Université de Montréal

Mise en correspondance active et passive pour la vision par ordinateur multivue

par

Marc-Antoine Drouin

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Thèse présentée à la Faculté des études supérieures
en vue de l'obtention du grade de
Philosophiae Doctor (Ph.D.)
en informatique

Mars, 2007

Université de Montréal

Faculté des études supérieures

Cette thèse intitulée :

Mise en correspondance active et passive pour la vision par ordinateur multivue

présentée par

Marc-Antoine Drouin

a été évaluée par un jury composé des personnes suivantes:

Pierre Poulin
président-rapporteur

Sébastien Roy
directeur de recherche

Max Mignotte
membre du jury

Patrick Hébert
examinateur externe

Jocelyn Faubert
représentant du doyen de la FES

Thèse acceptée le 21 mars 2007

RÉSUMÉ

Cette thèse explore la mise en correspondance entre vues multiples, une branche importante de la vision par ordinateur. Une vue peut être un appareil passif comme une caméra ou un système actif tel qu'un projecteur multimédia. L'objectif est de trouver les pixels appartenant à différentes vues qui correspondent au même point 3D. Plusieurs nouvelles méthodes actives et passives utilisant des minimisations d'énergie seront présentées. La thèse est séparée en deux parties. Dans la première, trois nouveaux algorithmes passifs permettant d'effectuer la mise en correspondance seront présentés. Ces derniers sont spécialement conçus pour palier aux problèmes des occultations présents en stéréoscopie à images multiples. Le premier de ces algorithmes détecte les occultations dans les cartes de profondeurs qui sont obtenues par des algorithmes standards de stéréoscopie. Les occultations sont détectées comme des anomalies obtenues en reprojetant la carte de profondeurs dans chacune des images. Les anomalies détectées permettent de modifier la fonction d'énergie en retirant les caméras non visibles du calcul de la fonction de coûts. L'algorithme modifie graduellement la fonction de coûts en utilisant une historique des anomalies. Cet algorithme est itératif et la convergence est garantie. Une extension de cet algorithme permettant une décomposition non-uniforme de la carte de profondeurs est également présentée. Cette dernière permet de combiner le calcul de la visibilité et le raffinement de la carte de profondeurs. Nous utiliserons deux algorithmes stéréoscopiques basés sur des méthodes de flot maximal dans les graphes. Notre méthode peut avoir recours à d'autres algorithmes de stéréoscopie. Le second algorithme est un hybride qui se situe entre les méthodes heuristiques rapides et les méthodes qui préservent un traitement exact de la visibilité. Cette approche est basée sur la programmation dynamique itérative

et calcule simultanément la disparité ainsi que la visibilité. La programmation dynamique permet de calculer de manière exacte une partie de l'information de la visibilité. Le reste de l'information est obtenu à l'aide de méthodes heuristiques. Finalement, nous présenterons un algorithme qui permet d'améliorer significativement la localisation des bordures des cartes de disparités obtenue par les algorithmes multi-caméras. Au lieu d'associer une disparité à chacun des pixels, celui-ci associe une position à chacune des discontinuités de profondeurs. Cette formulation permet d'obtenir une solution approximative à un problème d'étiquetage 2D possédant un terme de lissage non convexe. Elle est obtenue en effectuant plusieurs minimisations 1D à l'aide de la programmation dynamique. La validation des méthodes que nous proposons est établie en utilisant de vraies images possédant des solutions de référence. Nos résultats se comparent très favorablement avec les méthodes de pointe déjà existantes. Dans la seconde partie, une méthode de mise en correspondance active est introduite. Nous proposons un cadre général permettant de tenir compte des déplacements projectifs provenant de la configuration de la caméra, du projecteur et de la scène. Notre formulation permet un calcul très rapide lorsqu'un processeur graphique programmable est utilisé. Nous démontrons expérimentalement que notre approche permet d'obtenir des améliorations significatives lorsque les conditions d'acquisition sont difficiles. Notre méthode a également permis d'effectuer le calibrage de projecteurs et nos résultats ont été validés en utilisant un appareil de suivi au laser.

Mots clés : stéréoscopie, occultation, lumière structurée, minimisation d'énergie, mise en correspondance, multivue, vision par ordinateur

ABSTRACT

This thesis explores an important aspect of computer vision named multi-view matching. A view can either be a passive device such as a camera or an active one such as a multimedia projector. Our goal is to find the pixels in different views that correspond to the same 3D point of a scene. Many new energy-based methods for active and passive multi-view matching are presented. The thesis is composed of two main parts. Whilst the first one presents novel passive algorithms, the second describes a new algorithm for active matching. In the first part, three new passive multi-camera matching algorithms are introduced. These are designed to overcome the occlusion problems of multi-baseline stereo. The first one detects occlusions in the depth map obtained from regular efficient stereo matching algorithms. Occlusions are detected as inconsistencies of the depth map by computing the visibility of the map as it is reprojected into each camera. The matching cost function is modified according to the detected occlusions by removing the offending cameras from the computation of the matching cost. The algorithm gradually modifies the matching cost function according to the history of inconsistencies in the depth map, until convergence. We also provide an extension to this framework that features a non-uniform spatial decomposition of the disparity map. This extension allows the integration of the visibility computation and refinement of the disparity into a single iterative framework. While two graph-theoretic stereo algorithms are used in our experiments, our framework is general enough to be applied to many others. The second algorithm is a hybrid between fast heuristic occlusion overcoming algorithms that precompute an approximate visibility and slower methods that use correct visibility handling. Our approach is based on iterative dynamic programming and computes simultaneously

disparity and camera visibility. Interestingly, dynamic programming makes it possible to compute exactly part of the visibility information. The remainder is obtained through heuristics. Finally, we also propose an algorithm that improves the localization of disparity discontinuities of disparity maps obtained by multi-baseline stereo matcher. Rather than associating a disparity label to every pixel of a disparity map, it associates a position to every disparity discontinuity. This formulation allows us to find an approximate solution to a 2D labeling problem with robust smoothing term by minimizing multiple 1D problems, thus making possible the use of dynamic programming. The validity of the proposed novel algorithms is established using real imagery with ground truth and they compare favorably with other state-of-the-art algorithms. In the second part of the thesis, an active matching algorithm is introduced. It is a new real-time energy-based formulation of the matching problem encountered when establishing the correspondence using a stripe-based structured light system. We provide a general framework that takes into consideration the projective displacement which is induced by the camera, projector and scene configuration. This allows the selection of a neighborhood for the smoothing term which is geometrically plausible. The minimization uses a reformulation of the dynamic programming recurrences and is performed on a programmable graphic processor. We show experimentally that our approach provides major improvements under difficult conditions. Our method is also used for projector calibration: we validate the results with a laser tracker.

Keywords: stereo, occlusion, structured light, energy minimization, matching, multi-view, computer vision

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLES

REMERCIEMENTS

De nombreux remerciements s'adressent à mon directeur de recherche, Sébastien Roy, pour sa contribution à ma formation doctorale et pour son soutien à des publications associées aux travaux réalisés.

Je tiens à mentionner l'appui constant de Martin Trudeau et à le remercier, car sans lui, plusieurs de mes projets ne se seraient peût-être jamais concrétisés.

D'autres personnes et institutions méritent mes remerciements pour leur collaboration, leur accueil et leur aide toute particulière. Je nomme mes collègues du laboration de vision 3D de l'Université de Montréal, les membres du groupe de technologie de l'information visuelle (TIV) du Conseil national de recherches Canada (CNRC), Guy Godin pour avoir rendu possible ma présence à titre de chercheur invité dans ce centre de recherche, Luc Cournoyer et Michel Picard pour l'assemblage du banc de calibrage utilisé dans mes travaux et Luc Cournoyer qui m'a fourni plusieurs capsules de formation portant sur l'usage d'appareils de métrologie 3D.

Je remercie le Fonds québécois de la recherche sur la nature et les technologies (FQRNT) qui m'a permis de me consacrer à temps plein à l'élaboration de cette thèse.

Toute ma reconnaissance et ma gratitude à mes parents, Lucille Michaud et Pierre Drouin pour leur encouragement et leur soutien depuis toujours. L'implication de mes parents et de mon entourage immédiat a certainement contribué à maintenir ma motivation à poursuivre des études supérieures et à les réusir.

Je dois beaucoup à ma conjointe Sonya Banal pour sa compréhension, sa complicité, sa patience et son appui inconditionnel au cours de ces dernières années.

Chapitre 1

INTRODUCTION

Après plusieurs décennies de recherche, la vision par ordinateur est devenue une discipline mature comportant un grand nombre de champs de recherche et d'applications dont les problèmes complexes demeurent non résolus. Parmi les différents champs de recherche figurent : la détection de points d'intérêt, la segmentation, la reconnaissance de forme, l'analyse de cartes de profondeurs, la reconstruction tridimensionnelle, la vision active, la détection et suivi de mouvement, le calcul de trajectoire de caméras, la géométrie multi-vues et autres... Les différents champs d'applications s'intéressent à la télé-surveillance, à la bio-technologie, à l'inspection industrielle, à la réalité virtuelle, à l'exploration de l'espace, à la robotique, aux effets spéciaux cinématographiques et autres... Les travaux effectués dans le cadre de cette thèse portent sur la mise en correspondance entre des vues multiples. Nous utiliserons le terme "vue" de manière générique pour désigner une image prise d'un certain point de vue. Une vue peut représenter autant une caméra qu'un projecteur. Alors qu'une caméra perçoit la scène, le projecteur peut modifier la scène en y apposant des motifs lumineux d'intensités et de couleurs variables. Dans les deux cas, l'interaction du système avec la scène peut être modélisée à l'aide d'une branche de la géométrie projective appliquée à la vision par ordinateur. Celle-ci se nomme la géométrie multi-vues. La mise en correspondance entre deux vues de la même scène consiste à déterminer les paires de points correspondant aux mêmes points 3D de la scène. Les points peuvent être des pixels ou des points d'intérêt extraits des images associées aux vues. Généralement, on choisit une vue de référence et pour chacun des points de cette vue, on

essaie d'associer un point provenant de la seconde. Cette dernière se nomme vue de support. Lorsqu'on procède à la généralisation à plusieurs vues, chacun des points de la vue de référence est associé à un point dans chacune des vues de support. Peu importe le nombre de vues, on a toujours une seule vue de référence.

La mise en correspondance peut être effectuée à partir de vues calibrées et non-calibrées. Il faut trouver le point de support correspondant à un point de référence. Ceci nécessite une recherche dans toute l'image de support lorsqu'un système multi-vues n'est pas calibré. Lorsque le système est calibré, la recherche est réduite à une droite. Cette réduction de l'espace de recherche est rendue possible grâce à la contrainte épipolaire qui sera présentée dans le chapitre 2.

La mise en correspondance peut être de type épars ou dense. Dans le premier cas, les correspondances sont calculées uniquement pour certains pixels choisis. On débute généralement par l'extraction des points d'intérêt des images. Il peut s'agir de jonctions entre les arêtes présentes dans les images. Un point d'intérêt peut englober plusieurs pixels et posséder une coordonnée fractionnaire. Par la suite, on définit une métrique afin d'établir la distance entre deux points d'intérêt. La métrique peut dépendre de la couleur, de l'intensité, de la géométrie, du calibrage et autres. Le problème peut, par la suite, être résolu à l'aide d'une minimisation discrète ou continue. Les régions peu texturées ne contiennent pas de points d'intérêt. Ces dernières sont généralement sujettes aux erreurs et affectent uniquement la mise en correspondance dense. La mise en correspondance ponctuelle est généralement moins ambiguë. Lorsqu'on effectue de la mise en correspondance dense, les correspondances sont établies pour tous les pixels de l'image de référence. Dans ce cas, il faut choisir une métrique permettant de comparer les pixels. Cette métrique ne dépend pas nécessairement uniquement des pixels comparés, elle peut tenir compte de leur voisinage et utiliser des points d'intérêt. Le problème de la mise en correspondance dense est généralement formulé de manière discrète. Il est alors résolu en utilisant des techniques de minimisation d'énergie discrète. La présente thèse introduira de nouvelles méthodes de mise

en correspondance dense.

Lorsqu'un système effectuant une mise en correspondance utilise uniquement des caméras, il appartient à la classe des systèmes passifs. Un système qui utilise au moins un projecteur appartient à la classe des systèmes actifs. Des résultats nouveaux seront présentés autant en mise en correspondance active que passive. Pour la mise en correspondance passive, nous nous intéresserons au problème de la gestion des occultations en stéréoscopie calibrée à images multiples. Nous présenterons de nouvelles formulations discrètes du problème. Nous utiliserons des algorithmes classiques de minimisation discrète ainsi que de nouveaux algorithmes spécialement conçus pour les problèmes associés à nos formulations. Pour la mise en correspondance active, les systèmes utilisant simultanément des projecteurs et caméras seront explorés. Ces derniers sont usuellement appelés systèmes de reconstruction par lumière structurée. Pour ce type de mise en correspondance, nous proposons des algorithmes qui s'appliquent autant aux systèmes calibrés que non-calibrés. L'algorithme permettant d'établir la mise en correspondance actif fonctionne en temps réel lorsqu'il est implanté dans un processeur graphique programmable. Par contre, les algorithmes passifs ne fonctionnent pas en temps réel.

1.1 Organisation de la thèse

Dans un premier temps, nous présenterons les concepts géométriques nécessaires à la compréhension des chapitres suivants. Nous présenterons par la suite une introduction aux méthodes de minimisation combinatoire discrète. La thèse sera ensuite séparée en 2 parties :

- gestion des occultations lors de la mise en correspondance stéréoscopique à images multiples
- mise en correspondance par lumière structurée.

Titre	conférence	type	taux d'acceptation
Passifs			
Geo-consistence for wide baseline	CVPR	affiche	28%
Fast Multiple-baseline with Occlusion	3DIM	oral	31%
Improving Border Localization of Multi-Baseline Stereo Using Border-Cut	CVPR	oral	4.8%
Non-Uniform Hierarchical Geo-consistency for for Multi-baseline Stereo	CRV	oral	-
Actifs			
Real-Time Geometrically Plausible Energy-Based Structured Light Matching	soumis	-	-

TAB. 1.1 – Liste des publications associées à la thèse.
CVPR : IEEE Computer Society Conference on Computer Vision and Pattern Recognition
3DIM : International Conference on 3-D Digital Imaging and Modeling
CRV : Fourth Canadian Conference on Computer and Robot Vision

Chacune de ces parties contiendra un chapitre d'introduction suivi de chapitres présentant les contributions de cette thèse. Le tableau 1.1 contient la liste des publications associées aux travaux réalisés. Les travaux portant sur la mise en correspondance active ont été effectués à titre de chercheur invité dans le groupe de technologie de l'information visuelle (TIV) de l'Institut de technologie de l'information (ITI) du Conseil national de recherches Canada (CNRC).

1.1.1 Contribution des coauteurs

Dans cette sous-section, nous présenterons la contribution des coauteurs à chacune des publications associées aux travaux réalisés.

Geo-consistency for Wide Multi-Camera Stereo

Martin Trudeau a proposé la méthode de post-traitement présentée à la fin de la section 5.5. Il a également participé activement à la rédaction, à la révision de l'article et au choix de la notation utilisée. Sébastien Roy a supervisé le travail de recherche,

a contribué à la définition du problème et a proposé la notation. Il a participé à la révision de l'article et à la rédaction de la réponse expédiée aux arbitres.

Fast Multiple-baseline Stereo with Occlusion

Martin Trudeau a participé à la rédaction et à la révision de l'article. Il a participé activement au choix de la notation utilisée. Sébastien Roy a participé à la relecture de l'article.

Improving Border Localization of Multi-Baseline Stereo Using Border-Cut

Martin Trudeau a participé à la rédaction et à la révision de l'article. Il a participé activement au choix de la notation utilisée. Sébastien Roy a participé à la relecture de l'article et à la rédaction de la réponse expédiée aux arbitres.

Real-Time Geometrically Plausible Energy-Based Structured Light Matching

Guy Godin a supervisé le travail de recherche. Il a contribué à la définition du problème et à la préparation de la procédure expérimentale. Il a participé à la rédaction et à la révision de l'article. Sébastien Roy a participé à la rédaction de l'article et s'est assuré que les activités de recherche ne chevauchent pas celles effectuées par les autres étudiants du laboratoire.

Non-Uniform Hierarchical Geo-consistency for Multi-baseline Stereo

Martin Trudeau et Sébastien Roy ont participé à la révision de l'article.

Chapitre 2

LES PRÉLIMINAIRES GÉOMÉTRIQUES

Le présent chapitre présente les concepts géométriques qui sont nécessaires à la compréhension des chapitres suivants. Nous débuterons par une présentation du modèle de caméra utilisé. Par la suite, nous présenterons la transformation projective 2D ainsi qu'une méthode de calibrage de caméra adaptée à ce type de transformation. Nous terminerons par une énumération commentée des outils et des références qui pourront être consultés par le lecteur afin d'approfondir les notions présentées dans ce chapitre. Il est important de noter que les résultats obtenus dans ce chapitre sont également applicables lorsqu'un projecteur remplace une caméra.

2.1 Caméra sténopée

La caméra sténopée est le type de caméra le plus simple à modéliser mathématiquement et est également très simple à construire. Elle peut être assemblée à partir d'une boîte de carton dans laquelle on perce une petite ouverture sur un côté et sur le côté opposé un papier photosensible est appliqué. L'ouverture et le papier photosensible correspondent respectivement au centre de projection et au plan de l'image. La distance entre les deux côtés de la boîte correspond à la distance focale de la caméra. Une caméra sténopée est illustrée à la figure 2.1. Dans le modèle mathématique, le centre de projection de la caméra est à l'origine du système de coordonnées de la caméra. L'axe optique de la caméra se confond à l'axe des Z du système de coordonnées de la caméra et les axes des X et des Y sont alignés avec le système d'axes de l'image. L'axe optique est perpendiculaire au plan de l'image. Une caractéristique de la caméra sténopée est de permettre que tous les objets de la scène soient nets peu

importe la distance de ceux-ci. On peut définir la matrice K de passage d'un point 3D exprimé en coordonnées homogènes dans le système de coordonnées de la caméra vers celui de l'image comme étant

$$K = \begin{pmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}. \tag{2.1}$$

Le centre de l'image est (c_x, c_y) et f_x et f_y sont les distances focales des axes X et Y exprimées en pixels. Le cisaillement du plan image est dénoté par s. Les différents paramètres c_x,c_y,f_x,f_y et s sont les paramètres internes de la caméra. Le cisaillement s est de zéro avec les caméras numériques. Plusieurs caméras numériques modernes ont des pixels carrés et $f_x = f_y$. Le nombre de paramètres peut donc varier entre 3 et 5.

Puisque le système de coordonnées de la caméra ne correspond pas toujours au système de coordonnées du monde, une matrice G de transformation rigide entre le système de coordonnées du monde et celui de l'image de la caméra est définie comme étant

$$G = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{pmatrix}.$$

La matrice est composée d'une sous-matrice de rotation $\mathbf{R}$ et d'un vecteur de translation $\mathbf{T}$ qui représentent les paramètres externes de la caméra (position et orientation de la caméra dans le monde). Afin d'être plus concis, on écrit $G = (\mathbf{r_1}, \mathbf{r_2}, \mathbf{r_3}, \mathbf{T})$. Chacun des éléments est un vecteur colonne. Nous utiliserons également la notation $G = (\mathbf{R}|\mathbf{T})$. Le calibrage de caméra consiste à retrouver les paramètres internes et externes d'une caméra ou d'un projecteur. Un point 3D (X, Y, Z, W) du monde exprimé en coordonnées homogènes correspond au pixel $(u/t, v/t)$ lorsque $(u, v, t) = P\,(X, Y, Z, W)$ et $P = K\,G$. À partir de la matrice P, il est possible de

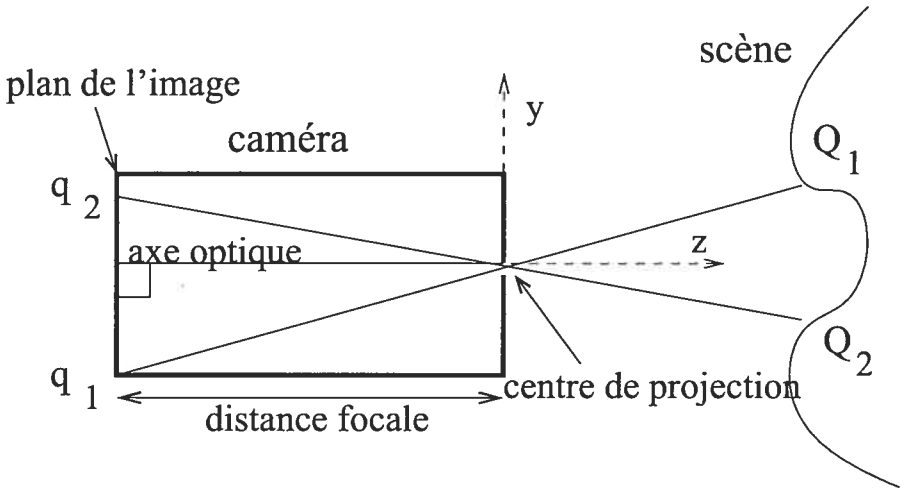

FIG. 2.1 – Caméra sténopée. Les points 3D Q_1 et Q_2 se projettent respectivement sur les points q_1 et q_2 du plan image.

retrouver la position du centre de projection de la caméra dans le monde. Le centre de projection est le vecteur $\mathbf{X}$ tel que $P\mathbf{X} = \mathbf{0}$.

Lorsqu'on travaille simultanément avec plusieurs caméras, on fixe généralement le système de coordonnées du monde pour qu'il corresponde au système de coordonnées d'une des caméras. Celle-ci se nomme caméra de référence. La matrice de rotation de cette dernière est la matrice identité I et le vecteur de translation $\mathbf{T}$ est égal à zéro.

