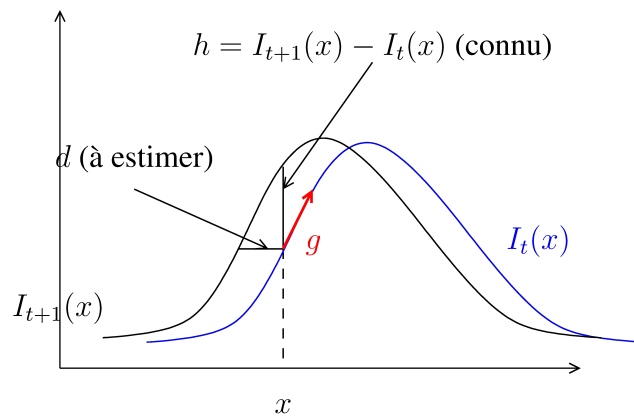


FIGURE 9.1 – Suivi par KLT – Ce que l'on cherche à estimer en fonction de ce que l'on connaît, c'est-à-dire d en fonction de g et h connus.

Ainsi les caractéristiques de la méthode proposée par KLT sont les suivantes :

- Utilisation des gradients de l'image ;
- Algorithme itératif s'appuyant sur la méthode de Newton-Raphson ;
- Approche multi-résolution ou pyramidale.

C'est-à-dire qu'on combine les approches de types (2) et (4), et, dans la partie de ce cours, on va surtout aborder les aspects du (2).

9.3 Recalage en 1D et sa solution naïve

Il s'agit de la première solution du recalage en 1D.

D'après la définition de la dérivée en x , et si on suppose d (notre déplacement) petit :

$$I'_t(x) \approx \frac{I_t(x+d) - I_t(x)}{d} = \frac{I_{t+1}(x) - I_t(x)}{d}, \quad (9.2)$$

puisque l'on suppose une invariance photométrique des éléments suivis et donc $I_{t+1}(x) = I_t(x+d)$. Ainsi d peut s'exprimer :

$$d \approx \frac{I_{t+1}(x) - I_t(x)}{I'_t(x)}. \quad (9.3)$$

De façon similaire sur une portion de courbe, on obtient l'approximation suivante :

$$d \approx \frac{1}{N} \sum_x \frac{I_{t+1}(x) - I_t(x)}{I'_t(x)},$$

avec N le nombre d'éléments considérés sur la fenêtre considérée. Cette expression suppose que le déplacement d est plus petit que la taille de fenêtre considérée. Cette somme peut être améliorée en modifiant les poids. En effet, l'approximation faite dans l'équation (9.2), suppose que l'on a une courbe linéaire. En conséquence, on souhaite que les points se trouvant sur une forte concavité de la courbe aient peu d'influence. Ainsi, plus la dérivée seconde est élevée (c'est-à-dire plus la convexité de la courbe est importante), plus le poids est faible. On pose :

$$d \approx \frac{1}{\sum_x w(x)} \sum_x w(x) \frac{I_{t+1}(x) - I_t(x)}{I'_t(x)}, \quad (9.4)$$

avec des poids $w(x)$ inversement proportionnels à la norme de la dérivée seconde :

$$w(x) = \frac{1}{\|I'_{t+1}(x) - I'_t(x)\|} \text{ avec } I''(x) \approx \frac{I'_{t+1}(x) - I'_t(x)}{d}. \quad (9.5)$$

en supprimant d qui sera constant pour tous les poids.

Cette expression de d met en évidence : le déplacement peut être ainsi estimé en projetant la différence d'intensité observée sur le gradient. En conséquence, si le déplacement a lieu suivant le gradient, il sera difficile d'estimer le déplacement.

Au final, d peut être estimé de manière itérative, en utilisant la méthode de **Newton-Raphson** (qui permet de trouver le zéro d'une fonction), en suivant ces étapes :

$$(1) \quad d_0 = 0;$$

$$(2)$$

$$d_{k+1} = d_k + \frac{\sum_x w(x) \frac{I_{t+1}(x) - I_t(x + d_k)}{I'_t(x + d_k)}}{\sum_x w(x)}. \quad (9.6)$$

Il faut déterminer la condition d'arrêt s'appuyant sur le nombre d'itérations et la convergence de l'algorithme.

Les inconvénients de cette approche sont les suivants :

$$(1) \quad \text{L'approximation linéaire en 2D est différente.}$$

$$(2) \quad \text{L'équation (9.3) n'est pas définie pour } I'_t(x) = 0.$$

9.4 Solution en 1D généralisable au 2D

Voici une seconde solution en 2D.

Ainsi, une approche légèrement différente est d'utiliser un développement de Taylor à l'ordre 1 et d'exprimer $I_t(x + d)$, de la manière suivante :

$$I_t(x + d) \approx I_t(x) + dI'_t(x), \quad (9.7)$$

en négligeant le reste (qu'on suppose nul car le déplacement est petit. Ainsi, on remplace l'équation (9.3) par l'équation (9.7). Sachant que l'on cherche à minimiser cette expression, l'équation (9.1) pour laquelle on a choisi une norme L_2 pondérée devient :

$$E = \sum_x w(x) (I_t(x + d) - I_{t+1}(x))^2 \approx \sum_x (I_t(x) + dI'_t(x) - I_{t+1}(x))^2 \quad (9.8)$$

Minimiser cette expression peut être également résolue en cherchant la solution de :

$$0 = \frac{\partial E}{\partial d} \approx \sum_x w(x) I'_t(x) (I_t(x) + dI'_t(x) - I_{t+1}(x)). \quad (9.9)$$

Ce qui nous permet d'extraire l'approximation de d suivante :

$$d \approx \frac{\sum_x w(x) I'_t(x) (I_{t+1}(x) - I_t(x))}{\sum_x w(x) I'_t(x)^2}. \quad (9.10)$$

Cette nouvelle expression est généralisable en 2D, et elle évite, de manière plus probable, une division par 0. On peut toutefois remarquer que l'on risque d'avoir un problème dans une zone parfaitement homogène.

La forme itérative associée à l'équation (9.6) est donnée par :

$$(1) \quad d_0 = 0;$$

$$(2)$$

$$d_{k+1} = d_k + \frac{\sum_x w(x) I'_t(x + d_k) (I_{t+1}(x) - I_t(x + d_k))}{\sum_x w(x) I'_t(x + d_k)^2}, \quad (9.11)$$

avec les poids définis par l'équation (9.5).

Voici quelques remarques sur les **performances de cette méthode** :

- (1) En présence de hautes fréquences, le **convergence de l'algorithme est compromise** et c'est la raison pour laquelle, les auteurs suggèrent un **pré-filtrage** de l'image permettant de supprimer les hautes fréquences.
- (2) Le problème engendré par ce pré-filtrage est de **perdre certains détails** et c'est la raison pour laquelle, entre autres, les auteurs préconisent d'utiliser une **approche pyramidale ou multi-échelle**.

9.5 Solution en 2D

L'expression (9.8) peut être réécrite de la même manière en 2D, c'est-à-dire :

$$E = \sum_{\mathbf{x}} w(\mathbf{x}) (I_t(\mathbf{x} + \mathbf{d}) - I_{t+1}(\mathbf{x}))^2,$$

où \mathbf{x} , \mathbf{d} ne sont plus des scalaires, mais des vecteurs de dimensions $N \times 1$ (généralement 2). De manière analogue à (9.7), on peut écrire :

$$I_t(\mathbf{x} + \mathbf{d}) \approx I_t(\mathbf{x}) + \mathbf{g}^T \mathbf{d}, \quad (9.12)$$

où $\mathbf{g} = \frac{\partial I_t(\mathbf{x})}{\partial \mathbf{x}}$ est un vecteur correspondant au gradient (en 2D) :

$$\mathbf{g} = \frac{\partial I_t(\mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial I_t(\mathbf{x})}{\partial x} & \frac{\partial I_t(\mathbf{x})}{\partial y} \end{bmatrix}^T.$$