2.1.1 Distorsion radiale

Les lentilles utilisées dans les caméras introduisent de la distorsion qui n'est pas modélisable à l'aide d'équations linéaires. La source la plus importante d'erreurs est probablement la distorsion radiale. Généralement, plus la distance focale est courte, plus les lentilles souffrent de distorsion radiale. Le prix des lentilles semble également être un facteur influençant la quantité de distorsion. Un point (X, Y, Z) dans le système de coordonnées de la caméra est projeté en un point $(x, y) = (X/Z, Y/Z)$. Ce point est par la suite *distorsionné* en un autre point (x', y') par la fonction de

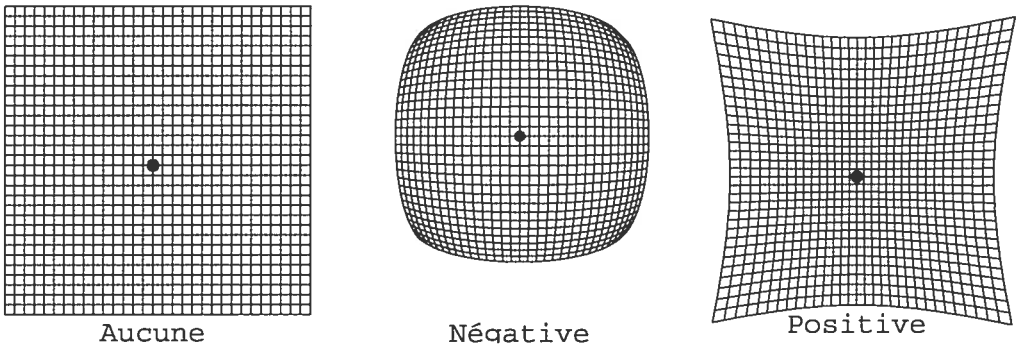

FIG. 2.2 – Illustration de l'effet de la distorsion radiale. Le centre de distorsion radiale est représenté par un point gris.

distorsion radiale

$$\begin{aligned} x' &= L\left(\sqrt{(x-x_c)^2+(y-y_c)^2}\right)(x-cx)+cx \\ y' &= L\left(\sqrt{(x-c_c)^2+(y-y_c)^2}\right)(y-cy)+cy. \end{aligned}$$

Le centre de distorsion radiale est (x_c, y_c) et L est la fonction d'amplitude de la distorsion radiale. La fonction L utilisée est de forme

$$L(r) = \kappa_1 r^3 + \kappa_2 r^5 + \kappa_3 r^7. \tag{2.2}$$

Cette fonction est l'amplitude du déplacement le long de la droite définie par les points (x, y) et (c_x, c_y). Dépendant des coefficients κ, la fonction L peut être positive ou négative. La figure 2.2 illustre les effets de la distorsion radiale. À partir du point distorsionné (x', y'), la coordonnée du pixel est $(f_x x' + sy' + c_x, f_y y' + c_y)$. Des erreurs d'assemblage des lentilles peuvent introduire un autre type de distorsion nommé distorsion tangentielle. Un terme peut être ajouté à la fonction L pour modéliser cette dernière. La distorsion radiale est une transformation non-linéaire du plan image. Connaissant la position du centre de déformation radiale ainsi que les différents coefficients κ il est possible d'appliquer une transformation à l'image afin que le modèle

sténopé puisse par la suite être utilisé. Dans nos expériences, nous supposons que la distorsion radiale est négligeable ou qu'elle a été corrigée.

2.2 *Géométrie épipolaire*

Ayant une paire d'images stéréo et un point 3D, $\mathbf{Q}$, on peut définir un plan passant par les centres de projection des deux caméras ainsi que par le point $\mathbf{Q}$. Ce plan se nomme le plan épipolaire (voir figure 2.3). La projection dans une caméra du centre de projection de l'autre caméra se nomme épipôle. Celui-ci fait partie de tous les plans épipolaires. Les lignes créées par l'intersection du plan épipolaire avec les plans des images se nomment lignes épipolaires. Un point d'une image se trouve sur une seule ligne épipolaire. Cependant il existe une exception c'est-à-dire toutes les lignes épipolaires passent par l'épipôle.

De manière formelle lorsque la première caméra est celle de référence,

$$\lambda_1 K_1^{-1}\mathbf{q}_1 = (\lambda_2 R K_2^{-1}\mathbf{q}_2 + T).$$

La distance entre le point 3D se projetant en $\mathbf{q}_1$ et $\mathbf{q}_2$ et les caméras 1 et 2 sont λ_1 et λ_2 respectivement. Les points $\mathbf{q}_1$ et $\mathbf{q}_2$ sont en coordonnées homogènes. En se servant du fait que $[\mathbf{T}]_\times \mathbf{T} = 0$* on a que

$$\lambda_1 [T]_\times K_1^{-1}\mathbf{q}_1 = \lambda_2 [T]_\times R K_2^{-1}\mathbf{q}_2.$$

Puisque $K_1^{-1}\mathbf{q}_1$ est perpendiculaire à $[T]_\times K_1^{-1}\mathbf{q}_1$ on a

$$0 = \mathbf{q}_1^T K_1^{-T} [T]_\times R K_2^{-1}\mathbf{q}_2.$$

---

* $[\mathbf{a}]_\times$ est une matrice anti-symétrique définie comme étant

$$[(a_1, a_2, a_3)]_\times = \begin{pmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{pmatrix}.$$

Cette matrice permet de représenter le produit vectoriel sous forme matricielle ($[\mathbf{a}]_\times \mathbf{b} = \mathbf{a} \times \mathbf{b}$).

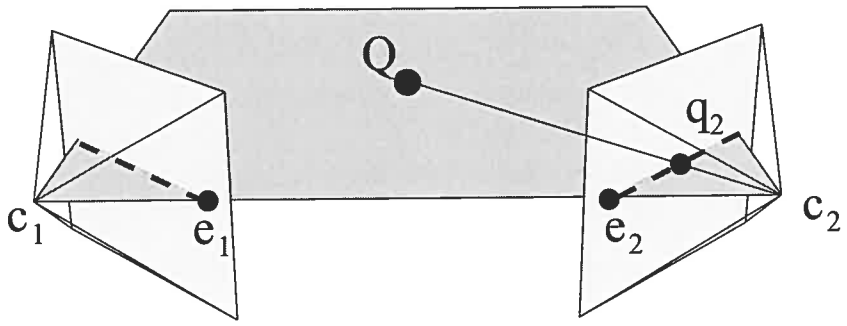

FIG. 2.3 – Illustraction de la contrainte épipolaire. c_1,c_2 et Q sont respectivement les centres de projection des deux caméras ainsi que le point 3D permettant de définir un plan épipolaire. les épipôles sont e_1 et e_2. Les deux lignes épipolaires induites par la projection du plan épipolaire sont illustrées en pointillés.

Cette dernière équation est représentée par

$$\mathbf{q}_1^T F \mathbf{q}_2 = 0. \tag{2.3}$$

La matrice F se nomme matrice fondamentale. Elle est de dimension 3×3 et est de rang 2 [†]. Notez qu'il n'est pas nécessaire de connaître les paramètres internes et externes des caméras pour trouver la matrice F. L'algorithme classique est l'algorithme des 8 points. Il s'agit d'un algorithme linéaire qui nécessite uniquement des correspondances entre les deux images sans avoir une connaissance sur les paramètres internes. Chaque paire de correspondance fournit une contrainte linéaire en utilisant l'équation 2.3. Avec 8 points, on peut donc fabriquer un système d'au moins 8 équations permettant de résoudre F. Pour ce faire, il faut 8 correspondances entre les images qui ne correspondent pas à des points coplanaires dans la scène. Lorsque le système est surdéterminé on minimise par moindres carrés en utilisant une décomposition en

[†] $[T]_\times$ est de rang 2.

valeurs singulières et en imposant la contrainte $|F| = 1$. Ceci permet d'éliminer la solution triviale $F = \mathbf{0}$. Par la suite, le rang 2 de la matrice fondamentale est imposé. Pour ce faire, on décompose F à l'aide d'une décomposition en valeurs singulières $F = UDV^T$. On impose le rang 2 de la manière suivante $F' = USDV^T$ avec

$$S = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Il existe un algorithme des 7 points qui exploite le rang 2 de la matrice fondamentale. Il existe également un algorithme des 6 points (4 coplanaires et 2 en configuration générale). L'algorithme des 8 points est généralement suivi d'une minimisation non-linéaire qui minimise un critère d'erreur géométrique.

2.2.1 Normalisation

Il a été démontré expérimentalement qu'il est préférable de normaliser les points avant d'utiliser l'algorithme des 8 points. On normalise généralement de façon à ce que le centre de masse des points soit remis à l'origine et que la distance moyenne à l'origine des points normalisés soit de $\sqrt{2}$. On normalise un point $\mathbf{q}$ en appliquant la matrice

$$H = \begin{pmatrix} s & 0 & -s\mu_x \\ 0 & s & -s\mu_y \\ 0 & 0 & 1 \end{pmatrix}.$$

Le facteur d'échelle est s et (μ_x, μ_y) est le centre de masse des points non normalisés. Si on calcule la matrice fondamentale F_N pour les points normalisés $H_1\mathbf{q}_1$ et $H_2\mathbf{q}_2$, la matrice fondamentale non normalisée est $H_1^T F_N H_2$.

2.2.2 Contrainte épipolaire

Lorsque la matrice fondamentale entre 2 caméras est connue, il est possible de réduire le nombre de dimensions du problème de la mise en correspondance de 2D à

1D grâce à la contrainte épipolaire. Pour un pixel de la vue de référence correspondant à un objet de la scène, la contrainte épipolaire permet de limiter la recherche du pixel de support correspondant au pixel de référence le long d'une ligne. Cette dernière se nomme ligne épipolaire. Par exemple dans la figure 2.3, le point dans l'image 1 correspondant au point $\mathbf{q}_2$ de l'image 2 se trouve nécessairement sur la ligne épipolaire qui est en pointillée. Si la matrice fondamentale est connue, la ligne épipolaire dans la première image correspondant au point $\mathbf{q}_2$ est $F\mathbf{q}_2$. Lorsque deux vues ont les mêmes paramètres internes et que la transformation rigide entre les deux vues est uniquement translationnelle, la matrice fondamentale est de forme

$$F = \begin{pmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{pmatrix}. \tag{2.4}$$

Dans les sections touchant la reconstruction passive, nous travaillerons avec des configurations de caméras possèdant des matrices fondamentales de cette forme avec $c = 0$. Les données utilisées ont été obtenues à l'aide d'une caméra montée sur une table de déplacement ou sur un bras robotisé.

Lorsque la matrice fondamentale ne respecte pas la forme de l'équation 2.4, il est possible d'appliquer une transformation à chacune des images. La procédure est très similaire à la normalisation décrite à la section 2.2.1 sauf que les matrices H_1 et H_2 sont des matrices 3×3 génériques. Ces dernières sont des homographies 2D. Cette transformation de la matrice fondamentale et des images se nomme rectification planaire. Généralement, on veut une matrice avec $c = b = 0$. Dans ce cas, les lignes épipolaires sont horizontales. Puisque les images utilisées dans les expériences effectuées n'ont pas nécessité de rectification plane, nous ne discuterons pas davantage de rectification des images.

2.2.3 Disparité et profondeur

Le point $\mathbf{Q} = (X, Y, Z, 1)$ du monde est projeté dans la première vue au point $\mathbf{q}_1$ et $\mathbf{q}_2$ dans la seconde. Explicitement, lorsque la matrice F est de la forme de l'équation 2.4, $\mathbf{q}_1 = K(I|\mathbf{0})\mathbf{Q}$ et $\mathbf{q}_2 = K(I|\mathbf{T})\mathbf{Q}$. Les points images en coordonnées non homogènes sont reliés par la relation

$$\mathbf{q}_2 = \mathbf{q}_1 + \frac{K'\mathbf{T}}{Z}$$

lorsque $\mathbf{T}$ est de la forme $(a, b, 0)^T$ et K' est la sous-matrice de K contenant uniquement les 2 premières lignes. La disparité du pixel $\mathbf{q}_1$ est $\frac{K'\mathbf{T}}{Z}$ et la disparité du pixel $\mathbf{q}_2$ est $-\frac{K'\mathbf{T}}{Z}$. Notons que nous considérons la disparité comme étant un vecteur. Cette définition diffère de celle qui est normalement utilisée et qui définit la disparité comme étant une distance.

Une propriété intéressante est que la disparité entre deux images rectifiées correspond à la même profondeur Z pour tous les pixels de l'image. Ceci n'est pas vrai dans le cas général où il y a une rotation entre les deux vues.

À partir de la disparité $\mathbf{d}$, il est possible de retrouver Z. Il suffit de résoudre le système d'équations

$$Z\mathbf{d} = K'\mathbf{T}.$$

Si les points $\mathbf{q}_1$ et $\mathbf{q}_1 + \mathbf{d}$ sont sur la même ligne épipolaire, il existe une seule solution. Lorsque cette dernière n'existe pas, on est confronté au problème de triangulation. Nous ne nous attarderons pas davantage sur ce problème.

La détection des occultations est facilitée lorsque les disparités sont utilisées au lieu des profondeurs. Par exemple, un point $\mathbf{q}_1$ dont la disparité est $\mathbf{d}_1$ et un point $\mathbf{q}_1 + \alpha$ dont la disparité est $\mathbf{d}_2$ sont en occultation lorsque

$$\begin{aligned} \mathbf{q}_1 + \alpha + \mathbf{d}_2 &= \mathbf{q}_1 + \mathbf{d}_1 \\ \alpha &= \mathbf{d}_1 - \mathbf{d}_2. \end{aligned}$$

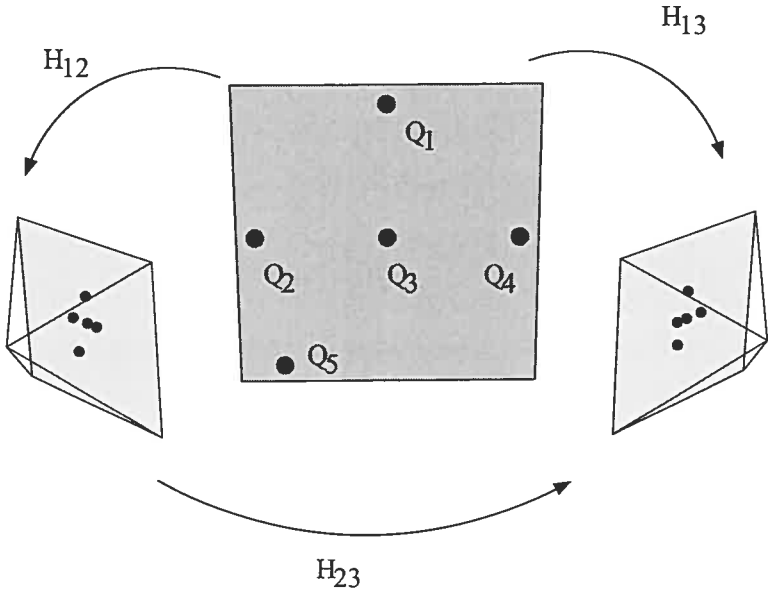

FIG. 2.4 – Application de 3 homographies entre 2 vues et un plan dans la scène.

La gestion des occultations en reconstruction stéréoscopique sera présentée dans le chapitre 4. Nous présenterons 3 algorithmes de reconstruction tri-dimensionnelle capables de gérer les occultations.

2.3 Homographie 2D

Une homographie 2D est une transformation projective qui prend des points sur un plan et les projette sur un second plan (voir figure 2.4). Explicitement,

$$\lambda \mathbf{q}_2 = H\mathbf{q}_1$$

et λ est un facteur d'échelle et $H = (\mathbf{h_1}, \mathbf{h_2}, \mathbf{h_3})$ est une matrice 3×3. Une propriété intéressante des homographies est la possibilité d'en faire la composition. Par exemple dans la figure 2.4, nous observons que $H_{12}H_{23} = \lambda H_{13}$ avec λ qui est encore une fois un facteur d'échelle.

La matrice H possède 8 degrés de liberté (9 entrées moins le facteur d'échelle).

On peut la calculer en utilisant une transformation linéaire directe. On procède de la manière suivante :

$$\begin{aligned} \lambda \mathbf{q}_2 &= H\mathbf{q}_1 \\ \lambda [\mathbf{q}_2]_\times \mathbf{q}_2 &= [\mathbf{q}_2]_\times H\mathbf{q}_1 \\ \mathbf{0} &= [\mathbf{q}_2]_\times H\mathbf{q}_1. \end{aligned}$$

Chacune des paires $(\mathbf{q}_1, \mathbf{q}_2)$ permet d'établir 3 contraintes dont 2 sont indépendantes. Il faut donc 4 points en configuration générale afin de calculer une homographie. Une configuration est générale lorsqu'aucun triplet de points n'est pas colinéaire. Par exemple, les points Q_1, Q_2, Q_3 et Q_4 de la figure 2.4 ne sont pas en configuration générale puisque les trois derniers sont colinéaires. Avec suffisamment de correspondances (au moins 4), on obtient un système d'équations homogène possiblement surdéterminé. Dans ce cas, il est possible de trouver les meilleures valeurs de H selon un critère de moindres carrés. On résout en ajoutant la contrainte $|H| = 1$ afin d'éliminer la solution triviale $H = \mathbf{0}$. Ceci peut être accompli en utilisant la décomposition en valeurs singulières. Il est important de normaliser les points en utilisant la méthode décrite à la section 2.2.1. La méthode de minimisation par moindres carrés est simple à implanter, mais malheureusement, son erreur résiduelle n'a pas de sens géométrique. On utilise généralement la méthode linéaire pour initier une méthode non-linéaire qui minimise une erreur géométrique.

2.3.1 Erreur géométrique

Lorsqu'un des ensembles de points provient d'un plan de calibrage[‡] et que l'autre provient d'une image associée à une vue qui regarde le plan, on peut faire l'hypothèse que les points du plan de calibrage $\mathbf{q}_1^i$ ne sont pas bruités et que seulement les points

[‡] Un objet plat usiné avec un niveau élevé d'exactitude.

appartenant à l'image $\mathbf{q}_2^i$ le sont. La figure 2.5 contient une photo d'un plan de calibrage. On minimisera la fonction d'erreur asymétrique suivante

$$\sum_{i<M} d(\mathbf{q}_2^i, H\mathbf{q}_1^i).$$

La distance euclidienne entre les points en coordonnées homogènes $\mathbf{s}$ et $\mathbf{r}$ est $d(\mathbf{s}, \mathbf{r})$ et M est le nombre de points. Lorsque l'on minimise une erreur géométrique, il n'est pas nécessaire de normaliser les points au préalable.

L'algorithme de minimisation le plus utilisé est celui de Levenberg-Marquardt. Il s'agit d'un algorithme hybride entre la descente de gradient et l'algorithme de Gauss-Newton. Il nécessite une solution initiale près du minimum global afin d'éviter de trouver une solution qui serait un minimum local. Il est également possible d'effectuer la minimisation avec du bruit gaussien sur les deux ensembles de points. On utilise cette formulation, lorsque les points ont été extraits de deux images d'un plan commun dans le monde. La fonction d'erreur est

$$\sum_{i<M} d(\mathbf{q}_1^i, \widehat{\mathbf{q}}_1^i) + d(\mathbf{q}_2^i, H\widehat{\mathbf{q}}_1^i).$$

Les points $\widehat{\mathbf{q}}_1^i$ sont des estimés des points mesurés $\mathbf{q}_1^i$ et se rajoutent donc aux 9 éléments de H devant être estimés. Puisque les points $\widehat{\mathbf{q}}_1^i$ sont des inconnus, le nombre de paramètres à estimer est $9 + 2M$ au lieu de 9 dans le cas asymétrique. Les homographies peuvent être utilisées dans une procédure de calibrage d'une caméra. La prochaine section décrit une méthode répandue de calibrage.

2.4 Calibrage plan

Il est possible de retrouver les paramètres d'une caméra à partir de plusieurs photos prises d'un plan selon différents points de vue. La figure 2.5 représente un système servant au calibrage plan. La procédure contient 3 étapes distinctes. La première consiste à estimer les paramètres internes et par la suite, les paramètres

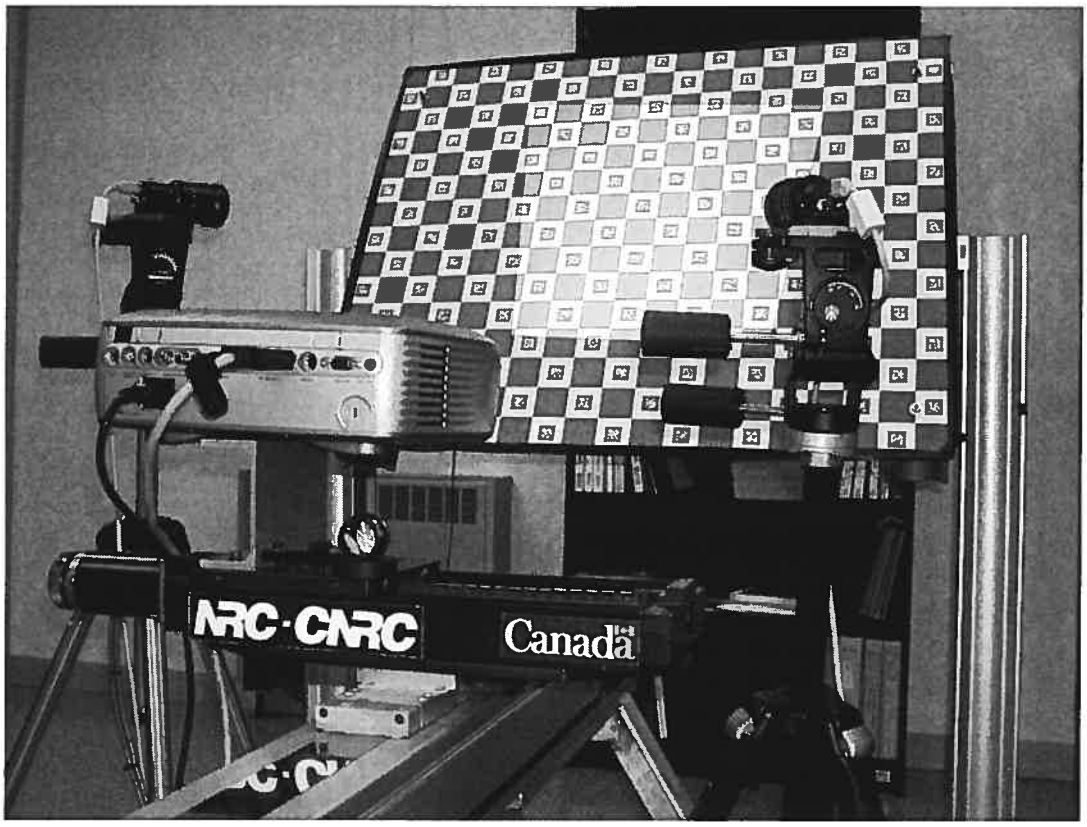

FIG. 2.5 – Photo du banc de calibrage plan pour projecteur. Chacun des carrés blancs contient une étiquette auto-identifiante ARTag permettant la mise en correspondance automatique entre le plan de calibrage et les caméras. Ce banc est utilisé pour générer les résultats du chapitre 9.

externes. Une minimisation non-linéaire est effectuée pour améliorer l'estimation des paramètres.