Ainsi, l'équation (9.9) devient :

$$\begin{aligned} 0 &= \frac{\partial E}{\partial \mathbf{d}} \\ &\approx \frac{\partial}{\partial \mathbf{d}} \left(\sum_{\mathbf{x}} w(\mathbf{x}) [I_t(\mathbf{x}) + \mathbf{g}^T \mathbf{d} - I_{t+1}(\mathbf{x})]^2 \right) \\ &= \sum_{\mathbf{x}} w(\mathbf{x}) \mathbf{g} [I_t(\mathbf{x}) + \mathbf{g}^T \mathbf{d} - I_{t+1}(\mathbf{x})], \end{aligned} \quad (9.13)$$

d'où, l'expression de \mathbf{d} donnée par l'équation (9.13) :

$$\mathbf{d} \approx \left[\sum_{\mathbf{x}} w(\mathbf{x}) \mathbf{g} (I_{t+1}(\mathbf{x}) - I_t(\mathbf{x})) \right] \left[\sum_{\mathbf{x}} w(\mathbf{x}) \mathbf{g} \mathbf{g}^T \right]^{-1}, \quad (9.14)$$

Pour la suite, on pose :

$$\mathbf{G} = \left[\sum_{\mathbf{x}} w(\mathbf{x}) \mathbf{g} \mathbf{g}^T \right]^{-1},$$

et :

$$e = \left[\sum_{\mathbf{x}} w(\mathbf{x}) \mathbf{g} (I_{t+1}(\mathbf{x}) - I_t(\mathbf{x})) \right],$$

ce qui donne la ré-écriture suivante de l'équation (9.14) :

$$\mathbf{d} = \mathbf{e} \mathbf{G}. \quad (9.15)$$

De la même manière, on peut appliquer la résolution par approche itérative, en utilisant l'équation (9.11) :

$$\mathbf{d}_{k+1} = \mathbf{d}_k + \left[\sum_{\mathbf{x}} w(\mathbf{x}) \frac{\partial I_t(\mathbf{x} + \mathbf{d}_k)}{\partial \mathbf{x}} (I_{t+1}(\mathbf{x}) - I_t(\mathbf{x} + \mathbf{d}_k)) \right] \left[\sum_{\mathbf{x}} \frac{\partial I_t(\mathbf{x} + \mathbf{d}_k)^T}{\partial \mathbf{x}} \frac{\partial I_t(\mathbf{x} + \mathbf{d}_k)}{\partial \mathbf{x}} \right]^{-1}, \quad (9.16)$$

Les expérimentations exposées dans la littérature, mettent en évidence le fait que seules **5 itérations suffisent pour estimer le déplacement**.

9.6 Généralisation aux transformations affines

Pour généraliser cette méthode aux transformations affines, il suffit de poser que :

$$I_{t+1}(\mathbf{x}) = I_t(\mathbf{A}\mathbf{x} + \mathbf{d}), \quad (9.17)$$

avec

$$\mathbf{A} = \begin{pmatrix} a_{xx} & a_{xy} \\ a_{yx} & a_{yy} \end{pmatrix}$$

qui représente n'importe quel type de transformations affines (rotation, mise à l'échelle, transvection) entre I_t et I_{t+1} . L'équation (9.12) devient :

$$E = \sum_{\mathbf{x}} w(\mathbf{x}) (I_t(\mathbf{A}\mathbf{x} + \mathbf{d}) - I_{t+1}(\mathbf{x}))^2, \quad (9.18)$$

et l'expression linéaire utilisée est la suivante :

$$I_t(\mathbf{A}\mathbf{x} + \mathbf{d}) \approx I_t(\mathbf{x}) + \mathbf{g}^T(\mathbf{A}\mathbf{x} + \mathbf{d}). \quad (9.19)$$

Si on calcule les dérivées partielles de l'équation (9.18), on obtient le système suivant :

$$\begin{aligned} 0 &= \frac{\partial E}{\partial \mathbf{A}} = \sum_{\mathbf{x}} w(\mathbf{x}) (I_t(\mathbf{A}\mathbf{x} + \mathbf{d}) - I_{t+1}(\mathbf{x})) \mathbf{g} \mathbf{x}^T \\ 0 &= \frac{\partial E}{\partial \mathbf{d}} = \sum_{\mathbf{x}} w(\mathbf{x}) (I_t(\mathbf{A}\mathbf{x} + \mathbf{d}) - I_{t+1}(\mathbf{x})) \mathbf{g}. \end{aligned} \quad (9.20)$$

Ainsi, en utilisant l'équation (9.19) dans l'équation (9.20), on obtient le système suivant :

$$\begin{aligned} \sum_{\mathbf{x}} w(\mathbf{x}) \mathbf{g} \mathbf{x}^T \mathbf{g}^T \mathbf{u} &= \sum_{\mathbf{x}} w(\mathbf{x}) (I_t(\mathbf{x}) - I_{t+1}(\mathbf{x})) \mathbf{g} \mathbf{x}^T \\ \sum_{\mathbf{x}} w(\mathbf{x}) \mathbf{g} \mathbf{g}^T \mathbf{u} &= \sum_{\mathbf{x}} w(\mathbf{x}) (I_t(\mathbf{x}) - I_{t+1}(\mathbf{x})) \mathbf{g}, \end{aligned} \quad (9.21)$$

avec

$$\mathbf{u} = \mathbf{Ax} + \mathbf{d}.$$

De la même manière que dans les autres sections, on peut estimer les paramètres de manière itérative en suivant les étapes suivantes, cf. [Shi 93] pour les détails de la résolution :

- (1) $\mathbf{A}_0 = \mathbf{1}$; $\mathbf{d}_0 = \mathbf{0}$; $I_t^0 = I_t(\mathbf{x})$.
- (2) Résoudre les équations (9.21) en remplaçant I_t^i par $I^{i-1}(\mathbf{A}_i\mathbf{x} + \mathbf{d}_i)$.

Estimer les 6 paramètres de la transformation complète est **coûteux** et lorsque le mouvement est plus simple, cela introduit même des **erreurs**. Ainsi, on a pour habitude d'estimer une **simple translation** lorsqu'on estime le mouvement entre images proches et l'estimation complète des 6 paramètres lorsqu'on estime le mouvement entre la première image de la séquence et l'image courante.

9.7 Généralisation aux changements de luminosité

De la même manière, ce modèle est généralisable aux changements de luminosité de la forme :

$$I_{t+1}(\mathbf{x}) = \alpha I_t(\mathbf{x}) + \beta. \quad (9.22)$$

Et ainsi l'énergie E à minimiser devient :

$$E = \sum_{\mathbf{x}} w(\mathbf{x}) (I_t(\mathbf{Ax} + \mathbf{d}) - (\alpha I_{t+1}(\mathbf{x}) + \beta))^2, \quad (9.23)$$

ce qui augmente encore le nombre de paramètres à estimer mais peut toujours s'appuyer sur le même type d'équation que l'équation (9.19).

9.8 Sélection des zones à suivre

À la suite de la présentation de cette méthode de suivi, deux nouvelles questions sont introduites :

- (1) Comment choisir la taille d'observation ?
- (2) Comment choisir les éléments à suivre ?

À ceci s'ajoutent deux questions subsidiaires :

- (1) Que se passe-t-il en présence de surfaces très fortement inclinées ?
- (2) Que se passe-t-il lorsque des objets sont occultés ?

La solution, simple envisagée est la suivante :

- L'apparence doit changer faiblement au cours de la séquence. Toute fenêtre dont l'apparence varie brusquement, c'est-à-dire que l'erreur résiduelle augmente significativement, est abandonnée. Ce choix de comportement permet d'éviter les problèmes liés à ces deux types de zones difficiles.
- Prise en compte des transformations plus complexes que la simple translation.
- Pour la taille d'observation, les auteurs préconisent simplement une petite taille de fenêtre (robuste aux occultations).