2.4.1 Paramètres internes

On peut fixer le système de coordonnées de manière à ce que tous les points sur le plan soient placés sur le plan $Z = 0$ du monde. La projection d'un point du plan $(X, Y, 0, W)$ dans la caméra est

$$(u, v, t) = K\,G\,(X, Y, 0, W).$$

Les matrices des paramètres internes et externes sont respectivement K et G. On peut constater que les éléments de la troisième colonne de G sont toujours multipliés par 0. On peut donc définir $G' = (\mathbf{r_1}, \mathbf{r_2}, \mathbf{t})$ pour obtenir

$$(u, v, t) = K\,G'\,(X, Y, W).$$

Il est possible d'estimer une homographie H entre les points du monde et les points de l'image en utilisant la méthode non-linéaire asymétrique décrite dans la section précédente. Nous avons donc $H = \lambda K\,G'$ et λ est un facteur d'échelle. Une fois que H est estimé, on peut utiliser les propriétés d'une matrice de rotation afin de former un système d'équations qui permettra de retrouver K. La première contrainte provient des colonnes d'une matrice de rotation qui sont orthogonales. La seconde provient des colonnes d'une matrice de rotation qui possèdent la même norme. Puisque λ est inconnu, la norme des colonnes peut être différente de un. Les contraintes sont respectivement

$$h_1^T K^{-T} K^{-1} h_2 = 0 \tag{2.5}$$

$$h_1^T K^{-T} K^{-1} h_1 - h_2^T K^{-T} K^{-1} h_2 = 0. \tag{2.6}$$

Avec suffisamment d'homographies, il devient possible de retrouver les paramètres internes de la caméra. Selon la forme de la matrice K définie à l'équation 2.1, la

manière de résoudre sera légèrement différente. Définissons $B = K^{-T}K^{-1}$, dans le cas le plus général de l'équation 2.1,

$$B = \begin{pmatrix} \frac{1}{f_x^2} & -\frac{s}{f_x^2 f_y} & \frac{-c_x f_y + c_y s}{f_x^2 f_y} \\ -\frac{s}{f_x^2 f_y} & \frac{f_x^2 + s^2}{f_x^2 f_y^2} & \frac{c_x f_y s - c_y(f_x^2 + s^2)}{f_x^2 f_y^2} \\ \frac{-c_x f_y + c_y s}{f_x^2 f_y} & \frac{c_x f_y s - c_y(f_x^2 + s^2)}{f_x^2 f_y^2} & 1 + \frac{c_y^2}{f_y^2} + \frac{(c_x f_y - c_y s)^2}{f_x^2 f_y^2} \end{pmatrix}.$$

Puisque B est symétrique, elle peut être représentée par un vecteur $\mathbf{b}$ de 6 éléments. Si $s = 0$ dans la matrice K alors, on aura

$$B = \begin{pmatrix} \frac{1}{f_x^2} & 0 & -\frac{c_x}{f_x^2} \\ 0 & \frac{1}{f_y^2} & -\frac{c_y}{f_y^2} \\ -\frac{c_x}{f_x^2} & -\frac{c_y}{f_y^2} & 1 + \frac{c_x^2}{f_x^2} + \frac{c_y^2}{f_y^2} \end{pmatrix}.$$

et B pourra être représenté par un vecteur $\mathbf{b}$ de 5 éléments.

Si $s = 0$ et $f_x = f_y$ alors, on aura

$$B = \begin{pmatrix} \frac{1}{f_x^2} & 0 & -\frac{c_x}{f_x^2} \\ 0 & \frac{1}{f_x^2} & -\frac{c_y}{f_x^2} \\ -\frac{c_x}{f_x^2} & -\frac{c_y}{f_x^2} & \frac{c_x^2 + c_y^2 + f_x^2}{f_x^2} \end{pmatrix}$$

dont le vecteur associé $\mathbf{b}$ contient 4 éléments distincts.

Les deux contraintes linéaires de l'équation 2.6 peuvent être réécrites sous la forme $\mathbf{vb} = 0$ dont $\mathbf{v}$ est une combinaison des éléments des homographies. Le système linéaire peut être résolu en utilisant les mêmes méthodes que celles utilisées pour l'homographie ou la matrice fondamentale. On peut par la suite récupérer la matrice K en utilisant la factorisation de Cholesky de la matrice B. Puisque les données sont bruitées, il est possible que la matrice B ne soit pas définie positivement. Dans un tel cas, il serait impossible de retrouver K.

2.4.2 Paramètres externes

Puisque la matrice K est maintenant connue et que

$$\frac{K^{-1}H}{\lambda} = G',$$

nous pouvons calculer $\lambda = ||K^{-1}\mathbf{h}_1|| = ||K^{-1}\mathbf{h}_2||$. On peut donc récupérer les valeurs de $\mathbf{r}_1$,$\mathbf{r}_2$ et $\mathbf{t}$. Le troisième vecteur colonne $\mathbf{r}_3$ de la matrice de rotation est obtenu en prenant le produit vectoriel des deux premières colonnes. Les trois vecteurs colonnes $\mathbf{r}$ ne forment pas toujours une matrice de rotation parce que $\mathbf{r}_1$ et $\mathbf{r}_2$ ne sont pas nécessairement orthogonaux. Pour obtenir une véritable matrice de rotation, il faut effectuer une décomposition en valeurs singulières $(\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3) = UDV^T$. La matrice de rotation est UV^T.

2.4.3 Minimisation non-linéaire

Afin d'effectuer la minimisation non-linéaire, la matrice de rotation est reparamétrisée en un vecteur de 3 éléments. La reparamétrisation utilisée se nomme formule de Rodrigues. Pour chacune des homographies, il y a donc 6 paramètres externes (3 de translation et 3 de rotation) à estimer en plus des paramètres internes dont le nombre peut varier entre 3 et 5. Cette minimisation est généralement effectuée en utilisant l'algorithme de Levenberg-Marquardt. La fonction d'erreur est

$$\sum_{i<N} \sum_{j<M} d(\mathbf{p}_i^j, KG_i'\mathbf{Q}^j).$$

Le nombre d'homographies est N et M est le nombre de points sur le plan de calibrage. Le point $\mathbf{Q}^j$ est un point sur le plan de calibrage et $\mathbf{p}_i^j$ est le point dans l'image i correspondant à $\mathbf{Q}^j$.

2.4.4 Distorsion radiale

Nous utilisons la méthode de calibrage plan dans le contexte du calibrage d'un projecteur. Le type de projecteur que nous avons utilisé contient une quantité négligeable de distorsion radiale. Nous n'avons donc pas estimé les paramètres de distorsion. Il est néanmoins possible d'estimer le centre de distorsion radiale et les paramètres κ (voir l'équation 2.2).

Une fois les paramètres internes et externes obtenus, il est possible d'estimer non-linéairement les paramètres reliés à la distorsion. Par la suite, les paramètres internes et externes sont réévalués en gardant fixes les paramètres de distorsion radiale. Ces deux minimisations non-linéaires sont répétées jusqu'à ce qu'un critère d'arrêt soit rencontré. Il est possible de calculer analytiquement les meilleurs paramètres de distorsion radiale lorsque les autres paramètres sont fixes. La minimisation non-linéaire utilisée pour obtenir les estimés des paramètres de distorsion est remplacée par une solution analytique. Il faut toujours alterner l'estimation des paramètres de distorsion avec ceux qui sont internes et externes.

Il est également possible d'exploiter la similarité entre le centre de distorsion radiale et l'épipôle associé à une matrice fondamentale lorsque l'épipôle est dans l'image. À partir des mêmes images qui servent au calibrage plan, une *matrice fondamentale pour la distorsion radiale* est estimée. Par la suite, il est possible d'extraire de cette matrice les homographies entre le plan de calibrage et la caméra. Ces dernières servent directement au calibrage plan.

2.5 Énumération commentée d'outils et de références

Les modèles de caméras, la matrice fondamentale, l'homographie ainsi que leurs propriétés et les algorithmes qui s'y rattachent sont présentés dans plusieurs monographies [42, 74, 22]. Les conditions minimales sont décrites en détail ainsi que les cas dégénérés. Le problème de triangulation est présenté dans différentes monographies [42, 74]. La normalisation des points est décrite dans la monographie de Hartley et Zisserman [42]. La méthode de minimisation non-linéaire de Levenberg-Marquardt ainsi que la méthode de résolution de systèmes linéaires par décomposition en valeurs singulières sont décrites dans la monographie de Hartley et Zisserman [42]. L'utilisation de la représentation de la matrice de rotation à l'aide de la formule de Rodrigues y est également décrite. La factorisation de Cholesky est

présentée dans différentes monographies et rapports techniques [124, 42, 74]. Les librairies *MINPACK* et *GSL* contiennent des implantations génériques de l'algorithme de Levenberg-Marquardt et peuvent être téléchargées des sites `http://www.netlib.org/minpack/` et `http://www.gnu.org/software/gsl/` respectivement. Les librairies *LAPACK* et *GSL* contiennent une implantation de la décomposition en valeurs singulières. *LAPACK* peut être téléchargée au `http://www.netlib.org/lapack/`. Une implantation de l'algorithme de Levenberg-Marquardt qui exploite la structure des problèmes rencontrés en vision par ordinateur est disponible sur le site web `http://www.ics.forth.gr/~lourakis/sba/`. Cette implantation est beaucoup plus rapide qu'une implantation générique et un rapport technique en décrit les détails [73]. Il existe d'autres algorithmes de minimisation non-linéaire parmi lesquels figure celui de Powell. Ce dernier a été comparé à celui de Levenberg-Marquardt sur un problème de minimisation non-linéaire fréquemment rencontré en vision par ordinateur [72]. L'algorithme de Powell obtient des résultats comparables mais il est plus rapide que celui de Levenberg-Marquardt. L'usage de la méthode de Levenberg-Marquardt demeure plus répandu. Le calibrage plan est présenté dans différents articles de journaux, de rapports techniques et de monographies [100, 42, 74, 125, 124]. Plusieurs implantations du calibrage plan sont disponibles gratuitement : *OpenCV* et le *Camera Calibration Toolbox for Matlab*. On peut les obtenir aux adresses `http://www.intel.com/technology/computing/opencv/` et `http://www.vision.caltech.edu/bouguetj/calib_doc/index.html`. Le programme *Microsoft Easy Camera Calibration Tool* peut être téléchargé de l'adresse `http://research.microsoft.com/~zhang/Calib/`. Les différents types de distorsion et d'aberration des systèmes optiques sont décrits dans la monographie de Mikhail *et al.* [77]. Les différentes manières d'incorporer la distorsion radiale au calibrage plan sont décrites dans différents articles et rapports techniques [124, 125, 41, 37]. Le système *ARTag* est décrit dans le rapport technique et l'article de Fiola [25, 26, 27]. La librairie peut être téléchargée du site web `http://www.cv.iit.nrc.ca/research/ar/artag/`.

Chapitre 3

INTRODUCTION À LA MINIMISATION COMBINATOIRE

Une classe de problèmes fréquemment rencontrée en vision par ordinateur consiste à associer une étiquette à chacun des pixels d'une image. Les images de scènes naturelles ou humaines sont en général spatialement cohérentes et deux pixels voisins appartiennent probablement au même objet du monde et possèdent des propriétés similaires. Une des manières efficaces de résoudre des problèmes d'étiquetage est d'attribuer une énergie à chacune des configurations qui associe une étiquette aux différents pixels et de trouver la configuration ayant la plus petite énergie.

Le présent chapitre s'intéresse aux algorithmes permettant de minimiser ces fonctions d'énergie qui comprennent un terme de vraisemblance indépendant pour chacun des sites et un terme de lissage qui dépend de deux sites simultanément. Nous ne présenterons pas l'ensemble des méthodes permettant de minimiser ce type de fonction d'énergie. Nous présenterons les méthodes suivante : programmation dynamique, propagation de croyances, flot maximal dans un graphe, coupe dans les graphes, propagation pondérée de croyances et programmation dynamique itérative. La description des algorithmes demeurera intuitive.

Une classe de problèmes fréquemment rencontrée en vision par ordinateur consiste à associer une étiquette à chacun des pixels d'une image. Les images de scènes naturelles ou humaines sont en général spatialement cohérentes et deux pixels voisins appartiennent probablement au même objet du monde et possèdent des propriétés similaires. Par exemple en stéréoscopie les étiquettes représentent des profondeurs et dans ce cas deux pixels voisins devraient posséder la même profondeur. Les algorithmes de vision par ordinateur exploitent généralement cette cohérence afin d'éliminer une partie des artefacts introduits par la présence de bruits dans les images.

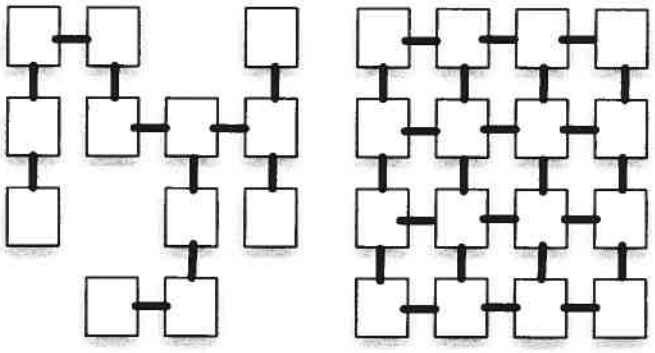

FIG. 3.1 – Graphes 2D associés à deux problèmes d'étiquetage : à gauche un graphe acyclique et à droite un graphe cyclique.

Typiquement, un voisinage 4 ou 8-connectés est utilisé. Une des manières efficaces de résoudre des problèmes d'étiquetage est d'attribuer une énergie à chacune des configurations qui associe une étiquette aux différents pixels et de trouver la configuration ayant la plus petite énergie.

Le présent chapitre s'intéresse aux algorithmes permettant de minimiser ces fonctions d'énergie qui comprennent un terme de vraisemblance indépendant pour chacun des sites et un terme de lissage qui dépend de deux sites simultanément. Nous ne présenterons pas l'ensemble des méthodes permettant de minimiser ce type de fonction d'énergie. Nous nous limiterons aux méthodes dont l'usage est répandu en stéréoscopie. Parmi les algorithmes qui ne seront pas couverts, mentionnons les algorithmes génétiques ainsi que l'ICM (*Iterative Conditional Modes*). La description des algorithmes demeurera intuitive et permettra aux lecteurs d'acquérir les bases nécessaires à la compréhension des prochains chapitres. Une récente étude compare les différents algorithmes de minimisation présentés dans ce chapitre [106].

À partir d'un ensemble de sites $\mathcal{P}$ et d'un ensemble d'étiquettes $\mathcal{D}$ on cherche à trouver une $\mathcal{D}$-configuration $f : \mathcal{P} \mapsto \mathcal{D}$ qui associe une étiquette à chacun des sites.

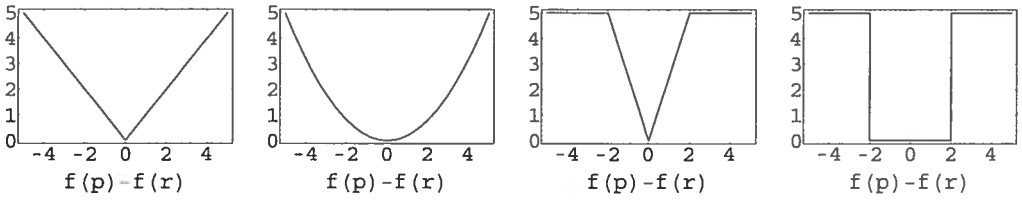

FIG. 3.2 – Différentes formes du terme de lissage. De gauche à droite : linéaire, convexe, linéaire tronqué et robuste.

La fonction d'énergie est

$$E(f) = \sum_{\mathbf{p}\in\mathcal{P}} \underbrace{e(\mathbf{p}, f(\mathbf{p}))}_{\textit{vraisemblance}} + \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} \sum_{\mathbf{r}\in\mathcal{N}_\mathbf{p}} s(\mathbf{p}, \mathbf{r}, f(\mathbf{p}), f(\mathbf{r}))}_{\textit{lissage}} . \tag{3.1}$$

$\mathcal{N}_\mathbf{p}$ est l'ensemble des sites appartenant au voisinage du site $\mathbf{p}$. Il est possible d'associer au problème, un graphe 2D où chacun des sites est représenté par un sommet. Deux sommets sont reliés par un arc lorsque les sites qu'ils représentent sont voisins (voir la figure 3.1). La forme du terme de lissage ainsi que la connectivité du voisinage influence la façon de minimiser cette fonction d'énergie. Lorsque le graphe associé au problème est acyclique, il est toujours possible de résoudre le problème de façon optimale par programmation dynamique. Lorsque le terme de lissage est de forme linéaire ou convexe (voir la figure 3.2), il est possible de minimiser la fonction de façon optimale, en utilisant les méthodes de flot maximal dans un graphe peu importe la structure du voisinage. Lorsque le terme est robuste avec un voisinage 4-connectés, le problème devient $\mathcal{NP}$, alors on se contente d'une solution approximative. La figure 3.2 illustre différentes formes du terme de lissage.

Dans les prochaines sections, plusieurs stratégies qui permettent de minimiser la fonction 3.1 seront présentées. La prochaine section présentera les méthodes basées sur les algorithmes de flot maximal dans les graphes. Puis, une seconde section présentera les méthodes utilisant la programmation dynamique et les autres méthodes dites de

propagation de messages.

3.1 Algorithmes de coupe minimale dans les graphes

Nous présenterons d'abord les algorithmes qui permettent de trouver le minimum global à des problèmes possédant des termes de lissage convexes. Par la suite, nous présenterons des méthodes permettant de trouver des solutions approximatives à des problèmes ayant des termes de lissage non convexes.

3.1.1 Flot maximal dans un graphe

Le problème de flot maximal dans un graphe consiste à déterminer la quantité maximale de flot qui peut être acheminée d'une source vers un drain en passant à travers un réseau d'arcs et de sommets. Chacun des arcs possède une capacité limitée de laisser passer du flot.

De manière formelle, nous définissons un graphe $G = (V, E)$ où V est l'ensemble des sommets et E l'ensemble des arcs reliant deux sommets de V. Il existe deux sommets spéciaux qui sont la source s et le drain t. On associe une capacité entière positive $cap(v, w)$ à l'arc partant de v et se terminant à w. Pour simplifier la discussion $cap(v, w) = 0$ lorsque l'arc n'existe pas. Un flot f dans un graphe G est défini comme étant $f : V^2 \rightarrow \mathcal{Z}^{\geq 0}$ et respecte les propriétés suivantes :

- $f(v, w) = -f(w, v)$ pour tous les v et w appartenant à V
- $f(v, w) \leq cap(w, v)$ pour tous les v et w appartenant à V
- $\sum_{w \in V} f(v, w) = 0$ pour tous les v appartenant à $V - \{s, t\}$

La dernière propriété se nomme contrainte de conservation du flot. La valeur du flot dans un graphe est $\sum_{w \in V} f(s, w)$. Le flot maximal d'un graphe est le flot qui maximise la sommation précédente tout en respectant les trois contraintes énoncées précédemment.

Une coupe dans le graphe G divise les sommets V en deux ensembles X et $X^C =$

$V - X$ de manière à ce que la source s appartienne à X et que le drain t appartienne à X^C. La capacité d'une coupe est $\sum_{v \in X, w \in X^C} cap(v, w)$. Trouver la coupe minimale d'un graphe consiste à trouver une coupe dont la capacité minimise la sommation précédente. Un théorème énoncé par Ford et Fulkerson stipule que la valeur du flot maximal dans un graphe est égale à la capacité de la coupe minimale [28].

Pour minimiser un problème d'étiquetage en utilisant le flot maximal dans un graphe, il suffit de construire un graphe dont la capacité de la coupe minimale correspond au minimum de la fonction d'énergie de l'équation 3.1. Notons que ce graphe est différent du graphe 2D associé au problème. La figure 3.3 illustre un graphe correspondant à un problème d'étiquetage possédant les sites i,j et k avec j voisin de i et de k. L'ensemble des étiquettes est $\{0, 1, 2\}$. Le terme de lissage est linéaire et défini comme étant

$$s(\mathbf{i}, \mathbf{j}, r, q) = \lambda |r - q|. \tag{3.2}$$

Cette formulation a été proposée par Roy [89]. La valeur de la coupe minimale illustrée à la figure 3.3 est égale à $e(i, 2) + e(j, 0) + e(k, 0) + 2\lambda$ et elle est aussi égale au minimum de la fonction d'énergie 3.1 lorsque le terme de lissage de l'équation 3.2 est utilisé. Notons que la capacité de certains arcs est égale au terme de vraisemblance qui n'est pas contraint d'être positif. Une astuce permettant de minimiser des fonctions contenant un terme de vraisemblance qui prend des valeurs négatives sera présentée à la section 3.1.2. Pour certains problèmes, il peut être préférable d'utiliser un terme de lissage convexe au lieu d'un terme linéaire. La figure 3.4 montre un graphe correspondant à un problème d'étiquetage avec un terme de lissage convexe. Cette formulation a été proposée par Paris [85]. Les arcs de capacité β permettent de s'assurer que le terme de lissage demeure convexe. Généralement, la valeur β est très petite. Un article d'Ishikawa contient plus de détails concernant l'utilisation du flot maximal dans les graphes afin de résoudre des problèmes avec des termes de lissage convexes [48].

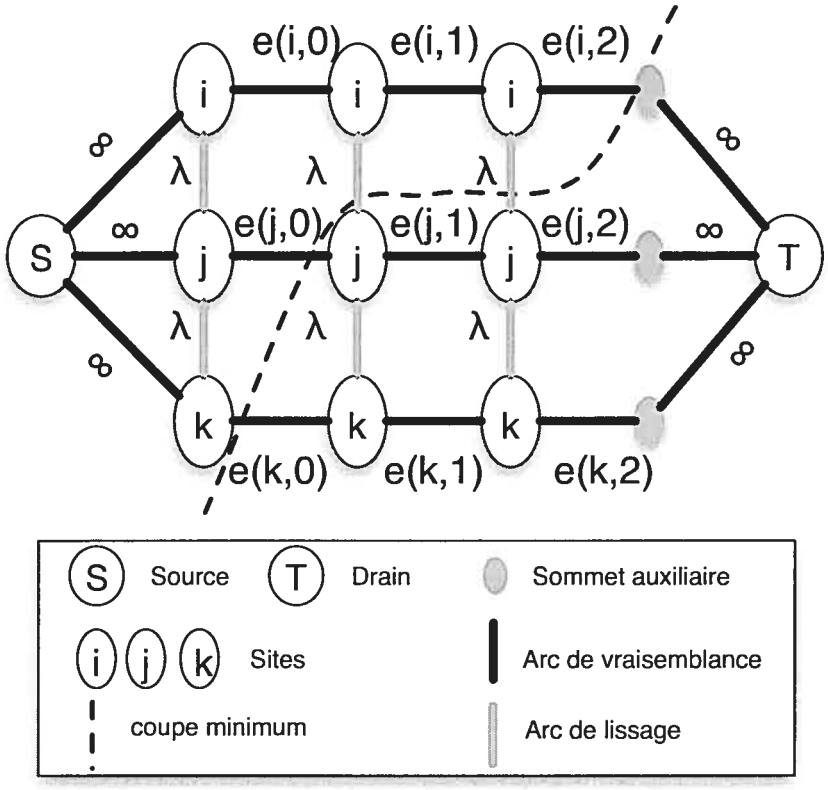

FIG. 3.3 – Formulation par flot maximal dans un graphe avec lissage linéaire Le problème comporte 3 sites et 3 étiquettes. La méthode a été proposée par Roy [89].