Ainsi, pour sélectionner les éléments à suivre, il faut prendre en compte les éléments de l'image faiblement affectés par la présence de bruit, c'est-à-dire que la zone considérée ne doit pas correspondre à une zone uniforme et doit être suffisamment texturée. Plus précisément cela impose la condition suivante sur la matrice \mathbf{G} , équation (9.15), si on pose λ_1 et λ_2 , ses deux valeurs propres, alors :

$$\min(\lambda_1, \lambda_2) > S_\lambda, \quad (9.24)$$

sachant que le seuil S_λ peut être fixé soit :

- en calculant les valeurs propres de zones uniformes préalablement identifiées ou,
- de manière plus automatique, en calculant toutes les valeurs propres minimales pour chaque point et en analysant l'histogramme de ces valeurs. On choisit la valeur de seuil permettant d'obtenir 2 modes significatifs de l'histogramme.

Bibliographie

- [Aanæs 12] H. AANÆS, A. LINDBJERG DAHL et K. STEENSTRUP PEDERSEN. *Interesting Interest Points*. *International Journal on Computer Vision, IJCV*, 97(1):18–35, 2012.
- [Abdi 07] H. ABDI. *Singular value decomposition (SVD) and generalized singular value decomposition*. *Encyclopedia of measurement and statistics*, pages 907–912, 2007.
- [Agarwal 11] S. AGARWAL, Y. FURUKAWA, N. SNAVELY, I. SIMON, B. CURLESS, S. SEITZ et R. SZELISKI. *Building Rome in a Day*. *Communications of the Association for Computing Machinery, ACM*, 54(10):105–112, 2011.
- [Amhaz 16] R. AMHAZ, S. CHAMBON, J. IDIER et V. BALTAZART. *Automatic Crack Detection on Two-Dimensional Pavement Images: An Algorithm Based on Minimal Path Selection*. *IEEE Transactions on Intelligent Transportation Systems, TITS*, 17(10):2718–2729, 2016.
- [Bakkay 18] M. C. BAKKAY, S. CHAMBON, C. LUBAT et S. N. BARSOTTI. *Automatic detection of individual and touching moths from trap images by combining contour-based and region-based segmentation*. *IET Computer Vision*, 12(2):138–145, 2018.
- [Bay 08] H. BAY, A. ESS, T. TUYTELAARS et L. VAN GOOL. *Speeded-Up Robust Features (SURF)*. *Computer Vision and Image Understanding*, 110(3):346–359, 2008.
- [Beaudet 78] P. R. BEAUDET. *Rotationnally invariant image operators*. Dans *International Conference on Pattern Recognition*, pages 579–583, 1978.
- [Birchfield 97] S. BIRCHFIELD. KLT: An implementation of the kanade-lucas-tomasi feature tracker. 1997. <http://www.ces.clemson.edu/~stb/klt/>.
- [Bleyer 10] M. BLEYER, C. ROTHER et P. KOHLI. *Surface Stereo with Soft Segmentation*. Dans *IEEE Conference on Computer Vision Pattern Recognition*, pages 1570–1577, juin 2010.
- [Bres 99] S. BRES et J.-M. JOLION. *Detection of Interest Points for Image Indexation*. Dans *Visual Information and Information Systems*, pages 427–435. Springer Berlin Heidelberg, 1999.
- [Brigger 99] P. BRIGGER, F. MÜLLER, K. ILLGNER et M. UNSER. *Centered Pyramids*. *IEEE Transactions on Image Processing, IP*, 8(9):1254–1264, septembre 1999.
- [Burt 83] P. J. BURT et E. H. ADELSON. *The Laplacian Pyramid as a Compact Image Code*. *IEEE Transactions on Communications*, 31(4):532–540, avril 1983.
- [Calonder 10] M. CALONDER, V. LEPETIT, C. STRECHA et P. FUA. *BRIEF: Binary Robust Independent Elementary Features*. Dans *European Conference on Computer Vision, ECCV*, volume 6314, 2010.
- [Canny 86] J. CANNY. *A Computational Approach To Edge Detection*. *PAMI*, 8(6):679–714, 1986.
- [Capel 04] D. CAPEL. *Image mosaicing, Chapitre Image Mosaicing and Super-resolution*, pages 47–79. Springer, London, 2004.
- [Chambon 11] S. CHAMBON. *Detection of Points of Interest for Geodesic Contours: Application on Road Images for Crack Detection*. Dans *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, VISAPP*, pages 210–213, 2011.

- [Dalal 05] N. DALAL et B. TRIGGS. *Histograms of oriented gradients for human detection*. Dans IEEE Conference on Computer Vision Pattern Recognition, pages 886–893, 2005.
- [Deng 07] H. DENG, W. ZHANG, E. MORTENSEN, T. DIETTERICH et L. SHAPIRO. *Principal Curvature-Based Region Detector for Object Recognition*. Dans IEEE Conference on Computer Vision Pattern Recognition, 2007.
- [Deriche 87] R. DERICHE. *Using Canny’s criteria to derive a recursively implemented optimal edge detector*. *IJCV*, 2(6):167–187, 1987.
- [Fischer 14] P. FISCHER et T. BROX. *Image Descriptors Based on Curvature Histograms*. Dans German Conference on Pattern Recognition, GCPR, pages 239–249, 2014.
- [Gales 10] G. GALES, A. CROUZIL et S. CHAMBON. *Complementarity of Feature Point Detectors*. Dans International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, VISAPP, pages 334–339, 2010.
- [Gales 11] G. GALES. *Mise en correspondance de pixels pour la stéréovision binoculaire par propagation d’appariements de points d’intérêt et sondage de régions*. Thèse de doctorat, Université Paul Sabatier – Toulouse III, 2011.
- [Gauglitz 11] S. GAUGLITZ, T. HÖLLERER et M. TURK. *Evaluation of interest point detectors and feature descriptors for visual tracking*. *International Journal on Computer Vision, IJCV*, 94(3):335–360, 2011.
- [Gonzalez 04] R. C. GONZALEZ, R. E. WOODS et S. L. EDDINS. *Digital image processing using matlab*. Pierson Prentice Hall, 2004.
- [Harris 88] C. HARRIS et M. STEPHENS. *A Combined Corner and Edge Detector*. Dans Alvey Vision Conference, pages 147–151, Manchester, Royaume-Uni, janvier 1988.
- [Hartley 04] R. HARTLEY et A. ZISSERMAN. *Multiple view geometry in computer vision*. Cambridge University Press, 2004.
- [Harzallah 11] H. HARZALLAH. *Contribution à la détection et à la reconnaissance d’objets dans les images*. Thèse de doctorat, Université Grenoble Alpes, 2011.
- [Hirschmüller 08] H. HIRSCHMÜLLER. *Stereo processing by semiglobal matching and mutual information*. *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI*, 30(2):328–341, 2008.
- [Hong 04] L. HONG et G. CHEN. *Segment-based Stereo Matching Using Graph Cuts*. Dans IEEE Conference on Computer Vision Pattern Recognition, volume 1, pages 74–81, 2004.
- [Hosni 13] A. HOSNI, M. BLEYER et M. GELAUTZ. *Secrets of adaptive support weight techniques for local stereo matching*. *Computer Vision and Image Understanding, CVIU*, 117(6):620–632, 2013.
- [Hu 15] Y. HU, S. GAO, K. JANOWICZ, B. YU, W. LI et S. PRASAD. *Extracting and understanding urban areas of interest using geotagged photos*. *Computers, Environment and Urban Systems*, 54:240–254, 2015.
- [Jolion 03] J. M. JOLION. *Stochastic Pyramid Revisited*. *Pattern Recognition Letters, PRL*, 24(8):1335–1342, mai 2003.
- [Kadir 04] T. KADIR, A. ZISSERMAN et M. BRADY. *An Affine Invariant Salient Region Detector*. Dans European Conference on Computer Vision, ECCV, pages 228–241, 2004.
- [Kitchen 82] L. KITCHEN et A. ROSENFELD. *Gray level corner detection*. *Pattern Recognition Letters, PRL*, 1(2):95–102, 1982.
- [Klaus 06] A. KLAUS, M. SORMANN et K. KARNER. *Segment-Based Stereo Matching Using Belief Propagation and a Self-Adapting Dissimilarity Measure*. Dans International Conference on Pattern Recognition, volume 3, pages 15–18, 2006.