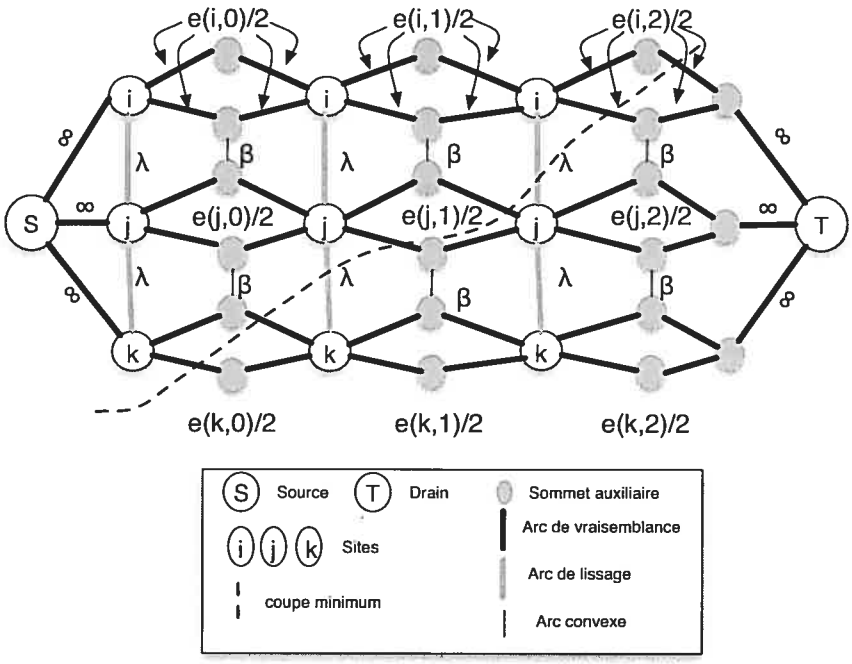

FIG. 3.4 – Formulation par flot maximal dans un graphe avec terme de lissage convexe. Le problème comporte 3 sites et 3 étiquettes. La méthode a été proposée par Paris [85].

Algorithme de flot maximal utilisé

Le flot maximal dans un graphe qui utilise un terme de lissage linéaire ou convexe peut être calculé par un algorithme basé sur le pré-flot. Un pré-flot est similaire à un flot à l'exception que la contrainte de conservation du flot est remplacée par la contrainte stipulant que $\sum_{w \in V} f(v,w) + e(w) = 0$ pour tous les sommets v appartenant à $V - \{s,t\}$ et où $e(w)$ est l'excès de flot du sommet w. L'excès ne peut jamais être négatif. L'algorithme de pré-flot générique est vorace et prend uniquement des décisions locales. Deux opérations de base peuvent être effectuées : *pousser* et *élever*. Une fois terminé, cet algorithme garantit que le pré-flot redevienne un flot et que celui-ci soit le flot maximal [34, 35, 13]. Une hauteur est associée à chacun des sommets et initialement, la source est à la hauteur $\#V$*. Les autres sommets ont une hauteur de zéro. Une opération *pousser* appliquée à un sommet v consiste à pousser le maximum de l'excès $e(v)$ dans l'arc reliant v au voisin w sans faire passer plus de flot que la capacité de l'arc, le permet. L'opération peut uniquement être effectuée sur un arc reliant un sommet d'une hauteur h à un sommet d'une hauteur $h-1$. Lorsque le sommet v possède un excès et qu'il n'est plus possible d'effectuer des opérations *pousser*, on effectue alors une opération *élever*. Une opération *élever* est appliquée à v afin de permettre à celui-ci de déverser son excès vers le voisin w qui est le plus bas et dont la hauteur est supérieure à h et dont l'arc (v,w) n'est pas saturé. Un arc (v,w) est saturé lorsque $cap(v,w) = f(v,w)$. Si la hauteur de w est h' alors, la hauteur de v deviendra $h'+1$ après l'opération *élever*. L'algorithme effectue ces deux opérations élémentaires sur les sommets du graphe selon un ordonnancement quelconque. L'algorithme se termine lorsqu'aucune opération élémentaire ne peut être effectuée.

Nous avons présenté la version générique de l'algorithme flot maximal de Golderg [34, 35, 13]. Les algorithmes basés sur le pré-flot possèdent généralement un ordonnancement plus sophistiqué. Ils effectuent également une opération supplémentaire

* $\#V$ représente le nombre d'éléments de l'ensemble V.

qui permet de changer simultanément la hauteur de tous les sommets du graphe en utilisant un critère global. La description de ce type d'algorithmes dépasse le cadre de ce chapitre et le lecteur pourra consulter [13, 34, 35].

Lorsqu'on utilise un terme linéaire ou convexe, on fait l'hypothèse que les étiquettes peuvent être ordonnées et que le lissage est proportionnel à l'écart entre les étiquettes. Lorsque les étiquettes ne peuvent pas être ordonnées en une dimension, un terme de lissage robuste doit être utilisé. Un des modèles non-convexe populaire est défini comme étant $l(\mathbf{p}, \mathbf{r}) = \delta(f(\mathbf{p}) - f(\mathbf{r}))$ où δ vaut 1 à 0 et 0 ailleurs ; celui-ci est nommé modèle de Potts. Il a été démontré que des problèmes d'étiquetage possédant plus de deux étiquettes et un terme de lissage de Potts appartiennent à la classe des problèmes $\mathcal{NP}$ [7]. Plusieurs méthodes qui permettent de trouver des solutions approximatives à la fonction d'énergie 3.1, en minimisant des fonctions d'énergie binaires seront présentées dans la prochaine section.

3.1.2 Coupe dans les graphes

Les méthodes de coupe dans les graphes sont itératives et permettent de minimiser des fonctions d'énergie avec un terme de lissage non-convexe. Elles débutent à partir d'une solution initiale pouvant être obtenue par recherche directe ou en fixant arbitrairement les étiquettes initiales. Nous présenterons deux algorithmes introduits par Boykov *et al.* [8] : l'échange α-β et l'expansion α. Le principe est de décomposer le problème difficile en une séquence de problèmes plus faciles, par des minimisations binaires, qui peuvent généralement être résolues globalement et efficacement.

L'échange α-β est constitué d'une boucle permettant d'examiner toutes les combinaisons de deux étiquettes. Pour chacune de ces combinaisons, une fonction binaire est minimisée de façon optimale. L'argument du minimum est utilisé afin de mettre à jour la solution courante. À chacune des mises à jour, il est garanti que l'énergie de la fonction va diminuer ou va demeurer identique. Le nombre de minimisations croît de façon quadratique avec le nombre d'étiquettes. L'expansion α est similaire, sauf

que la boucle examinant les combinaisons d'étiquettes est remplacée par une boucle examinant toutes les étiquettes. La fonction binaire à minimiser est évidemment différente. Le nombre de minimisations devant être effectué croît linéairement avec le nombre d'étiquettes.

Dans les deux cas, à l'intérieur des boucles, le problème de minimisation binaire est résolu à l'aide d'un algorithme de flot maximal dans un graphe. L'exécution de la boucle de l'échange α-β ou de la boucle de l'expansion α est un *cycle*. On exécute plusieurs *cycles* jusqu'à ce que l'énergie arrête de diminuer. Ces deux algorithmes sont garantis de converger [8, 111, 9]. Les pseudo-codes des algorithmes sont illustrés aux figures 3.5 et 3.6. Dans les prochaines sous-sections, les particularités de chacun des deux algorithmes seront décrites. Finalement, nous présenterons une généralisation de l'expansion α présentée par Kolmogorov et Zabih [58].

Échange α-β

Les sites sont séparés en deux ensembles. Le premier comprend les sites possédant les étiquettes α et β et se nomme un ensemble *actif*. Dans l'algorithme de la figure 3.5, l'ensemble *actif* est dénommé $\mathcal{R}$. Seulement les sites *actifs* peuvent changer d'étiquettes et font partie du problème de minimisation binaire. Après la mise à jour de la ligne 9, un site *actif* peut prendre la valeur de l'étiquette α ou β indépendamment de sa valeur initiale. Le terme $E_{\mathcal{R}}$ à la ligne 5 contient le terme de vraisemblance des sites *actifs* ainsi que le lissage entre les sites *actifs*. Les sites n'appartenant pas à l'ensemble *actif* appartiennent à l'ensemble *passif*. Le lissage entre un site *actif* et *passif* est calculé dans le terme $R_{\mathcal{R}}$ à la ligne 5. Lorsque le modèle de Potts est utilisé, il n'est pas nécessaire d'inclure le terme $R_{\mathcal{R}}$ puisque ce dernier ne fait qu'ajouter une constante à la fonction d'énergie à la ligne 5, ceci ne change pas la coupe minimale. Afin de pouvoir construire les graphes utilisés pour effectuer les

minimisations binaires, le terme de lissage doit respecter les conditions suivantes :

$$s(\mathbf{p}, \mathbf{q}, \alpha, \beta) \geq 0 \tag{3.3}$$

$$s(\mathbf{p}, \mathbf{q}, \alpha, \beta) = s(\mathbf{p}, \mathbf{q}, \beta, \alpha) \tag{3.4}$$

$$s(\mathbf{p}, \mathbf{q}, \alpha, \beta) \Leftrightarrow \alpha = \beta \tag{3.5}$$

pour tous les $\mathbf{p}$ et $\mathbf{q}$ appartenant à $\mathcal{P}$ et pour tous les α et β appartenant à $\mathcal{D}$. Le graphe associé à une des minimisations binaires d'un échange α-β pour un problème comportant les sites i, j et k et utilisant un modèle de lissage de Potts est illustré à la figure 3.7-gauche. Les étiquettes de i,j et k sont initialement α, β et γ. Puisque l'étiquette initiale du site k est différente de α et β, celle-ci est exclue de la fonction d'énergie binaire. La coupe minimale de la fonction d'énergie binaire est $e(\mathbf{i}, \beta) + e(\mathbf{j}, \alpha) + s(\mathbf{i}, \mathbf{j}, \beta, \alpha)$. Les étiquettes des sites i,j et k deviendront β, α et γ respectivement après la mise à jour de la ligne 9 de l'algorithme de la figure 3.5.

Expansion α

Dans un expansion α, tous les sites sont actifs simultanément. La minimisation binaire est faite de façon à ce que chaque site garde son étiquette courante ou prenne l'étiquette α. Cette minimisation se trouve à la ligne 4 de l'algorithme de la figure 3.6. La solution trouvée par l'expansion α est à un facteur de deux de la solution optimale lorsque le modèle de Potts est utilisé [111, 9]. Pour utiliser la méthode de l'expansion α, le terme de lissage doit respecter les conditions des équations 3.3, 3.4 et 3.5. De plus, l'inégalité du triangle doit être respectée. Explicitement,

$$s(\mathbf{p}, \mathbf{q}, \alpha, \beta) \leq s(\mathbf{p}, \mathbf{q}, \alpha, \gamma) + s(\mathbf{p}, \mathbf{q}, \gamma, \beta)$$

pour tous les $\mathbf{p}$ et $\mathbf{q}$ appartenant à $\mathcal{P}$ et pour tous les α, β et γ appartenant à $\mathcal{D}$. Le terme de lissage doit donc être une métrique. Un graphe représentant une expansion α avec les sites i,j et k est illustré à la figure 3.7-droite. Le site j est voisin des sites i et k. Les sites i,j et k avaient initialement les étiquettes respectives : α, β et γ.

ÉCHANGE $\alpha - \beta(f)$

1 **repeat**
2 $\quad$ *change* $\leftarrow 0$
3 $\quad$ **for** $\{\alpha, \beta\} \subset \mathcal{D}$
4 $\quad$ **do** $\mathcal{R} \leftarrow \{\mathbf{p} | f(\mathbf{p}) = \alpha$ ou $f(\mathbf{p}) = \beta\}$
5 $\quad\quad$ $E \leftarrow \min_{f_\mathcal{R}} [E_\mathcal{R}(f_\mathcal{R}) + R_\mathcal{R}(f|_{\mathcal{R}^c}, f_\mathcal{R})]$
6 $\quad\quad$ $f^\star_{\alpha\beta} \leftarrow$ minimum dans la formule précédente
7 $\quad\quad$ **if** $E < E_\mathcal{R}(f) + R_\mathcal{R}(f|_{\mathcal{R}^c}, f)$
8 $\quad\quad\quad$ **then** *change* $\leftarrow 1$
9 $\quad\quad$ Mettre à jour f avec $f^\star_\mathcal{R}$
10 $\quad$ **until** *change* $= 0$
11 **return** f

FIG. 3.5 – Pseudo-code de l'échange $\alpha - \beta$.

Notons que le graphe serait plus simple si les étiquettes initiales des sites j et k étaient identiques puisque $s(\mathbf{j}, \mathbf{k}, \beta. \beta) = 0$ et $s(\mathbf{j}, \mathbf{k}, \alpha, \beta) = s(\mathbf{j}, \mathbf{k}, \beta, \alpha)$. Le site i possède initialement l'étiquette α et il aura automatiquement l'étiquette α après l'expansion α. Pour ce faire, on ajoute un arc ayant une capacité infinie dans le graphe. La coupe minimale de la fonction d'énergie binaire est $e(\mathbf{i}, \alpha) + e(\mathbf{j}, \beta) + e(\mathbf{k}, \alpha) + s(\mathbf{i}, \mathbf{j}, \alpha, \beta) + s(\mathbf{j}, \mathbf{k}, \beta, \alpha)$. Les étiquettes des sites i, j, k deviennent α, β et α après la mise à jour de la ligne 8 de l'algorithme de la figure 3.6. Les preuves que ces constructions de graphes sont correctes se trouvent dans [8, 114, 9].

Autres fonctions binaires

Kolmogorov et Zabih fournissent une façon systématique de construire le graphe permettant de minimiser une fonction d'énergie binaire qui posséde des termes de lissage qui font interagir jusqu'à 3 sites simultanément [58]. Ce résultat est généralisé

EXPANSION $\alpha(f)$

```
1  repeat
2        change ← 0
3        for α ∈ 𝒟
4        do E ← min_{f_α} E(f_α))
5           f*_α ← minimum dans la formule précédente
6           if E < E(f)
7              then change ← 1
8           Mettre à jour f avec f*_α
9    until change = 0
10 return f
```

FIG. 3.6 – Pseudo-code de l'expansion α

à N sites par Freedman et Drineas [30]. Nous présenterons uniquement les résultats s'appliquant au cas où le lissage est limité à des interactions entre deux sites. La condition qui fait en sorte que le terme de lissage soit une métrique, sera remplacée par celle de sous-modularité [58]. Cette dernière stipule que

$$s(\mathbf{p},\mathbf{q},\beta,\beta) + s(\mathbf{p},\mathbf{q},\alpha,\alpha) \leq s(\mathbf{p},\mathbf{q},\beta,\alpha) + s(\mathbf{p},\mathbf{q},\alpha,\beta)$$

pour tous les $\mathbf{p}$ et $\mathbf{q}$ appartenant à $\mathcal{P}$ et pour les deux valeurs de la fonction binaire α et β. Elle est une condition nécessaire et suffisante pour permettre de minimiser une fonction binaire à l'aide de coupe minimale dans les graphes. La classe des fonctions pouvant être minimisées avec cette nouvelle formulation de graphes est plus large que celle présentée dans la sous-section précédente. Notons que l'ajout d'un terme constant à la fonction d'énergie ne change pas la coupe minimale du graphe. Par exemple, à la figure 3.8, le graphe de gauche possède la même coupe minimale que celui de droite. Le flot maximal est cependant différent de $e(\mathbf{i},\alpha) + e(\mathbf{j},\beta)$. Ceci

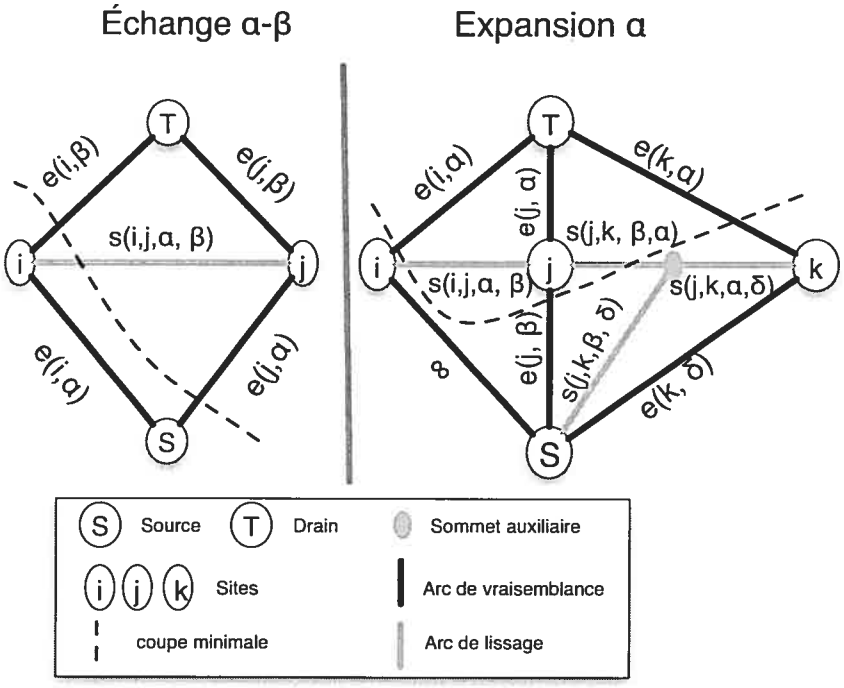

FIG. 3.7 – Minimisation d'une fonction binaire par coupe dans les graphes. La méthode a été proposée par Boykov *et al.* [8].

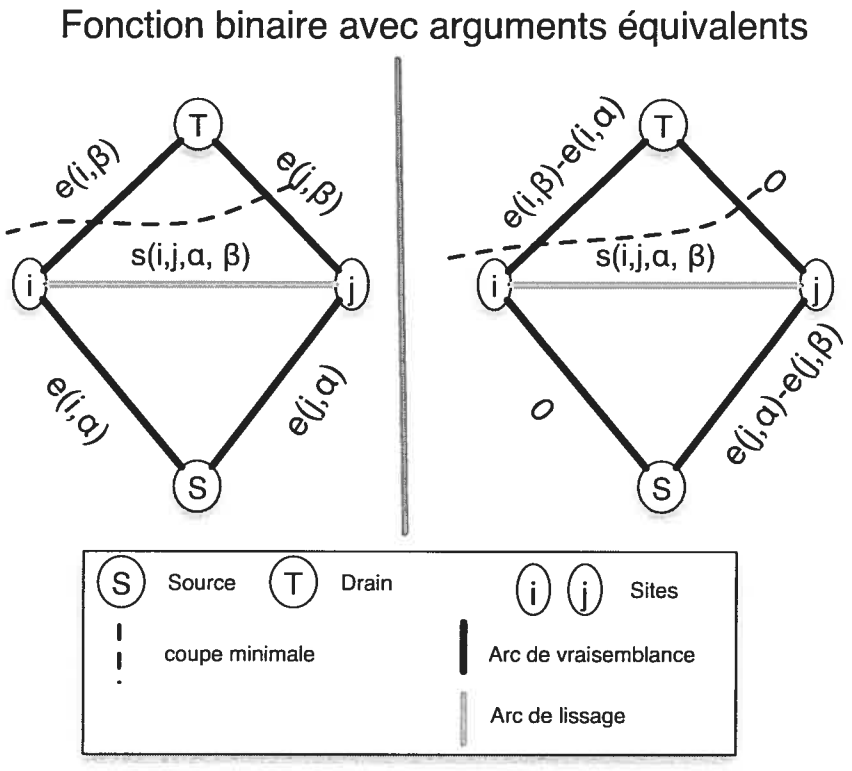

FIG. 3.8 – Graphe avec un terme de vraisemblance négative. Cette construction est possible lorsque $e(\mathbf{i}, \alpha) < e(\mathbf{i}, \beta)$ et $e(\mathbf{j}, \alpha) > e(\mathbf{j}, \beta)$.

permet de minimiser des fonctions d'énergie possédant un terme de vraisemblance pouvant prendre des valeurs négatives. Cette méthode s'applique également aux algorithmes de flot maximal utilisant un terme linéaire ou convexe. Lorsqu'on examine le cas se limitant à des interactions entre deux sites, la formulation de Kolmogorov et Zabih n'utilise aucun sommet auxiliaire. Les graphes sont donc plus compacts que ceux obtenus avec la formulation de l'expansion α de Boykov *et al.* [8]. La figure 3.9 illustre un graphe d'une expansion α pour les sites $\mathbf{i}$ et $\mathbf{j}$ dont les étiquettes avant la minimisation binaire sont β et γ. Les coupes minimales de gauche et de droite sont identiques, le flot maximal est cependant différent d'une constante. La contrainte de *sous-modularité* garantit que l'arc $(\mathbf{i}, \mathbf{j})$ est toujours positif. Ce type de contruction est possible grâce à un théorème prouvé par Kolmogorov [58]. Les règles de construction des graphes y sont également présentées.