- [Leutenegger 11] S. LEUTENEGGER, M. CHLI et R. Y. SIEGWART. *BRISK: Binary robust invariant scalable keypoints*. Dans International Conference on Computer Vision, ICCV, pages 2548–2555, 2011.
- [Lin 03] M. H. LIN et C. TOMASI. *Surfaces with Occlusions from layered Stereo*. *IEEE Conference on Computer Vision Pattern Recognition*, 2003.
- [Lowe 04] D. G. LOWE. *Distinctive image features from scale-invariant keypoints*. *International Journal of Computer Vision, IJCV*, 60(2):91–110, novembre 2004.
- [Lucas 81] B. D. LUCAS et T. KANADE. *An Iterative Image Registration Technique with an Application to Stereo Vision*. Dans International Joint Conference on Artificial Intelligence, pages 674–679, Vancouver, Canada, août 1981.
- [Malon 18] T. MALON, P. GUYOT, G. ROMAN-JIMENEZ, S. CHAMBON, V. CHARVILLAT, A. CROUZIL, A. PÉNINOU, J. PINQUIER, F. SÈDES et C. SÉNAC. *Toulouse campus surveillance dataset: scenarios, soundtracks, synchronized videos with overlapping and disjoint views*. Dans ACM Multimedia Systems Conference, MMSys, pages 393–398, 2018.
- [Matas 02] J. MATAS, O. CHUM, U. MARTIN et T. PAJDLA. *Robust wide baseline stereo from maximally stable extremal regions*. Dans British Machine Vision Conference, pages 384–393, 2002.
- [Mikolajczyk 04] K. MIKOLAJCZYK et C. SCHMID. *Scale & Affine Invariant Interest Point Detectors*. *International Journal of Computer Vision*, 60(1):63–86, 2004.
- [Mokhtarian 98] F. MOKHTARIAN et R. SUOMELA. *Robust Image Corner Detection Through Curvature Scale Space*. *PAMI*, 20:1376–1381, 1998.
- [Montanvert 91] A. MONTANVERT, P. MEER et A. ROSENFELD. *Hierarchical Image Analysis Using Irregular Tessellations*. *PAMI*, 13(4):307–316, avril 1991.
- [Moravec 80] H. MORAVEC. *Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover*. Thèse de doctorat, Université de Carnegie Mellon, CMU, Pittsburgh, États-Unis, septembre 1980.
- [Muresan 15] M. P. MURESAN, M. NEGRU et S. NEDEVSCHI. *Improving local stereo algorithms using binary shifted windows, fusion and smoothness constraint*. Dans IEEE International Conference on Intelligent Computer Communication and Processing, ICCP, pages 179–185, 2015.
- [Ojala 02] T. OJALA, M. PIETIKAINEN et T. MAENPAA. *Multiresolution gray-scale and rotation invariant texture classification with local binary patterns*. *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI*, 24(7):971–987, 2002.
- [Rashwan 16] H. A. RASHWAN, S. CHAMBON, P. GURDJOS, G. MORIN et V. CHARVILLAT. *Towards multi-scale feature detection repeatable over intensity and depth images*. Dans International Conference on Image Processing, ICIP, pages 36–40, 2016.
- [Rashwan 17] H. A. RASHWAN, S. CHAMBON, P. GURDJOS, G. MORIN et V. CHARVILLAT. *Towards Recognizing of 3D Models Using A Single Image*. Dans Eurographics Workshop on 3D Object Retrieval, 3DOR, 2017.
- [Rosten 06] E. ROSTEN et T. DRUMMOND. *Machine Learning for High-Speed Corner Detection*. Dans European Conference on Computer Vision, ECCV, pages 2564–2571, 2006.
- [Rublee 11] E. RUBLEE, V. RABAUD, K. KONOLIGE et G. BRADSKI. *ORB: an efficient alternative to SIFT or SURF*. Dans International Conference on Computer Vision, ICCV, pages 2564–2571, 2011.
- [Scharstein 02a] D. SCHARSTEIN et R. SZELISKI. *A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms*. *International Journal of Computer Vision, IJCV*, 47(1):7–42, avril 2002.

- [Scharstein 02b] D. SCHARSTEIN et R. SZELISKI. *A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms*. *IJCV*, 47(1):7–42, avril 2002.
- [Shi 93] J. SHI et C. TOMASI. Good features to track. Rapport de recherche TR 93-1399, Cornell University, novembre 1993.
- [Shi 94] J. SHI et C. TOMASI. *Good Features to Track*. Dans IEEE Conference on Computer Vision and Pattern Recognition, pages 593–600, Seattle, États-Unis, juin 1994.
- [Shum 98] H.-Y. SHUM et R. SZELISKI. *Construction and refinement of panoramic mosaics with global and local alignment*. Dans International Conference on Computer Vision, ICCV, pages 953–956, 1998.
- [Smith 97] S. SMITH et J. BRADY. *SUSAN - A New Approach to Low Level Image Processing*. *International Journal of Computer Vision, IJCV*, 23(1):45–78, 1997.
- [Szeliski 10] R. SZELISKI. Computer vision: Algorithms and applications. Springer, 2010. <http://szeliski.org/Book/>.
- [Tomasi 91] C. TOMASI et T. KANADE. Detection and tracking of point features. Rapport de recherche CMU-CS-91-132, Carnegie Mellon University, avril 1991.
- [Trajković 98] Miroslav TRAJKOVIĆ et Mark HEDLEY. *Fast corner detection*. *Image and vision computing*, 16(2):75–87, 1998.
- [Tuytelaars 04] T. TUYTELAARS et L. VAN GOOL. *Matching Widely Separated Views Based on Affine Invariant Regions*. *International Journal of Computer Vision*, 59:61–85, 2004.
- [Tuytelaars 06] T. TUYTELAARS. Local Invariant Features: What? Why? When? How? Tutorial in European Conference on Computer Vision, ECCV, 2006.
- [Tuzel 06] O. TUZEL, F. PORIKLI et P. MEER. *Region covariance: A fast descriptor for detection and classification*. Dans European Conference on Computer Vision, ECCV, pages 589–600, 2006.
- [Veksler 02] O. VEKSLER. *Stereo correspondence with compact windows via minimum ratio cycle*. *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI*, 24(12):1654–1660, 2002.
- [Wang 08] Z. F. WANG et Z. G. ZHENG. *A region based stereo matching algorithm using cooperative optimization*. Dans IEEE Conference on Computer Vision Pattern Recognition, 2008.
- [Wang 10] J. WANG, J. LI, W. YAU et E. SUNG. *Boosting dense SIFT descriptors and shape contexts of face images for gender recognition*. Dans IEEE Conference on Computer Vision Pattern Recognition Workshop, CVPR Workshop, pages 96–102, 2010.
- [Yang 09] Q. YANG, L. WANG, R. YANG, H. STEWÉNIUS et D. NISTÉR. *Stereo matching with color-weighted correlation, hierarchical belief propagation and occlusion handling*. *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI*, 31(3):492–504, 2009.
- [Yoon 06] K.-J. YOON et I. S. KWEON. *Adaptive support-weight approach for correspondence search*. *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI*, 28(4):650–656, 2006.
- [Yu 11] G. YU et J.-M. MOREL. *ASIFT: An Algorithm for Fully Affine Invariant Comparison*. *Image Processing On Line*, 2011.
- [Zhang 09] K. ZHANG, J. LU et G. LAFRUIT. *Cross-Based Local Stereo Matching Using Orthogonal Integral Images*. *IEEE Transactions on Circuits and Systems for Video Technology, CSVT*, 19(7):1073–1079, 2009.