Algorithme de flot maximal utilisé

Alors que les algorithmes basés sur le pré-flot sont généralement utilisés pour résoudre les problèmes utilisant un terme de lissage linéaire ou convexe, il a été démontré expérimentalement, qu'il est préférable d'utiliser un autre type d'algorithme pour les graphes représentant des fonctions binaires. Ce type d'algorithme est basé sur une stratégie de chemin *augmentant*. Définissons la capacité résiduelle d'un arc entre les sommets v et w comme étant $res(v, w) = cap(v, w) - f(v, w)$. Lorsqu'un arc ne contient aucun flot, la capacité résiduelle est égale à sa capacité. À partir des capacités résiduelles, on peut fabriquer un graphe résiduel dans lequel la capacité d'un arc est remplacée par sa capacité résiduelle. Un chemin *augmentant* est une suite $\{s_0, s_1, \ldots, s_{n-1}, s_n\}$ de sommets débutant par la source et se terminant par le drain tel que $res(s_i, s_{i+1}) > 0$ pour tous les sommets de la suite. La capacité d'un chemin *augmentant* est le minimum des capacités des arcs qui en font partie. On peut alors se servir de la capacité du chemin afin d'augmenter le flot qui passe dans les arcs appartenant à celui-ci, de manière à saturer l'arc de capacité minimale. Par la suite,

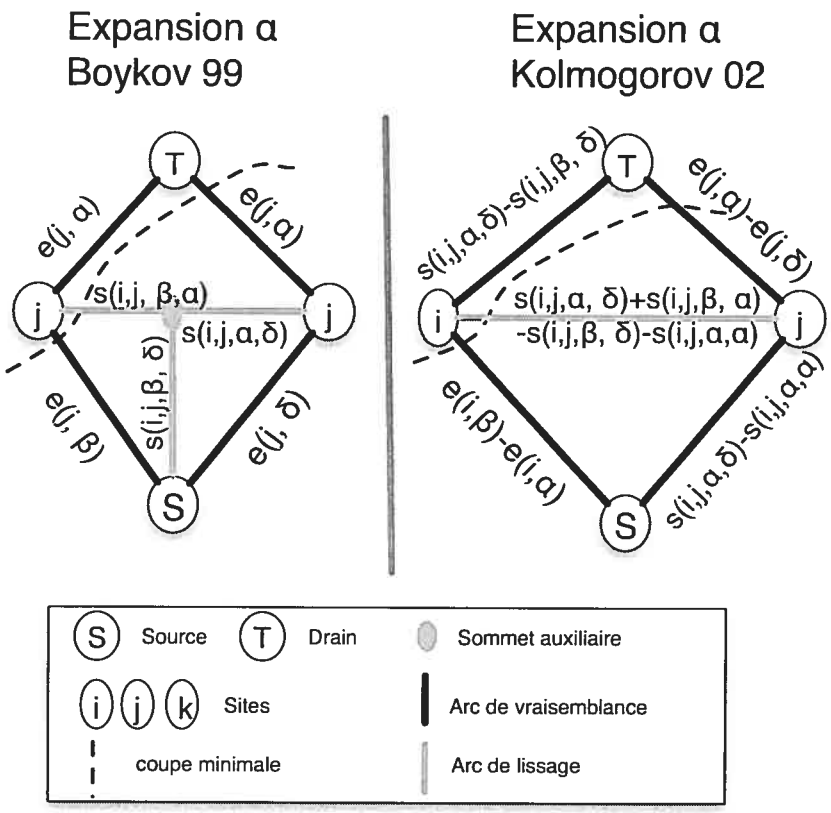

FIG. 3.9 – Différents graphes permettant d'effectuer un expansion α. La construction de droite est possible lorsque, nous devons avoir $e(\mathbf{i}, \alpha) < e(\mathbf{i}, \beta)$ et $e(\mathbf{j}, \alpha) > e(\mathbf{j}, \gamma)$. De plus, nous devons également avoir $s(\mathbf{i}, \mathbf{j}, \alpha, \gamma) \geq s(\mathbf{i}, \mathbf{j}, \beta, \gamma)$ et $s(\mathbf{i}, \mathbf{j}, \alpha, \gamma) \geq s(\mathbf{i}, \mathbf{j}, \alpha, \alpha)$.

un nouveau graphe résiduel est calculé. On répète cette procédure jusqu'à ce qu'il n'y ait plus de chemin *augmentant*. Généralement, il existe plus d'un chemin *augmentant* dans un graphe résiduel. Il faut donc un critère pour sélectionner l'un des chemins. L'algorithme de Ford et Fulkerson sélectionne le chemin de capacité maximale, alors que celui de Edmonds et Karp sélectionne le chemin de longueur minimale. Pour plus de détails, il est possible de consulter plusieurs monographies [13, 109]. Boykov et Kolmogorov ont développé un algorithme spécialement conçu pour les problèmes de minimisation de fonction binaire typiques à la vision par ordinateur [6]. Les graphes sont peu profonds et la longueur des chemins *augmentant* est généralement très courte. Deux graphes résiduels successifs sont très similaires et les chemins *augmentant* ne changent que très peu. L'algorithme de Boykov et Kolmogorov construit deux arbres de recherche en largeur, l'un ayant comme racine la source et l'autre le drain. Lorsque les deux arbres se touchent, c'est parce qu'il existe un chemin *augmentant* entre la source et le drain. Les capacités résiduelles des arcs le long du chemin sont modifiées et les arbres sont mis à jour sans devoir être complètement recalculés. Lorsque plusieurs problèmes similaires de flot maximal doivent être résolus, il existe des algorithmes spécialisés permettant de recycler le flot [54] ou les coupes [51].

Jusqu'à maintenant, nous avons présenté plusieurs méthodes qui utilisent des algorithmes de flot maximal dans les graphes. Dans la prochaine section, nous présenterons différentes méthodes basées sur la propagation de messages.

3.2 Les méthodes de propagation de messages

Nous débuterons par la présentation des méthodes qui sont basées sur la propagation de messages et qui permettent d'obtenir un minimum global : la programmation dynamique et la propagation de croyances sur un graphe acyclique. Par la suite, plusieurs méthodes approximatives de propagation de messages seront introduites. Nous terminerons en présentant différentes méthodes permettant d'accélérer le pas-

sage de messages. Ces dernières s'appliquent à tous les algorithmes présentés dans cette section.

3.2.1 Méthode exacte

Programmation dynamique

Afin d'être capable d'utiliser la programmation dynamique, il faut que le problème d'étiquetage que l'on tente de résoudre possède les propriétés suivantes :

- Les différents sites doivent être ordonnés en attribuant un index unique à chacun des sites.
- Le calcul de la meilleure carte d'étiquettes débutant à l'index 0 et se terminant à l'index i doit uniquement dépendre des étiquettes des sites 0 à $i-1$.

Nous débuterons en illustrant le principe de la programmation dynamique en utilisant un problème pour lequel, tous les sites sont alignés horizontalement et possèdent un voisinage 2-connectés. Il est possible d'indexer les sites de gauche à droite. Ce type de configuration a été utilisée fréquemment pour résoudre des problèmes de stéréoscopie [14]. Afin de calculer la meilleure carte se terminant à l'index i avec une étiquette d, il suffit d'examiner la meilleure carte allant jusqu'à l'index $i-1$ et terminer avec l'étiquette d' pour toutes les valeurs de d'. Avant de procèder, deux tables t et t' doivent être remplies. De façon explicite,

$$\begin{aligned} t(0,d) &= e(\mathbf{p}_0, d) \\ t'(0,d) &= d \end{aligned}$$

et pour $i > 0$,

$$\begin{aligned} t(i,d) &= e(\mathbf{p}_i, d) + \min_{d' \in \mathcal{D}} \begin{pmatrix} s(\mathbf{p}_{i-1}, \mathbf{p}_i, d', d) \\ + \quad t(i-1, d') \end{pmatrix} \\ t'(i,d) & \quad \text{est l'argument minimum dans la formule précédente} \end{aligned}$$

avec $\mathbf{p}_i$ qui est le site correspondant à l'index i. On dit que le calcul de $t(i,d)$ pour tous les d forme un message de $i-1$ vers i. Notons que $t(i,d)$ est l'énergie de la meilleure carte comprenant les sites 0 à i avec le site i possédant l'étiquette d. La solution peut-être récupérée en utilisant les relations de récurrence

$$f_{i,d}(i) = d \tag{3.6}$$

$$f_{i,d}(j) = t'(j+1, f_{i,d}(j+1)) \quad \text{pour} \quad 0 \leq j < i \tag{3.7}$$

où $f_{i,d}$ est la carte d'étiquettes de moindre énergie pour les sites dont les index sont entre 0 et i lorsque le dernier prend la valeur d'étiquette d. Pour le dernier site possédant l'index n, la valeur $t(n,d)$ correspond à l'énergie de la meilleure carte d'étiquettes pour tous les sites dont le dernier site possède l'étiquette d. Il est également possible de trouver l'énergie correspondant à la meilleure carte d'étiquettes ayant l'étiquette d associée au site d'index i pour tous les i et d. Explicitement, la table u contient l'énergie de la meilleure carte d'étiquettes pour tous les sites dont le site i possède l'étiquette d et est

$$\begin{aligned} u(n,d) &= t(n,d) \\ u'(n,d) &= d \end{aligned}$$

et pour $i < n$,

$$\begin{aligned} u(i,d) &= t(i,d) + \min_{d' \in \mathcal{D}} \left(\begin{array}{ll} & e(\mathbf{p}_{i+1}, d') + s(\mathbf{p}_i, \mathbf{p}_{i+1}, d, d') \\ + & u(i+1, d') - t(i+1, d') \end{array} \right) \\ u'(i,d) & \quad \text{est l'argument minimum dans la} \\ & \quad \text{formule précédente.} \end{aligned}$$

La table u' est similaire à la table t'. En utilisant les tables u' et t', il est possible de calculer cette carte de plus faible énergie en se servant d'une relation de récurrence similaire à celle des équations 3.6 et 3.7. Dans la prochaine section, nous généraliserons les résultats obtenus dans cette section, afin d'effectuer de la programmation dynamique sur des arbres.

Programmation dynamique sur un arbre

Lorsque le graphe associé au problème de minimisation est un arbre, un sommet peut être choisi arbitrairement et il devient la racine de l'arbre. Il est possible d'ordonner de manière linéaire les sommets de l'arbre en le parcourant en post-ordre (voir la figure 3.10). L'index de chacun des sommets est déterminé par l'ordre dans lequel les sommets sont visités. Une caractéristique de ce parcours en profondeur fait en sorte que les descendants d'un sommet sont visités avant celui-ci. Cet ordonnancement permet l'utilisation de la programmation dynamique et est illustré à la figure 3.10. Notons que dans la section précédente un site d'index i était connecté à un seul site d'index inférieur ; ici ce n'est plus le cas. La table t' d'étiquettes est remplacée par une table de vecteurs d'étiquettes. De façon explicite le remplissage des tables t et t' devient

$$t(i,d) = e(\mathbf{p}_i,d) + \sum_{j\in\mathcal{E}_i} \min_{d'\in\mathcal{D}} \begin{pmatrix} s(\mathbf{p}_j,\mathbf{p}_i,d',d) \\ + \quad t(j,d') \end{pmatrix} \tag{3.8}$$

$t'(i,d)$ est le vecteur des minimums de la formule précédente qui sont ordonnés par l'index croissant. (3.9)

$\mathcal{E}_i$ est l'ensemble formé des index des enfants du sommet ayant l'index i. Lorsque $\mathcal{E}_i$ est vide, le sommet est une feuille et

$$\begin{aligned} t(i,d) &= e(\mathbf{p}_i,d) \\ t'(i,d) &= (d). \end{aligned}$$

La carte d'étiquettes de moindre énergie est obtenue en parcourant l'arbre en post-ordre et en examinant pour chacun des sommets, l'entrée appropriée dans la table t'. Il est également possible de calculer les tables u et u'.

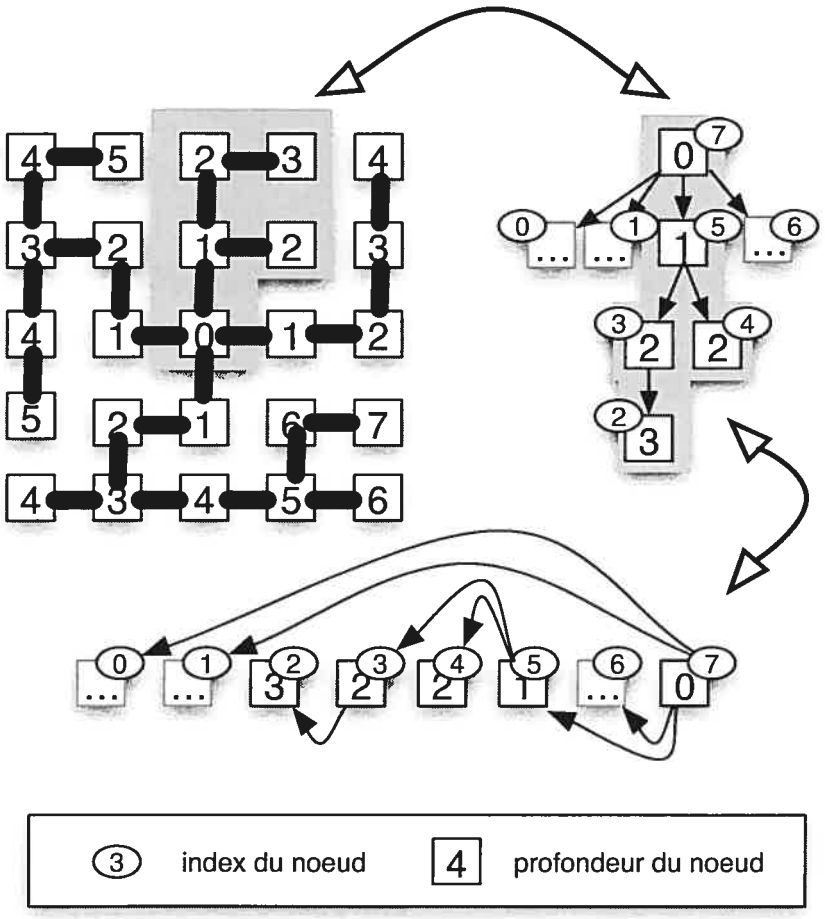

FIG. 3.10 – Programmation dynamique sur un arbre. Gauche) Arbre. Droite) Représentation d'une section de l'arbre de gauche. Bas) ordonnancement des sommets de l'arbre illustré à droite.

Propagation de croyances

Il est possible de reformuler le calcul des tables u et t d'une façon symétrique en calculant des messages. Cet algorithme est nommé propagation de croyances et peut être utilisé, lorsque le graphe associé au problème est un arbre. On utilise la terminologie de passage de messages entre les différents sommets. En programmation dynamique, le calcul de $t(i,d)$ pour un i fixe et tous les d correspondrait au passage d'un message entre $\mathbf{p}_{i-1}$ et $\mathbf{p}_i$. De la même manière, le calcul de $u(i,d)$ pour un i fixe et tous les d correspondrait au passage d'un message entre $\mathbf{p}_{i+1}$ et $\mathbf{p}_i$. Un message m de $\mathbf{p}_i$ vers $\mathbf{p}_j$ lorsque $\mathbf{p}_i$ et $\mathbf{p}_j$ possèdent respectivement les étiquettes d et d', est défini comme étant

$$m(i,j,d,d') = e(\mathbf{p}_j,d') + s(\mathbf{p}_i,\mathbf{p}_j,d,d') + \sum_{k \in \mathcal{N}_{\mathbf{p}_i} - \mathbf{p}_j} \min_{d'' \in \mathcal{D}} m(k,i,d'',d) - c_{i,j}$$

avec $c_{i,j}$ qui est une constante indépendante de d et d', mais qui peut varier avec i et j. Passer un message de $\mathbf{p}_i$ vers $\mathbf{p}_j$ consiste à calculer $\min_{d \in \mathcal{D}} m(i,j,d,d')$ pour tous les d'. L'ordonnancement utilisé pour la programmation dynamique sur un arbre est repris. Les messages sont passés en débutant par ceux qui partent des feuilles. Lorsque la racine a reçu tous ces messages, on passe les messages vers les feuilles, en utilisant l'ordonnancement inverse.

Le nombre total de messages passés est deux fois le nombre d'arcs de l'arbre. Une fois tous les messages passés, nous pouvons calculer l'accumulation de croyances du site d'index i à l'étiquette d comme étant

$$m(i,d) = \sum_{k \in \mathcal{N}_{\mathbf{p}_i} - \mathbf{p}_j} \min_{d' \in \mathcal{D}} m(k,i,d',d) - c_i \tag{3.10}$$

avec c_i qui est une constante indépendante de d. La valeur $m(i,d)$ correspond à l'énergie minimale, à une constante près, de la meilleure carte d'étiquettes comprenant tous les sites incluant le site $\mathbf{p}_i$ à l'étiquette d. En programmation dynamique, lors du calcul des tables t et u, il est également possible d'ajouter une constante. L'énergie finale différera d'une constante, mais la carte d'énergie minimale demeurera inchangée.

Une fois que tous les messages sont passés, l'algorithme converge sur un point fixe. Nous avons pour chacun des index i

$$f^*(\mathbf{p}_i) \in \arg\min_{q \in \mathcal{D}}(m(i,q)) \tag{3.11}$$

et pour chacun des i et j pour lesquels m est calculé

$$(f^*(\mathbf{p}_i), f^*(\mathbf{p}_j)) \in \arg\min_{r,q \in \mathcal{D}}(m(i,j,r,q)) \tag{3.12}$$

avec f^* qui est la configuration d'énergie minimale. Notons que $m(i,j,d,d')$ correspond à l'énergie, à une constante près, de la carte d'étiquettes d'énergie minimale dont les sites $\mathbf{p}_i$ et $\mathbf{p}_j$ possèdent respectivement les étiquettes d et d'. Lorsque l'ensemble de l'équation 3.11 contient un seul élément, alors cet élément fait partie de la solution de moindre énergie. De même, il est possible de fixer des étiquettes en examinant l'ensemble de l'équation 3.12. Néanmoins, la programmation dynamique doit être utilisée pour trouver les solutions optimales lorsque les équations 3.11 et 3.12 ne permettent pas de trouver une solution unique. Notons que la propagation de croyances est généralement présentée sous forme probabiliste.

3.2.2 Méthodes approximatives

Lorsque le graphe associé au problème contient au moins un cycle, il n'est plus possible de le résoudre directement par programmation dynamique. Il est cependant possible, de regrouper plusieurs sommets formant un cycle en un seul site. Par exemple, on peut regrouper 3 sites possédant chacun D étiquettes en un seul site possédant D^3 étiquettes. Cette méthode permet de résoudre de manière optimale, un problème d'étiquetage lorsque le graphe associé au problème contient quelques cycles. La complexité de l'algorithme augmente rapidement avec le nombre de cycles. Le lecteur désirant plus d'information sur les méthodes permettant d'effectuer des regroupements de sommets, peut consulter la littérature portant sur les arbres de jonctions (*Junction Tree*) [43].

Lorsque le graphe possède plusieurs cycles, il est possible de trouver une approximation de la solution. Par exemple, dans le graphe associé à un problème d'étiquetage les arcs n'ont pas tous la même *importance*. Rappelons-nous qu'un arc représente une interaction de lissage entre deux sites et que certains arcs pourraient être retirés du graphe sans que la carte d'étiquettes résultante ne change (l'énergie pourrait cependant varier). Si l'on possède une heuristique permettant d'associer des coûts faibles à chacun des arcs *importants* et des coûts élevés aux autres, il devient possible de construire un arbre de poids minimal contenant chacun des sites. Par la suite, la carte d'étiquettes d'énergie minimale associée à l'arbre est calculée. Cette stratégie a été proposée dans le contexte de la mise en correspondance stéréoscopique par Veksler [113]. Plusieurs autres stratégies d'approximation seront décrites dans les sous-sections suivantes.

Programmation dynamique itérative

Cette méthode a été proposée par Leung *et al.* [68]. On s'intéresse tout d'abord au cas pour lequel la carte d'étiquettes peut être séparée en lignes et en colonnes. On divise les sommets en deux ensembles. Le premier est celui des sommets actifs et il est composé des sites situés sur une des lignes ou une des colonnes. Les étiquettes des sites associés aux sommets actifs peuvent varier lors de la minimisation. Le second est composé des autres sites et leurs étiquettes sont fixes. Il s'agit de l'ensemble des sites passifs. On applique la programmation dynamique sur chacune des colonnes ou lignes de manière indépendante (voir la figure 3.11). Les arcs sont également séparés en deux ensembles : *actif* et *passif*. Les arcs reliant deux sommets *actifs* sont aussi *actifs* et les autres sont de type *passif*. Notons qu'au moins une des deux extrémités d'un arc *passif* est *passive*, étant donné que l'arc ne représente plus une interaction entre 2 sites. À chacune des évaluations du terme de vraisemblance on ajoute le coût de lissage avec les sommets passifs voisins. L'algorithme traite toutes les colonnes, ensuite toutes les lignes et recommence jusqu'à ce qu'il n'y ait plus de

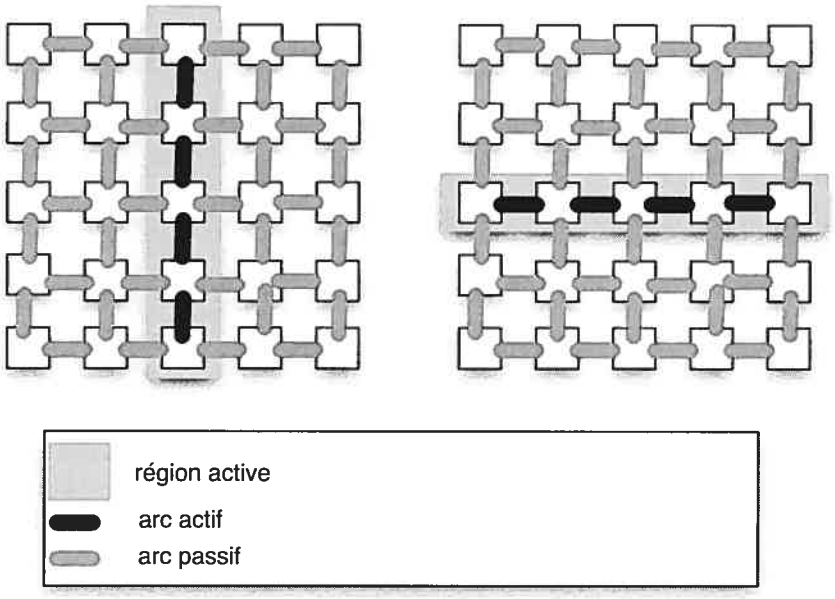

FIG. 3.11 – Programmation dynamique itérative.

changement. Une itération est appelée un *cycle*. Si après un cycle, l'énergie ne change pas alors l'algorithme a convergé. Autrement, l'énergie doit diminuer d'au moins la variation (positive) du coût associé à l'échange de l'étiquette d'un des sites par une autre étiquette. L'algorithme est garanti de converger, mais non pas vers le minimum global. Plus généralement, il est possible de remplacer les lignes et colonnes par un ensemble d'arbres et de faire la programmation dynamique itérative en utilisant des forêts.

Propagation de croyances sur graphe cyclique

Lorsque le graphe est acyclique, il existe un ordonnancement permettant de calculer chaque message une seule fois. Ce n'est plus le cas, lorsque le graphe possède des cycles. On utilise une formulation itérative avec un paramètre de temps t. On débute par l'initiation des messages au temps $t = 0$ à zéro. Puis, les messages sont

calculés et passés de manière itérative. Un message du site $\mathbf{p}_i$ vers $\mathbf{p}_j$ au temps t est calculé de la manière

$$m^t(i,j,q,r) = e(\mathbf{p}_j, q) + s(\mathbf{p}_i, \mathbf{p}_j, q, r) + \sum_{k \in \mathcal{N}_{\mathbf{p}_i} - \mathbf{p}_j} \min_{q' \in \mathcal{D}} m^{t-1}(k,i,q',q) - c_{i,j}. \quad (3.13)$$

Il y a plusieurs ordonnancements possibles pour le traitement des sites. Nous en décrirons trois :

- **Synchrone** Tous les sites calculent les messages qu'ils envoient vers leurs voisins au temps t en utilisant les messages reçu au temps $t-1$. C'est seulement après le calcul de tous les messages au temps t, que ceux au temps $t+1$ sont calculés. Le temps avance donc de façon synchrone.
- **Asynchrone** Le premier site d'une ligne calcule et passe son message vers le second site de la même ligne. Ce dernier se sert immédiatement de ce message pour calculer et passer son message dans la même direction. Le reste de la ligne est traité de manière similaire. Lorsque tous les messages dans cette direction sont passés, on recommence dans la direction inverse. Par la suite, cette procédure est répétée pour l'autre orientation. Le temps avance donc de façon asynchrone.
- **Bi-partie** Les sommets sont séparés en deux ensembles qui forment un graphe bi-partie. Les messages qui doivent être passés du premier ensemble vers le second sont calculés et passés. Puis, les messages du second vers le premier sont calculés et passés en utilisant les messages reçus les plus récemment.

Les deux premières méthodes sont tirées d'un article de Tappen et Freeman [107]. La méthode bi-partie a été proposée par Felzenszwalb et Huttenlocher [24]. Cette dernière peut être efficacement implantée sur un processeur graphique programmable [11].

Lorsqu'on utilise l'algorithme de propagation de croyances tel que nous l'avons décrit, il est possible que l'algorithme ne converge pas. Murphy *et al.* ont effectué une étude empirique afin d'essayer de comprendre ce phénomène [78]. Ils proposent d'uti-

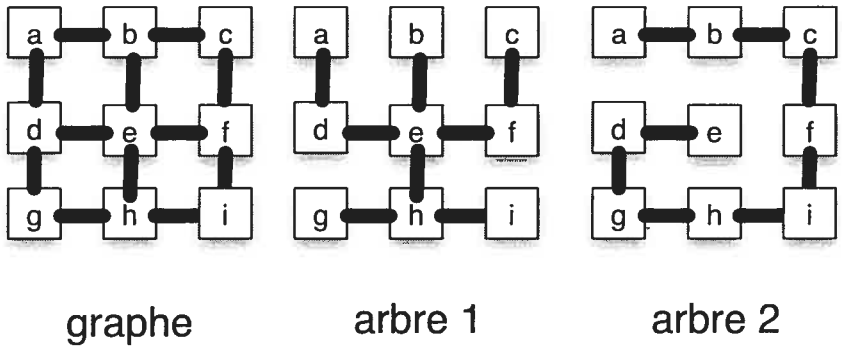

FIG. 3.12 – Partition en arbre d'un graphe cyclique : deux arbres dont la probabilité d'apparence est de $p^1 = p^2 = 1/2$. Les probabilités d'apparence c_{cf},c_{de},c_{gh} et c_{hi} sont toutes égales à 1 et les autres sont égales à $1/2$.

liser un facteur d'atténuation linéaire [†]. Ceci peut permettre de réduire les problèmes d'oscillations.

Il existe plusieurs méthodes pour extraire la solution. Nous décrirons uniquement la plus simple : les $f(\mathbf{p}_i)$ sont sélectionnés arbitrairement parmi les éléments de l'ensemble de l'équation 3.11. Il existe d'autres méthodes plus sophistiquées permettant d'extraire la solution [76]. Dans les prochaines sous-sections, nous présenterons d'autres algorithmes de propagation de croyances possédant des propriétés additionnelles.

Propagation pondérée de croyances

Dans cette méthode, la fonction d'énergie E est divisée en plusieurs fonctions d'énergie E^i comprenant un terme de vraisemblance e^i et un terme de lissage s^i.

[†] Le calcul de m^t est suivi de la mise à jour $m^t \leftarrow \alpha m^{t-1} + (1-\alpha)m^t$ avec α défini par l'usager.

Cette partition est faite de manière à ce que pour tous les $\mathbf{p}$ et $\mathbf{q}$ appartenant à $\mathcal{P}$,

$$\begin{aligned} \sum_{i<N} p^i e^i(\mathbf{p}, d) = &\quad e(\mathbf{p}, d) \\ \sum_{i<N} p^i s^i(\mathbf{p}, \mathbf{q}, d, d') = &\quad s(\mathbf{p}, \mathbf{q}, d, d') \end{aligned}$$

avec p^i qui est la probabilité d'apparence de la partition i. N est le nombre de partition. Lorsqu'une telle division est faite,

$$\min_f E(f) \geq \sum_{i<N} \min_{f^i} p^i E^i(f^i). \tag{3.14}$$

Les détails sont contenus dans un rapport technique de Wainwright *et al.* [117]. Notons que lorsque les termes de gauche et de droite sont égaux, le minimum global est atteint. Afin que cette inégalité soit utile, la fonction d'énergie de droite doit être plus facile à minimiser que celle de gauche. On décompose généralement le graphe associé au problème de gauche en plusieurs arbres de couverture de manière à ce que chacun des arcs du graphe soit inclus dans au moins un arbre. La figure 3.12 illustre la décomposition en arbre.

Afin de minimiser la fonction d'énergie de droite de l'équation 3.14, on utilise une méthode de propagation de croyances. Nous présenterons une décompostion en arbre qui permet de garantir que la borne inférieure de l'équation 3.14 ne diminuera jamais. Cette variante a été présentée par Kolmogorov [55]. Pour chacun des arcs du graphe, on calcule une probabilité d'apparence $c_{i,j}$ (voir la figure 3.12). On attribue à chacun des sommets un index. Les arbres sont des chaînes monotoniques selon leur index. L'ordonnancement sélectionne les sommets en ordre croissant d'index. Par la suite, on passe les messages avec l'ordonnancement inverse. On répète jusqu'à ce qu'un critère d'arrêt soit respecté. Le calcul d'un message $m^t(i, j, d, d')$ devient

$$m^t(i, j, d, d') = \min_{q' \in \mathcal{D}} \left(\begin{array}{ll} & c_{i,j} \left(e(\mathbf{p}_j, d) + \sum_{k \in \mathcal{N}_{\mathbf{p}_i}} m^{t-1}(k, i, q', d) \right) \\ - & m^{t-1}(j, i, d', d) + s(\mathbf{p}_i, \mathbf{p}_j, d, d') \end{array} \right) - c_{i,j}. \tag{3.15}$$

Notons que lorsque $c_{i,j}$ est 1, les équations 3.13 et 3.15 sont équivalentes. Cette variante de la propagation pondérée de croyances trouve une solution optimale lorsque la fonction d'énergie est binaire et sous-modulaire [56].

3.2.3 Accélération du passage de messages

Accélérer le passage de messages permet de réduire significativement le temps de calcul. Il existe deux catégories de méthodes permettant d'accélérer le passage de messages. La première famille est exacte et ne change pas la solution. Il est possible d'utiliser ce type d'accélération avec plusieurs types de termes de lissage. La seconde famille de méthodes permet d'obtenir une solution approximative.

Les méthodes exactes

Le modèle de Potts est le plus facile à accélérer. Le principe est illustré en utilisant la relation de récurrence de la programmation dynamique. On peut reformuler la relation de récurrence de l'équation 3.9 afin d'effectuer un passage accéléré des messages

$$t(i,d) = e(\mathbf{p}_i, d) + \min \left\{ \begin{array}{l} t(i-1, d'_{min}) + s(\mathbf{p}_{i-1}, \mathbf{p}, d'_{min}, d), \\ t(i-1, d) \end{array} \right\}$$

avec $d'_{min} = \arg\min t(i-1, d')$. Ceci est possible, puisque le terme $s(\mathbf{p}_{i-1}, \mathbf{p}, d', d)$ est identique pour toutes les valeurs de d et d' lorsque $d \neq d'$. L'index d'_{min} est calculé une seule fois pour chaque i. Le passage d'un message peut donc être calculé en $O(\#D)$ plutôt qu'en $O(\#D^2)$.

Il est également possible d'accélérer le passage de messages lorsque le terme de lissage est linéaire (voir l'équation 3.2). La figure 3.13 contient pour chacune des étiquettes d un point $(d, t(i-1, d))$. En plus de ces points, les cônes de coûts de lissage associés aux points sont également montrés. Par exemple, le point $(1, t(i-1, 2) + \lambda)$ correspond au coût $s(i-1, i, 2, 1) + e(i-1, 2)$ et $(3, t(i-1, 2) + \lambda)$ au coût

$s(i-1,i,2,3)+e(i-1,2)$. Tous ces points sont sur le même cône dont le sommet est $(2,t(i-1,2))$. Le passage d'un message peut être fait efficacement, puisqu'il s'agit uniquement d'examiner l'enveloppe inférieure formée par les cônes. Par exemple, la table $t'(i,d)$ ne prendra jamais la valeur 4 peu importe la valeur de d, puisque $t(i-1,4)$ ne fait pas partie de l'enveloppe inférieure. Lorsqu'on parcoure les étiquettes de gauche à droite (ou *vice-versa*), à chaque étiquette d, le cône qui fait partie de l'enveloppe minimale demeure le même ou il est remplacé par celui dont le sommet se trouve vis-à-vis d.

Toutes les valeurs de la table t pour le site i peuvent être calculées en parcourant trois fois les valeurs de la table t pour le site $i-1$. La figure 3.14 contient le pseudo-code permettant de calculer $t(i,d)$ pour toutes les valeurs de d. La première boucle (lignes 1 et 2) permet de considérer le cas où les sites $i-1$ et i possèdent la même étiquette. La deuxième boucle (lignes 3 et 4) permet de considérer les cas dans lesquels $t'(i,d)$ sera plus petit que d pour toutes les valeurs de d. La dernière boucle considère la situation inverse. Cette méthode peut également être adaptée aux modèles utilisant un terme de lissage convexe et convexe tronqué [24].

Les méthodes approximatives

Un algorithme heuristique d'émondage d'étiquettes a été proposé afin de résoudre des problèmes d'étiquetage, par propagation de croyances cyclique possédant un nombre très élevé d'étiquettes [60]. Pour ce faire, on doit attribuer une priorité basée sur une mesure de l'ambiguïté du terme de vraisemblance à chacun des sites. Les sites peu ambigus propagent leurs messages de façon prioritaire. On peut décider d'éliminer certaines étiquettes de la liste de celles qui sont associées à un site, en utilisant uniquement l'accumulation d'évidence de celui-ci. On réduit ainsi la taille des messages devant être passés, de manière significative.

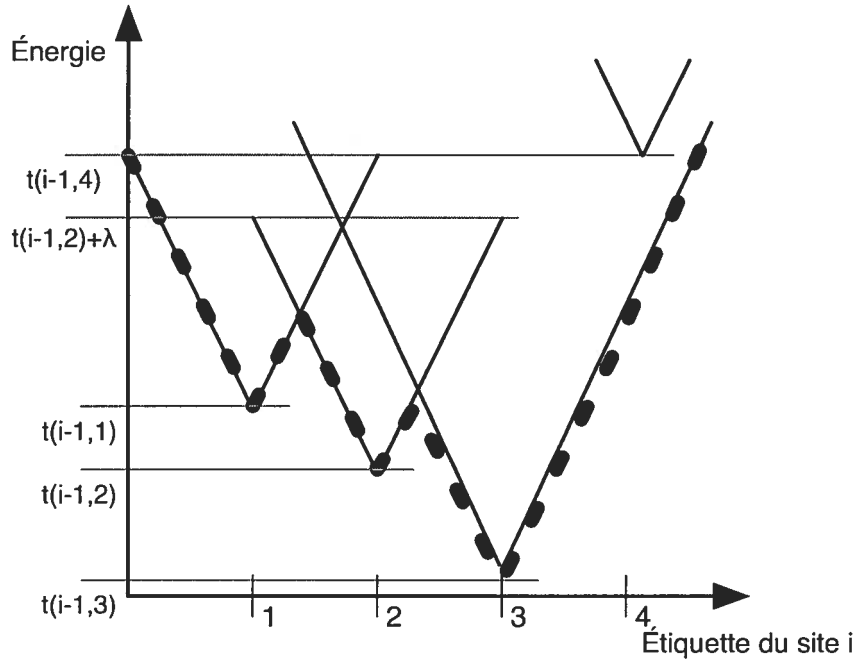

FIG. 3.13 – Passage rapide de messages avec terme de lissage linéaire

LINÉAIRERAPIDE(t, i, λ, N)

1 **for** $d \leftarrow 0$ **to** $N-1$
2 **do** $t(i,d) \leftarrow t(i,d) + t(i-1,d)$
3 **for** $d \leftarrow 1$ **to** $N-1$
4 **do** $t(i,d) \leftarrow \min\{t(i,d), t(i,d-1)+\lambda\}$
5 **for** $d \leftarrow N-2$ **to** 0
6 **do** $t(i,d) \leftarrow \min\{t(i,d), t(i,d+1)+\lambda\}$
7 **return**

FIG. 3.14 – Pseudo-code du passage rapide de messages avec terme de lissage linéaire où N est le nombre d'étiquettes et λ est la constante de lissage.

3.3 Discussion

Les méthodes de coupe dans les graphes ainsi que la propagation pondérée de croyances sont parmi les algorithmes de minimisation les plus efficaces [106]. Elles permettent de résoudre de façon satisfaisante les problèmes de stéréoscopie à deux caméras [106]. Meltzer *et al.* [76] ont réussi à trouver le minimum global sur les paires stéréoscopiques utilisées dans une étude comparative. Pour ce faire, ils ont utilisé une méthode de propagation pondérée de croyances. Ils commencent par exécuter leur algorithme, ce qui permet d'identifier les pixels qui ont une solution optimale. Ils présentent trois stratégies afin d'essayer de trouver la solution des autres sites. Il existe un lien entre la programmation linéaire et la propagation pondérée de croyances. De plus amples informations sont contenues dans le rapport technique de Wainwright *et al.* [117]. La méthode de coupe dans les graphes a été modifiée afin de fournir une borne sur l'énergie similaire à celle obtenue par la propagation pondérée de croyances [61]. Veksler [114] a proposé plusieurs stratégies permettant de réduire l'espace de recherche. Cet émondage de l'espace de recherche est effectué avant d'utiliser l'algorithme de coupe dans les graphes afin de trouver une solution approximative. Il existe également des méthodes permettant de décomposer la carte d'étiquettes en blocs de pixels [90]. Les sites du problème de minimisation seront les blocs. Certaines de ces méthodes utilisent une segmentation des images pour former les blocs [67]. D'autres méthodes utilisent une taille initiale fixe de blocs. Elles procèdent en minimisant la fonction d'énergie et en subdivisant les blocs selon le résultat de la minimisation [90]. Ces deux étapes sont répétées jusqu'à ce qu'un critère d'arrêt soit rencontré.

Dans les prochains chapitres portant sur la stéréoscopie à vues multiples, nous examinerons différentes méthodes permettant d'incorporer les interactions entre les sites provenant de la modélisation et de la visibilité.

Chapitre 4

INTRODUCTION À LA MISE EN CORRESPONDANCE STÉRÉOSCOPIQUE

La stéréoscopie consiste à retrouver la structure 3D d'une scène à partir de deux images de celle-ci. Typiquement en reconstruction stéréoscopique, une des deux caméras est désignée comme étant la caméra de référence. Pour chacun des pixels de l'image associée à celle-ci, on cherche le pixel de l'image de support correspondant au même point 3D. Lorsque la calibration ou la matrice fondamentale est connue, cette recherche est limitée à une recherche uni-dimensionnelle le long d'une ligne épipolaire. Le problème de stéréoscopie consiste à partir d'un ensemble de pixels de référence $\mathcal{P}$, d'un ensemble de disparités $\mathcal{D}$ à trouver une $\mathcal{D}$-configuration $f : \mathcal{P} \mapsto \mathcal{D}$ qui associe une disparité à chacun des pixels de référence. Le problème est fréquemment formulé sous forme d'une minimisation d'énergie comportant un terme de vraisemblance indépendant pour chacun des pixels de l'image de référence et un terme de lissage qui pénalise la différence de disparité entre deux pixels voisins. Typiquement, un voisinage 4 ou 8-connectés est utilisé et $\mathcal{N}_{\mathbf{p}}$ est l'ensemble des pixels appartenant au voisinage du pixel $\mathbf{p}$. La fonction d'énergie est

$$E(f) = \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} e(\mathbf{p}, f(\mathbf{p}))}_{\textit{vraisemblance}} + \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} \sum_{\mathbf{r}\in\mathcal{N}_{\mathbf{p}}} s(\mathbf{p}, \mathbf{r}, f(\mathbf{p}), f(\mathbf{r}))}_{\textit{lissage}} . \tag{4.1}$$

Le terme de vraisemblance est e et se définit comme étant $e(\mathbf{p}, d) = c_i(\mathbf{p}, d)$, c_i est la fonction de coût pour la caméra i. Cette fonction d'énergie peut être minimisée avec l'un des algorithmes présentés dans le chapitre 3.

Notons que dans le contexte de la reconstruction à images multiples, le terme de vraisemblance n'est plus directement égal à la fonction de coût. Une fonction de coût

qui pourrait être utilisée serait simplement $c_i(\mathbf{q}, d) = (I_{ref}(\mathbf{q}) - I_i(\mathbf{q}+d))^2$. I_{ref} et I_i sont les images respectivement associées à la caméra de référence et à la caméra de support i. En pratique, pour obtenir de meilleurs résultats, il est préférable d'utiliser une fonction de coût modélisant les problèmes d'échantillonnage. La mesure de *dissimilarité* de Birchfield et Tomasi sera utilisée [4]. Nous débutons par l'introduction des fonctions intermédiaires

$$I_{ref}^{min}(\mathbf{p}) = \min_{\mathbf{p}' \in S_\mathbf{p}} I_{ref}(\mathbf{p}')$$

et

$$I_{ref}^{max}(\mathbf{p}) = \max_{\mathbf{p}' \in S_\mathbf{p}} I_{ref}(\mathbf{p}').$$

$S_\mathbf{p}$ est l'ensemble $\{\mathbf{p} \pm (\frac{1}{2}, 0), \mathbf{p} \pm (0, \frac{1}{2})\}$. L'intensité d'un point est obtenue par interpolation bi-linéaire *. *Mutatis mutandis* I_i^{min} et I_i^{max} sont définis pour l'image de support i. Puis, nous introduisons

$$D(i, j, k) = \max\{0, k - i, j - k\}.$$

Finalement, la fonction de coût est

$$c_i(\mathbf{q}) = \min\{D(I_{ref}^{max}(\mathbf{p}_{ref}), I_{ref}^{min}(\mathbf{p}_{ref}), I_i(\mathbf{p}_i)), D(I_i^{max}(\mathbf{p}_i), I_i^{min}(\mathbf{p}_i), I_{ref}(\mathbf{p}_{ref}))\}$$

avec $\mathbf{p}_i = \mathbf{p}_{ref} + d$, d est la disparité. Il a été démontré expérimentalement qu'il est préférable d'utiliser l'interpolation bi-cubique au lieu d'une interpolation bi-linéaire [104]. Tous les résultats expérimentaux présentés ont été générés en utilisant l'interpolation bi-linéaire afin de faciliter la comparaison avec d'autres algorithmes. Il existe d'autres fonctions de coût qui permettent de se prémunir contre les artefacts provenant de l'échantillonnage. Ces dernières sont décrites dans un article Szeliski

* Notons que dans l'article original présentant cette mesure, l'interpolation était faite uniquement le long des lignes épipolaires. L'ensemble $S_\mathbf{q}$ contenait donc uniquement 3 éléments. Nous présentons la variante introduite par Kolmogorov et Zabih [59].

et Scharstein [104]. Plus récemment, une nouvelle fonction de coût a été présentée par Yoon *et al.* [121]. Cette dernière permet d'améliorer significativement la qualité des cartes de disparités. Notons que toutes ces fonctions de coût supposent que les surfaces de la scène sont lambertiennes. L'*a priori* stipulant que deux pixels voisins devraient avoir des disparités identiques, permet d'enlever l'ambiguïté. Cette dernière peut être dû aux images qui contiennent des textures répétitives, ou pas de texture, ou des surfaces non lambertiennes. Cette information supplémentaire est intégrée dans la fonction d'énergie grâce au terme de lissage s. Celui-ci est de la forme

$$s(\mathbf{p}, \mathbf{r}, f(\mathbf{p}), f(\mathbf{r})) = h(\mathbf{p}, \mathbf{r})\, l(f(\mathbf{p}) - f(\mathbf{r})). \tag{4.2}$$

h est une fonction *ad hoc* qui dépend des intensités de l'image de référence et est fréquemment définie comme étant

$$h(\mathbf{p}, \mathbf{r}) = \begin{cases} 3\lambda & si\ |I_{ref}(\mathbf{p}) - I_{ref}(\mathbf{r})| < 5 \\ \lambda & sinon. \end{cases} \tag{4.3}$$

La fonction l caractérise le modèle de lissage. On peut utiliser l'un des modèles décrits au chapitre 3. Récemment, une nouvelle fonction h tenant compte des variations d'intensité présentes dans les images de référence et de support a été proposée [121]. Néanmoins, dans les résutats présentés on a utilisé la fonction h qui est définie à l'équation 4.3. Le paramètre λ est défini par l'usager. Une étude comparative portant sur les différents algorithmes de stéréoscopie binoculaire a été publiée en 2002 par Scharstein et Szeliski [94]. Le site web de cette étude est au `www.middlebury.edu/stereo`. Il contient les résultats de nombreux algorithmes que nous ne décrirons pas.

4.1 Gestion des occultations

L'hypothèse stipulant que chacun des pixels de l'image de référence correspond à un pixel de l'image de support est faite lorsque la fonction d'énergie de l'équation 4.1 est utilisée. Cette hypothèse n'est pas satisfaite dans les zones d'occultations. Une

partie de la scène peut être vue par la première caméra et non par la seconde (voir figure 4.1). Certains pixels de l'image de référence n'ont pas de pixels correspondants dans la caméra de support. L'erreur introduite par les occultations peut être considérée négligeable lorsque la distance entre les caméras est petite. La quantité d'erreurs ne peut plus être négligée lorsque la distance est grande puisque la quantité d'occultations est proportionnelle à la distance entre les caméras. Lorsqu'un pixel de l'image de référence n'est pas visible dans la caméra de support, il est impossible de lui associer une profondeur. Toutes les disparités, faisant en sorte que le point 3D (formé par le pixel et la disparité) ne soit pas visible dans l'image de support, sont valides.

Afin de modéliser les occultations, il est possible d'ajouter une étiquette d'occultation o à l'ensemble $\mathcal{D}$. La fonction d'énergie demeure la même que celle présentée à l'équation 4.1. Le terme de vraisemblance e est cependant remplacé par

$$e_o(\mathbf{p}, d) = \begin{cases} \lambda_c & \text{si } d = o \\ c(\mathbf{p}, d) & \text{sinon} \end{cases} \tag{4.4}$$

avec λ_c qui est un coût d'occultation défini par l'usager. Il a été proposé de minimiser cette nouvelle fonction d'énergie par coupe dans les graphes [57]. Intuitivement, lorsque l'algorithme attribue l'étiquette d'occultation à un pixel, cela ne signifie pas qu'il y a une occultation géométrique. Le coût (en tenant compte du lissage) est simplement trop élevé pour toutes les étiquettes de disparités.

Plusieurs méthodes utilisent la programmation dynamique pour résoudre des fonctions d'énergie possédant des étiquettes d'occultations. La programmation dynamique a l'avantage de travailler avec des termes plus complexes. Néanmoins, le lissage est 2-connectés [14, 2, 32]. Une méthode proposant un traitement plus géométrique de la visibilité a été introduite par Kolmogorov et Zabih [59]. Dans leur méthode, l'ensemble des pixels de référence est étendu à l'ensemble des pixels des deux images et la solution est garantie d'être géométriquement compatible. Egnal et Wildes ont effectué une étude comparative présentant cinq stratégies de gestion des occultations [20] pour les configurations binoculaires.

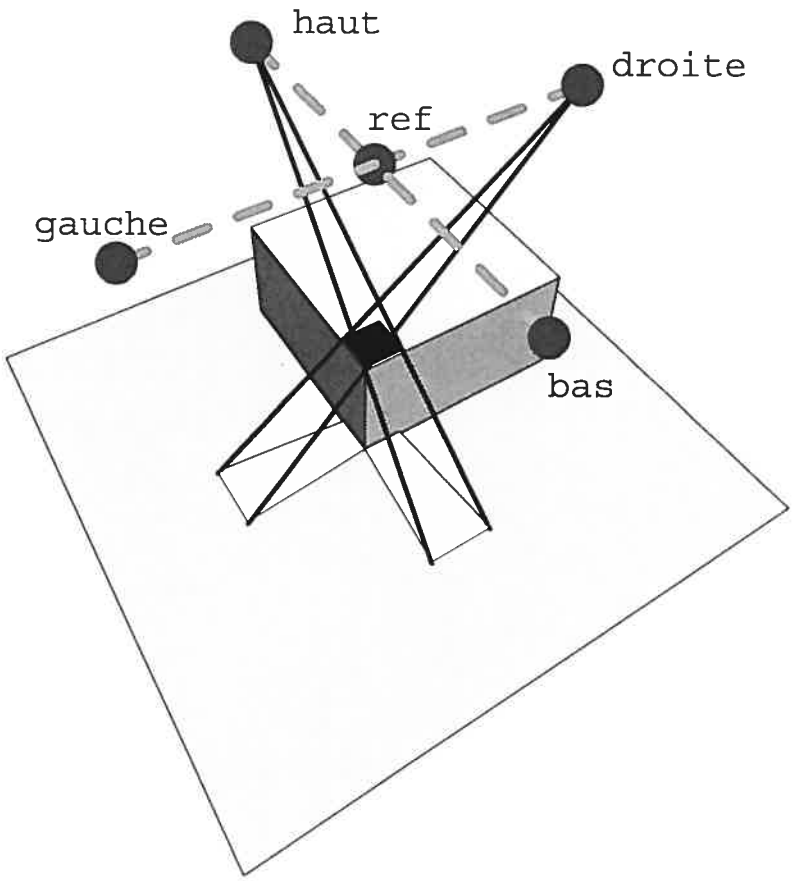

FIG. 4.1 – Exemple de zones d'occultations. La zone sombre est visible dans les 4 caméras de support. Les deux zones claires ne sont pas visibles dans toutes les caméras de support.

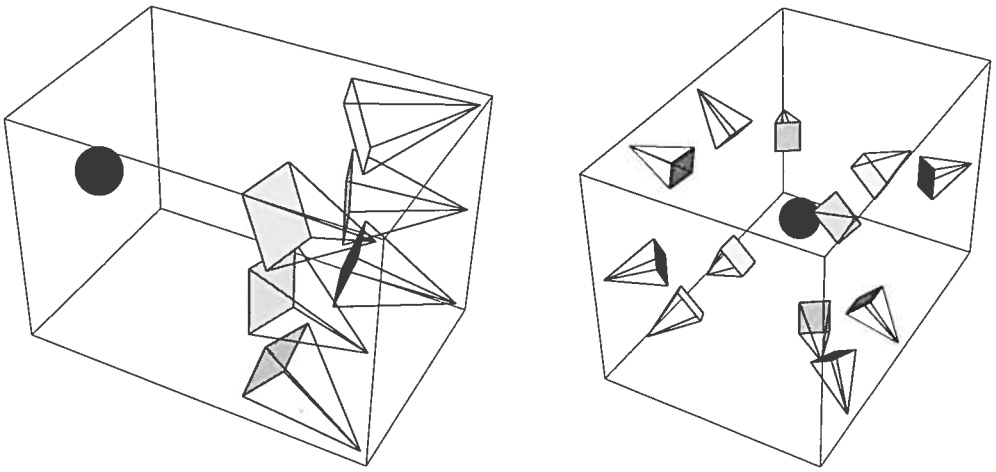

FIG. 4.2 – Configuration de caméras. **Gauche)** Configuration de caméras typique de la reconstruction à caméras multiples de la seconde famille. **Droite)** Configuration de caméras typique de la reconstruction à caméras multiples de la première famille.

Lorsque le problème de la reconstruction stéréoscopique est étendu à plus de deux caméras, il faut associer une profondeur et une liste des caméras visibles à chaque pixel de référence. La prochaine section traitera de l'intégration de la gestion des occultations à la stéréoscopie à images multiples.

4.2 Stéréoscopie à images multiples

Les méthodes stéréoscopiques qui produisent une carte de disparités uniquement pour l'une des deux caméras sont prépondérantes en reconstruction tridimensionnelle binoculaire. En reconstruction à caméras multiples, ce n'est plus nécessairement le cas. Il existe deux familles de méthodes de stéréoscopie à images multiples. La première permet d'obtenir un modèle tridimensionnel complet de l'objet. Dans ce cas, les caméras entourent généralement la scène (voir figure 4.2-droite). Une étude récente compare les différents algorithmes [97] de ce type. La deuxième famille de méthodes permet de reconstruire une carte de disparités uniquement du point de vue d'une

caméra de référence. C'est une généralisation directe de la stéréoscopie binoculaire introduite précédemment. Cette dernière famille de méthodes de stéréoscopie à images multiples sera étudiée. Les caméras sont généralement toutes du même côté de la scène (voir figure 4.2-gauche). Avec ce type de configuration des caméras, il est raisonnable de formuler l'hypothèse suivante : chacun des pixels de référence correspond à un pixel dans au moins une des images de support (voir figure 4.1). Il est donc possible de retrouver une profondeur pour chacun des pixels de référence. Cette hypothèse est particulièrement plausible lorsqu'on utilise des configurations pour lesquelles il y a au moins une caméra de support de chacun des côtés de la caméra de référence (voir figure 4.1). Une configuration en croix avec la caméra de référence au centre et les quatre caméras de support situées à égale distance de celle-ci sera utilisée. Les images ont été obtenues à l'aide d'un bras robotisé. Elles possèdent les propriétés décrites à la section 2.2.2 du chapitre 2. Une configuration en "L" sera également utilisée. Par exemple, dans la figure 4.1, la configuration du type "L" comprend la caméra de référence ainsi que les caméras du haut et de droite.

Il faut maintenant associer à chacun des pixels de référence une disparité et un masque des caméras de support. Par exemple, le masque $(0,\ldots,0)$ signifie qu'aucune caméra de support n'est utilisée alors que le masque $(1,\ldots,1)$ signifie que toutes les caméras de support sont utilisées. Nous nommons $\mathcal{M}$ l'ensemble de tous les masques possibles ; une $\mathcal{M}$-configuration $g : \mathcal{P} \mapsto \mathcal{M}$ associe un masque à chacun des pixels de référence. La fonction d'énergie avec masque devient

$$E(f,g) = \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} e(\mathbf{p}, f(\mathbf{p}), g(\mathbf{p}))}_{\textit{vraisemblance}} + \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} \sum_{\mathbf{r}\in\mathcal{N}_\mathbf{p}} s(\mathbf{p}, \mathbf{r}, f(\mathbf{p}), f(\mathbf{r}))}_{\textit{lissage}} . \tag{4.5}$$

Le terme de vraisemblance est redéfini comme étant

$$e(\mathbf{p}, d, \mathbf{m}) = \frac{\mathbf{m} \cdot C(\mathbf{p}, d)}{|\mathbf{m}|} \quad \text{for } \mathbf{p} \in \mathcal{P},\ d \in \mathcal{D},\ \mathbf{m} \in \mathcal{M} . \tag{4.6}$$

$C(\mathbf{q}, d') = (c_1(\mathbf{q}, d'), \ldots, c_N(\mathbf{q}, d'))$ est le vecteur de fonction de coût du point 3D formé par le point $\mathbf{q}$ et la disparité d' pour chacune des caméras de support. Nous

utilisons $|\mathbf{m}|$ pour représenter la norme l_1 qui est le nombre de caméras utilisées dans le masque $\mathbf{m}$. Certains algorithmes que nous présenterons utilisent l'hypothèse que $|\mathbf{m}|$ n'est jamais égale à zéro. D'autres ajoutent un coût d'occultation similaire à celui introduit dans l'équation 4.4.

Il est possible d'utiliser les masques pour représenter l'état de visibilité V. Une valeur de 1 pour un élément du masque associé au pixel $\mathbf{p}$ signifie que le point 3D formé par le pixel $\mathbf{p}$ et la disparité $f(\mathbf{p})$ est visible par la caméra de support. Une valeur de 0 signifie que la caméra est invisible. Un algorithme effectue un traitement *exact* de la visibilité lorsque

$$g(\mathbf{p}) = V(\mathbf{p}, f(\mathbf{p}), f)$$

pour tous les pixels de référence $\mathbf{p}$. Puisque le masque d'un pixel peut dépendre de la disparité de plusieurs autres pixels, il devient difficile de minimiser la fonction d'énergie précédente et de garantir un traitement *exact* de la visibilité. Néanmoins, Kolmogorov et Zabih ont proposé une méthode traitant *exactement* de la visibilité en utilisant les coupes dans les graphes [59]. La méthode qui a été proposée par Sun *et al.* alterne entre la minimisation de la carte de disparités et celle des masques de visibilité [102]. Cette méthode utilise des configurations binoculaires.

Une façon de simplifier le problème est de pré-sélectionner les masques de visibilité en utilisant des heuristiques et de manière indépendante, résoudre la disparité. Nakamura *et al.* ont démontré expérimentalement que le nombre de masques plausibles pour une configuration de caméras donnée est relativement petit [79]. En utilisant une configuration de 9 caméras placées dans une grille 3×3 et pour 2 scènes différentes, ils ont calculé pour les pixels en état d'occultation, la probabilité d'occurrence de chacun des masques de visibilité. Seulement 5 masques [†] permettent de représenter 84.23% et 97.40% des masques d'occultation observés dans les deux scènes. Les masques de

[†] en ajoutant les masques obtenus par des rotations.

visibilité sont obtenus en minimisant pour tous les $\mathbf{p}$

$$g^*(\mathbf{p}) = \arg \min_{m \in \mathcal{M}_p} e(\mathbf{p}, f(\mathbf{p}), m)\, w(m).$$

$\mathcal{M}_p$ est l'ensemble des masques plausibles et $w(m)$ permet de pondérer pour favoriser certains masques au détriment des autres. Sans cette pondération, le masque utilisant toutes les caméras de support ne serait jamais sélectionné et un masque utilisant uniquement une caméra de support serait toujours sélectionné. Ceci est une conséquence de l'équation 4.6 et du minimum d'un ensemble qui est toujours plus petit que la moyenne des éléments de celui-ci. Une fois les masques pré-calculés $E(f, g^*)$ est minimisé en f. Ceci est équivalent à résoudre un problème de stéréoscopie sans modélisation des occultations. Dans ce type d'approche, l'hypothèse prévoit que le masque permettant d'obtenir le plus petit terme de vraisemblance pondéré est nécessairement celui dont la visibilité est *correcte*. Comme il sera démontré dans le chapitre 6, cette hypothèse n'est pas toujours vraie. Plusieurs variantes de ce type d'heuristiques ont été utilisées dans différents algorithmes de stéréoscopie à caméras multiples [79, 53, 92, 86, 87].

4.3 Contribution à la mise en correspondance passive

Dans les prochains chapitres, nous examinerons différentes méthodes pour obtenir des solutions minimisant le recours aux heuristiques. Nous tenterons de trouver des solutions de faible énergie à l'équation 4.5 et soumise à la contrainte

$$g(\mathbf{p}) \leq V(\mathbf{p}, f(\mathbf{p}), f) \tag{4.7}$$

pour tous les $\mathbf{p}$. Nous nommons cette inéquation *contrainte de géo-cohérence*. Une carte de disparités qui respecte l'inéquation précédente est appelée *géo-cohérente*. Pour certaines configurations de caméras et certaines méthodes, nous relaxerons la *contrainte de géo-cohérence* en l'appliquant uniquement à un sous-ensemble des caméras de support. Les caméras de support seront séparées en 2 ensembles. Pour

l'ensemble $\mathcal{C}_e$ la visibilité sera traitée de manière *exacte* alors que pour le second $\mathcal{C}_h$ la visibilité sera *heuristique*. À partir des caméras $\mathcal{C}_h$, il est possible de pré-calculer un ensemble de masques $\mathcal{M}_h$. Les caméras de l'ensemble heuristique sont utilisées uniquement lorsqu'aucune des caméras de l'autre ensemble n'est visible. Lorsqu'une caméra de l'ensemble *exact* est visible, le masque de visibilité qui en résulte est automatiquement *géo-cohérent* parce qu'il n'utilise pas les caméras de l'ensemble heuristique. Ces méthodes sont donc *hybrides*. Elles se situent entre celles qui sont dites *heuristiques* et *géo-cohérentes*.

4.4 Contrainte d'ordre

La contrainte d'ordre stipule que lorsqu'on parcourt une ligne épipolaire, l'ordre dans lequel on rencontre deux points visibles dans les deux images demeure le même (voir la figure 4.3-gauche). Cette contrainte est vraie dans la majorité des scènes, mais elle est cependant brisée dans certaines scènes (voir la figure 4.3-droite). Imposer la contrainte d'ordre est une forme de régulation qui dépend de la configuration des caméras ainsi que de la variation de la disparité maximale entre différents points le long d'une ligne épipolaire. Lorsque la contrainte d'ordre est brisée, cela signifie que les deux points ne peuvent plus faire partie d'un même objet relié par un maillage continu le long de la ligne épipolaire (voir la figure 4.4-gauche). Imposer la contrainte d'ordre équivaut à garantir que la représentation en maillage continu le long de la ligne épipolaire demeure une hypothèse plausible, compte tenu de la configuration des caméras. Le lissage (tel que présenté au début de ce chapitre) est une forme de régulation basée sur l'observation à l'effet que deux pixels adjacents devraient avoir la même disparité. Ce type de régulation est appliqué aux images alors que l'imposition de la contrainte d'ordre est une forme de régulation géométrique qui dépend simultanément de la géométrie de la scène et de la configuration des caméras.

Il est possible d'appliquer la contrainte d'ordre uniquement sur les masques de

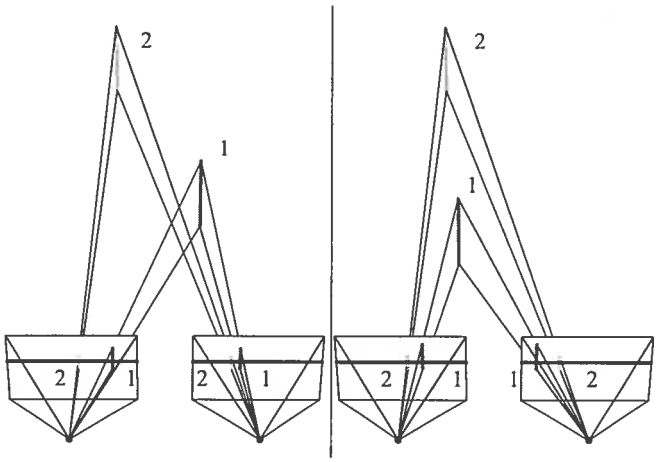

FIG. 4.3 – Illustration de la contrainte d'ordre. **Gauche)** Bris de la contrainte. **Droite)** Respect de la contrainte.

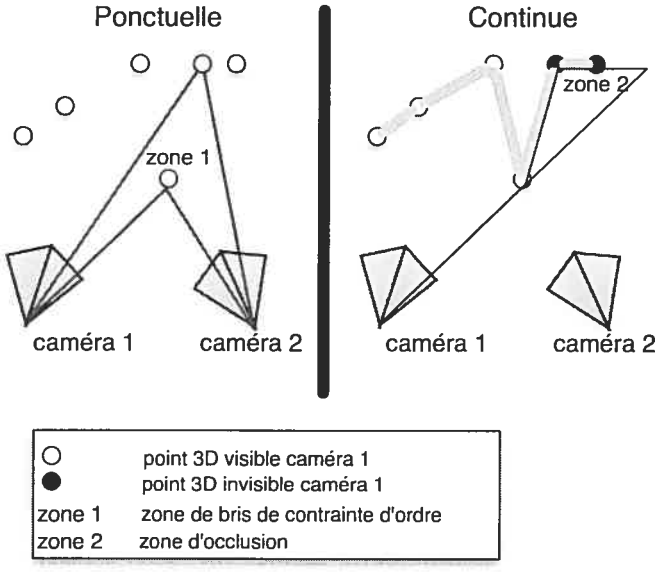

FIG. 4.4 – Représentation en maillage. **Gauche)** Représentation ponctuelle de la carte de disparités avec bris de la contrainte d'ordre. **Droite)** Représentation continue de la carte de disparités.

visibilité g. Il s'agit d'une manière déguisée d'appliquer la contrainte de *géo-cohérence*. Selon le type de maillage utilisé, le calcul de la visibilité sera différent. On a donc deux visibilités dont l'une est continue (V_c) et l'autre ponctuelle (V_p). Si une solution est *géo-cohérente* lorsqu'on considère un maillage continu, elle l'est nécessairement si on considère un maillage ponctuel étant donné que pour tous les pixels $\mathbf{p}$ nous avons

$$V_c(\mathbf{p}, f(\mathbf{p}), f) \leq V_p(\mathbf{p}, f(\mathbf{p}), f).$$

La comparaison des vecteurs est faite élément par élément (voir la figure 4.4-droite). Il est important de mentionner que l'application de la contrainte d'ordre sur le choix des masques de visibilité n'empêche pas de retrouver une géométrie qui brise la contrainte d'ordre. On se sert de la contrainte de *géo-cohérence* pour repousser le choix du type de maillage devant être utilisé pour représenter la carte de disparités. L'avantage d'utiliser une représentation continue est que celle-ci permet un calcul récursif plus efficace de la visibilité lorsque les images sont rectifiées. Ce calcul récursif sera examiné dans les chapitres 6 et 7.

4.4.1 Les méthodes proposées

La première contribution a été publiée dans un article intitulé *Geo-consistence for wide baseline*. Le problème de sur-estimation de la taille des objets et de la mauvaise localisation des bordures de différents algorithmes classiques de stéréoscopie binoculaire [8, 103] y est illustré. La méthode permet d'ajouter une gestion des occultations à une vaste gamme d'algorithmes de stéréoscopie [8, 103, 89]. Pour ce faire, on trouve une carte de disparités avec les masques de visibilité fixes. Les masques sont par la suite mis à jour avec la nouvelle carte de disparités et une nouvelle carte est estimée. Le concept de *géo-cohérence* est une relaxation de la contrainte de visibilité qui est central à la méthode. La méthode débute à partir d'une solution non *géo-cohérente* et converge progressivement vers une solution *géo-cohérente*. Il existe une similitude entre notre méthode et la méthode vorace d'occupation de l'espace (*Space Carving*)

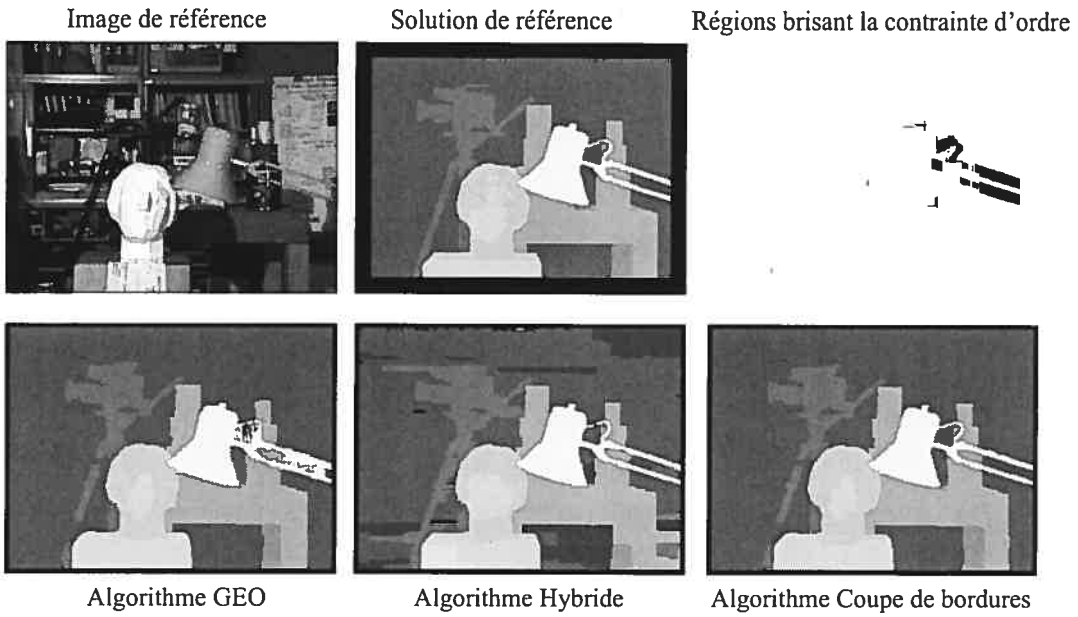

FIG. 4.5 – Résultats sur la séquence de la Tête et de la lampe. Ce jeu de données provient du *Multiview Image Database from the University of Tsukuba*.

[66]. Le retrait d'un voxel vorace est remplacé par un retrait vorace des caméras dans les masques de visibilité. La méthode a pour caractéristique de garantir que la géométrie de la solution finale respecte la contrainte d'ordre entre la caméra de référence et n'importe laquelle des caméras de support. Une nouvelle métrique permettant d'évaluer la qualité de la modélisation des occultations d'un algorithme de stéréoscopie sera également introduite. Une extension de cet algorithme permettant une décomposition non-uniforme de la carte de disparités est présentée dans l'annexe A. Cette dernière permet de combiner le calcul de la visibilité et le raffinement de la carte de disparités.

La seconde contribution a été publiée dans un article intitulé *Fast Multiple-baseline with Occlusion*. Il s'agit d'une application de la programmation dynamique itérative aux configurations à caméras multiples [68]. La méthode est *hybride*. Elle est rapide et peut être adaptée pour utiliser plusieurs processeurs. La méthode telle que

présentée impose la contrainte d'ordre dans le choix des masques de visibilité, mais non dans la géométrie de la scène. Il est possible de ne pas appliquer la contrainte d'ordre sur les masques de visibilité, mais ceci augmente la complexité analytique de l'algorithme. La méthode peut servir d'algorithme auxiliaire dans les zones brisant la contrainte d'ordre. La première méthode peut les détecter, mais ne peut les reconstruire correctement. Cette intégration des deux méthodes est présentée dans un rapport technique [16].

La troisième contribution a été publiée dans un article intitulé *Improving Border Localization of Multi-Baseline Stereo Using Border-Cut.* Le problème de localisation des bordures est séparé du problème d'estimation de la disparité. Tel qu'il sera démontré expérimentalement dans le prochain chapitre, les algorithmes de stéréoscopie classiques produisent des cartes de disparités possédant généralement des bordures mal définies. Par contre les disparités de chacun des côtés de celles-ci sont généralement bien estimées. Il est possible de reformuler le problème afin d'associer une position à chacune des discontinuités plutôt qu'une disparité à chacun des pixels. L'algorithme nécessite une carte initiale de disparités. Il ne s'agit donc pas d'un algorithme de stéréoscopie, mais d'un algorithme de post-traitement. Le voisinage 4 ou 8-connectés entre les pixels voisins d'une carte de disparités est remplacé par un voisinage 2-connectés dans la carte de discontinuités permettant d'utiliser la programmation dynamique. Cette dernière permet de calculer efficacement la visibilité sur 3 des 4 côtés de la caméra de référence. Avec certaines configurations de caméras la visibilité est traitée *exactement* alors que pour d'autres, elle l'est de manière *hybride.*

Le tableau 4.1 présente les caractéristiques des trois nouveaux algorithmes proposés. Les trois méthodes proposées sont identifiées par les termes GEO, Hybrid et Coupe de bordures. La figure 4.5 illustre les cartes de disparités, la solution de référence ainsi que l'image de référence. Les régions brisant la contrainte d'ordre sont également affichées. Notons que pour la solution obtenue par l'algorithme GEO, un post-traitement a été appliqué aux régions suspectées de briser la contrainte d'ordre.

Ce dernier est décrit au chapitre 5. La carte de disparités obtenue par l'algorithme GEO a été utilisée comme initialisation par l'algorithme de coupe de bordures. Les prochains chapitres présenteront le détail de ces différentes méthodes.

configuration de caméras	**Algorithmes**							
	GEO (Chapitre 5)	Hybride (Chapitre 6)			Coupe de bordures (Chapitre 7)			
	croix	2 cam.	L cam.	croix	2 cam.	L cam.	croix	
contrainte d'ordre (géométrie)	oui	non			non			
contrainte d'ordre (masque de visibilité)	oui	oui			oui			
convergence garantie	oui	oui		non	oui		non	
nombre d'itérations (dans nos expériences)	4 à 8	1 à 8			1 à 2			
complexité d'une itération	dépendant de l'algorithme de stéréo utilisé.	$\Theta(N\,D^2)$ [a]			dépend du nombre de discontinuités.			
temps de calcul sur la Tête et de la lampe Tsukuba [b]	environ 20 min.	2.3 sec.			14 sec.			
pourcentages d'erreur sur la Tête et la lampe Tsukuba [c]	2.23%	1.67%			1.05%			

TAB. 4.1 – Caractéristiques des méthodes passives proposées

[a] Il est possible de ne pas appliquer la contrainte d'ordre ; la complexité devient alors $\Theta(N\,D^3)$ avec N qui est le nombre de pixels et D qui est le nombre de pas de disparités.

[b] Image de référence d'une résolution de 384 × 288 avec 16 pas de disparités. Les temps d'exécution ont été calculés sur un AMD Athlon 64 2.0 Ghz.

[c] Selon la méthodologie de l'étude comparative de Middlebury [94].

Chapitre 5

GEO-CONSISTENCY FOR WIDE MULTI-CAMERA STEREO

Cet article [17] a été publié comme l'indique la référence bibliographique.

Marc-Antoine Drouin, Martin Trudeau and Sébastien Roy, Geo-consistency for Wide Multi-Camera Stereo, *International Conference on Computer Vision and Pattern Recognition (CVPR'05)*, San-Diego, USA, June 2005, pages 351-358

Abstract

This paper presents a new model to overcome the occlusion problems coming from wide baseline multiple camera stereo. Rather than explicitly modeling occlusions in the matching cost function, it detects occlusions in the depth map obtained from regular efficient stereo matching algorithms. Occlusions are detected as inconsistencies of the depth map by computing the visibility of the map as it is reprojected into each camera. Our approach has the particularity of not discriminating between occluders and occludees. The matching cost function is modified according to the detected occlusions by removing the offending cameras from the computation of the matching cost. The algorithm gradually modifies the matching cost function according to the history of inconsistencies in the depth map, until convergence. While two graph-theoretic stereo algorithms are used in our experiments, our framework is general enough to be applied to many others. The validity of our framework is demonstrated using real imagery with different baselines.

5.1 Introduction

The goal of binocular stereo is to reconstruct the 3D structure of a scene from two views. As the baseline gets wider, the problem of occlusion, wich is often considered negligible with small baseline configurations, can become severe and limit the quality of the obtained depth map. Occlusion occurs when part of a scene is visible in one camera image but not in the other (see figure 5.1). The difficulty of detecting occlusion comes from the fact that it is induced by the 3D structure of the scene, which is unknown until the correspondence is established, as it is the final goal of the algorithm. We propose a novel multiple camera stereo algorithm that relies on photometric and geometric inconsistencies in the depth map to detect occlusions. As this algorithm is iterative, it does not explicitly model an occlusion state or add extra constraints to the cost function. This makes it possible to use a standard efficient algorithm during each iteration, instead of tackling a very difficult optimization problem. Futhermore, our approach guarantees to preserve the consistency between the recovered visibility and geometry, a property we call geo-consistency. In this paper, the maximum flow [89] and graph cut [8] formulations are used to solve each iteration. Our framework is general enough to be used with many other stereo algorithms. A survey paper by Scharstein and Szeliski [94] compares various standard algorithms.

The rest of this paper is divided as follows: in Section 5.2, previous work will be presented. Section 5.3 describes occlusion modeling and geometric inconsistency. Our proposed algorithm is described in Section 5.4. Experimental results are presented in Section 5.5.

5.2 Previous work

In a recent empirical comparison of strategies to overcome occlusion for 2 cameras, Egnal [20] enumerates 5 basic ones: left-right checking, bimodality test, goodness Jumps constraint, duality of depth discontinuity and occlusion, and uniqueness constraint.

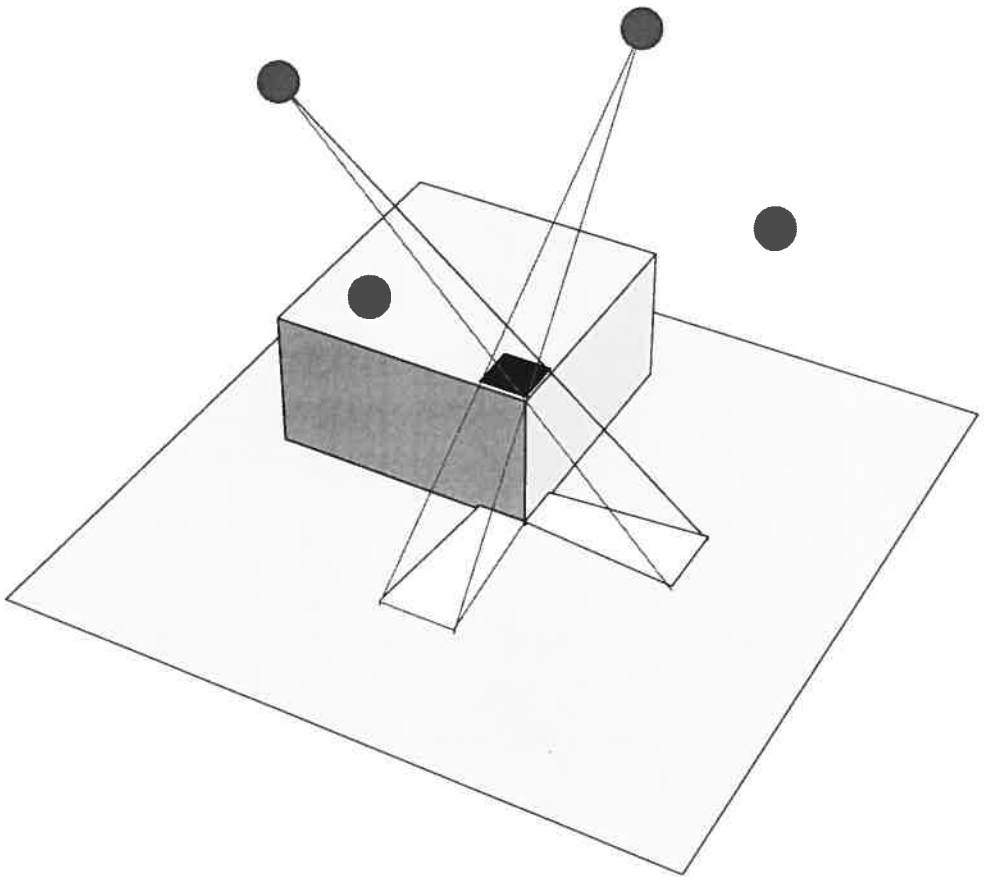

Figure 5.1: Example of occlusion. Occluded pixels appear in white, occluders in black.

Some algorithms that have been proposed rely on one or more of these strategies, and are often based on varying a correlation window position or size [52, 31, 112, 53]. These methods are binocular in nature and do not generalize well to the case of multiple arbitrary cameras. Other algorithms use dynamic programming [80, 47, 14] because of its ability to efficiently solve more complex matching costs and smoothing terms. Two methods using graph theoretical approaches [49, 57] have been proposed, but again they do not generalize to multiple camera configurations.

When extending binocular stereo to multiple cameras, the amount of occlusion increases since each pixel of the reference camera can be hidden in more than one supporting camera. This is particularly true when going from a single to a multiple-baseline configuration, such as a regular grid of cameras [79]. Some researchers have proposed specially designed algorithms to cope with occlusion in multiple camera configurations. Amongst these, Kang et al. [53] proposed a visibility approach. While they did not improve over adaptive windows, their scheme was based on the hypothesis that a low matching cost function implies the absence of occlusion. This hypothesis is also made in [79, 92, 86, 87]. In contrast, we do not rely on such an assumption. In [116], a relief reconstruction approach based on belief propagation is presented where the correct visibility is approximated by using a low resolution base surface obtained from manually established correspondences. In [66, 98], *visibility-based* methods are introduced. The matching cost incorporates the visibility information into a photo-consistency matching criteria, thereby implicitly modeling occlusion in the reconstruction process. Our method differs completely in the way it handles smoothing and by its ability to recover from bad "carving". Similarly, a level-set method [23] uses the visibility information from the evolving reconstructed surface to explicitly model occlusion. In [59] a stereo algorithm based on graph cuts is presented. It strictly enforces visibility constraints to guide the matching process and ensures that it does not contain any geometric inconsistencies. The formulation imposes strict constraints on the form of the smothing term, constraints that will not

apply to our method as we will see.

5.3 *Modeling occlusion and geo-consistency*

We have a set of reference pixels $\mathcal{P}$, for which we want to compute depth, and a set of depth labels $\mathcal{Z}$. A $\mathcal{Z}$-configuration $f : \mathcal{P} \mapsto \mathcal{Z}$ associates a depth label to every pixel. When occlusion is not modeled, the energy function to minimize is

$$E(f) = \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} e(\mathbf{p}, f(\mathbf{p}))}_{\substack{\textit{pointwise} \\ \textit{likelihood}}} + \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} \sum_{\mathbf{r}\in\mathcal{N}_\mathbf{p}} s(\mathbf{p}, \mathbf{r}, f(\mathbf{p}), f(\mathbf{r}))}_{\textit{smoothing}} \tag{5.1}$$

where $\mathcal{N}_\mathbf{p}$ is a neighborhood of pixel $\mathbf{p}$. This can be solved efficiently because the likelihood term $e(\mathbf{p}, f(\mathbf{p}))$ is independent from $e(\mathbf{p}', f(\mathbf{p}'))$ for $\mathbf{p} \neq \mathbf{p}'$, and the smoothing term has a simple 2-site clique form.

To model occlusion, we must compute the volumetric visibility $V_i(\mathbf{q}, f)$ of a 3D reference point $\mathbf{q}$ from the point of view of a camera i, given a depth configuration f. It is set to 1 if the point is visible, and 0 otherwise. Visibility is a long range interaction and knowledge about immediate neighborhood configuration is insufficient most of the time for computing it. The visibility information is collected into a vector, the *visibility mask*

$$V(\mathbf{q}, f) = (V_1(\mathbf{q}, f), \ldots, V_N(\mathbf{q}, f))$$

where N is the number of cameras outside the reference; a vector $(1, \ldots, 1)$ means that the 3D point is visible in all supporting cameras, $(0, \ldots, 0)$ that it is invisible instead. We call $\mathcal{M}$ the set of all possible visibility masks; an $\mathcal{M}$-configuration $g : \mathcal{P} \mapsto \mathcal{M}$ associates a mask to every pixel. Using this, we transform Eq. 5.1 into an energy function with mask

$$E(f, g) = \sum_{\mathbf{p}\in\mathcal{P}} e(\mathbf{p}, f(\mathbf{p}), g(\mathbf{p})) + \textit{smoothing}. \tag{5.2}$$

Typically, we define

$$e(\mathbf{p}, z, \mathbf{m}) = \frac{\mathbf{m} \cdot C(\mathbf{p}|z)}{|\mathbf{m}|} \quad \text{for } \mathbf{p} \in \mathcal{P},\ z \in \mathcal{Z},\ \mathbf{m} \in \mathcal{M}$$

where the 3D point $\mathbf{p}|z$ is $\mathbf{p}$ augmented by z and $C(\mathbf{q}) = (C_1(\mathbf{q}), \ldots, C_N(\mathbf{q}))$ is the vector of matching costs of 3D point $\mathbf{q}$ for each camera. We use $|\mathbf{m}|$ to represent the l_1-norm which is just the number of cameras used from $\mathbf{q}$. The case where $|\mathbf{m}| = 0$ is discussed in section 5.4.2. A simple cost function is $C_i(\mathbf{q}) = (I_{ref}(\mathbf{M}_{ref}\mathbf{q}) - I_i(\mathbf{M}_i\mathbf{q}))^2$ where $\mathbf{M}_{ref}$ and $\mathbf{M}_i$ are projection matrices from the world to the images of camera *ref* and i respectively, and I_{ref} and I_i are these images. Now, in order to model occlusion properly, we simply need to examine the case $g(p) = V(\mathbf{p}|f(\mathbf{p}), f)$.

If the visibility masks were already known and fixed, the occlusion problem would be solved and only photometric ambiguity would remain to be dealt with; the energy function (5.2) would then be relatively easy to minimize. Since this is not the case and f and $V(., f)$ are dependent, we relax the problem by introducing the concept of *g*eo-consistency: we say that a $\mathcal{Z}$-configuration f is geo-consistent with an $\mathcal{M}$-configuration g if

$$g(\mathbf{p}) \leq V(\mathbf{p}|f(\mathbf{p}), f) \tag{5.3}$$

for each component of these vectors and all $\mathbf{p} \in \mathcal{P}$. The inequality thus allows the mask to contain a subset of the visible cameras. The removal of extra cameras has been observed to have little impact on the quality of the solution [79]. Our problem becomes the minimization of Eq. 5.2 in f and g, with the constraint that f is geo-consistent with g.

5.3.1 Solving simultaneously for depth and visibility

Lets define $g^0(\mathbf{p}) = (1, \ldots, 1)$ for all $\mathbf{p} \in \mathcal{P}$; this corresponds to the case where all cameras are visible by all points. Minimizing $E(f, g^0)$ in f is equivalent to minimizing $E(f)$. In general, it is possible to minimize $E(f, g)$ by explicitly testing all combinations of depth labels and visibility masks in $\mathcal{Z} \times \mathcal{M}$. Since $\#\mathcal{M} = 2^N$, this

effectively makes the problem too big to be solved except in the simplest cases. One way to reduce the number of visibility masks is to realize that for a given camera configuration, some masks may occur for no configuration f. This makes it possible to precompute a smaller subset of $\mathcal{M}$. Another way to reduce the number of masks is simply to decide on a *reasonable* subset to use [79]. Unfortunately, even with a small number of masks, it is still not practical to minimize in f and g simultaneously. We can however use photo-consistency alone to select the visibility mask of a pixel, if it is assumed equivalent to geo-consistency. In order to determine the mask for a pixel $\mathbf{p}$ at depth $f(\mathbf{p})$, we can try each mask and select the most photo-consistent one, i.e. we define g_f^* as

$$g_f^*(\mathbf{p}) = \arg\min_{m \in \mathcal{M}} e(\mathbf{p}, f(\mathbf{p}), m)\, w(m)$$

where $w(m)$ is a weight function favoring g^0 and eliminating improbable masks. The problem thus becomes the minimization of $E(f, g_f^*)$ in f. Since e is point-wise independent, the new problem is reduced to the original formulation of Eq. 5.1 and is easily solved using standard algorithms. This technique is used in [79, 92, 86]. However, the selected masks are not guaranteed to preserve geo-consistency.

In space carving [66], the depth $f(\mathbf{p})$ of a pixel is increased at a given step if it is not photo-consistent (which is determined using a threshold). When depth is changed at a point, the mask configuration g is updated accordingly, and so preserves geo-consistency. Space carving is a greedy algorithm that solves Eq. 5.2 subject to the constraint of Eq. 5.3 without smoothing. Kolmogorov and Zabih [59] tried to minimize an approximation of Eq. 5.2 subject to the constraint of Eq. 5.3 with spatial smoothing by moving iteratively from one geo-consistent solution to the other.

5.4 Stereo with a new implicit occlusion model

We propose to reduce the dependency between f and g by making it *temporal*: we let f^0 be the $\mathcal{Z}$-configuration minimizing $E(f, g^0)$ in f and for $t > 0$, we define iteratively

f^t as the function minimizing

$$\sum_{\mathbf{p}\in\mathcal{P}} e(\mathbf{p}, f^t(\mathbf{p}), V(\mathbf{p}|f^t(\mathbf{p}), f^{t-1}) + \textit{smoothing} \tag{5.4}$$

and g^t as

$$g^t(\mathbf{p}) = V(\mathbf{p}|f^t(\mathbf{p}), f^{t-1}),$$

that is to say, f^t minimizes $E(f^t, g^t)$, where g^t depends on f^t according to the above equation. Now, this can be done using any standard algorithm. Unfortunately, this process does not always converge [53].

5.4.1 Using history for convergence

Because of the way g^t is defined, cameras that are removed at one iteration can be kept at the next, possibly introducing cycles. To guarantee convergence, we introduce a *visibility history mask* independent of the matching cost function value

$$H(\mathbf{q}, t) = (H_1(\mathbf{q}, t), \ldots, H_N(\mathbf{q}, t))$$

where N is again the number of cameras other than the reference and

$$H_i(\mathbf{q}, t) = \prod_{0\leq k\leq t} V_i(\mathbf{q}, f^k) = \min_{0\leq k\leq t} V_i(\mathbf{q}, f^k). \tag{5.5}$$

The new problem is obtained by substituting H for V in Eq. 5.4 to obtain

$$E^t_H(f^t) = \sum_{\mathbf{p}\in\mathcal{P}} e(\mathbf{p}, f^t(\mathbf{p}), H(\mathbf{p}|f^t(\mathbf{p}), t-1) + \textit{smoothing} \tag{5.6}$$

Mutatis mutandis, f^t now minimizes $E^t_H(f^t)$ and $g^t(\mathbf{p}) = H(\mathbf{p}|f^t(\mathbf{p}), t-1)$. This iterative process always converges (or stabilizes) in a polynomial number of steps. Indeed, $H(\mathbf{q}, t)$ is monotonically decreasing in t for all $\mathbf{q}$; moreover, if $H(\mathbf{q}, t-1) = H(\mathbf{q}, t)$ for all $\mathbf{q}$, then $f^t = f^{t+1}$ since both are solutions to the same minimization problem, and the process has stabilized. We see that the number of iterations is bounded by $N \cdot \#\mathcal{P} \cdot \#\mathcal{Z}$.

Furthermore, after convergence, the final configuration $f^{T+1} = f^T$ is geo-consistent with g^{T+1}; this comes from the fact that for all $\mathbf{p}$:

$$\begin{aligned} g^{T+1}(\mathbf{p}) &= H(\mathbf{p}|f^{T+1}(\mathbf{p}), T) = H(\mathbf{p}|f^{T}(\mathbf{p}), T) \\ &\leq V(\mathbf{p}|f^{T}(\mathbf{p}), f^{T}) = V(\mathbf{p}|f^{T+1}(\mathbf{p}), f^{T+1}). \end{aligned}$$

We thus have an algorithm that converges to a geo-consistent solution, but that can transit through intermediate ones that are not. This type of behavior differentiates our approach from others that strictly enforce geo-consistency during the optimization process [66, 23, 59].

5.4.2 Pseudo-visibility

For a given f, an occluder $\mathbf{p}|f(\mathbf{p})$ is a 3D point blocking an occludee $\mathbf{p}'|f(\mathbf{p}')$ in some camera. Figure 5.1 illustrates the phenomenon. Each pixel of a depth map can be classified as an occluder, an occludee, or a regular pixel (neither occluder nor occludee). We have observed experimentally that many algorithms have a tendency to overestimate the disparity of occluded pixels. This has the effect of making close objects larger, creating a shift in the pixel classification of occludees and occluders. Occludees have a tendency to be classified as occluders, occluders as regular pixels and regular pixels as occludees (see figure 5.2). To validate this assertion, we used the results of two of the best stereo matchers evaluated with the Middlebury dataset [94, 103, 8]. The method in [8] was ranked the best stereo matcher in two comparative studies [105, 94]. [103] appeared later and achieved an even lower error rate. For each obtained depth map, we computed the percentage of pixels classified as occluder by the depth map that really are occludees and that of pixels classified as occludees that really are regular (table 5.1). Both turned out to be quite high. Since most pixels are regular, the percentage of wrong classification for them is low. Nevertheless, there is a clear bias: more pixels classified as regular are occluders than occludees. The observation above discourages the direct use of visibility to update the visibility

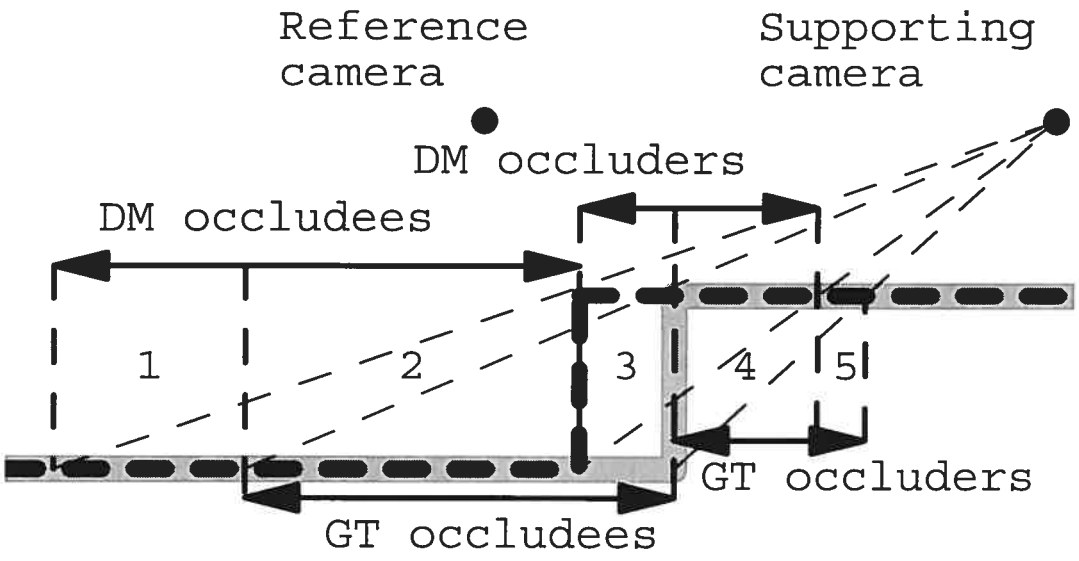

Figure 5.2: Effect of object enlargement on classification of occluders and occludees of a scene viewed by 2 cameras. The ground truth is in thick gray and the depth map in thick dashes. Occluders and occludees are shown for both ground truth (GT) and computed depth map (DM). Illustration of classification shift. Respectively, the 5 zones represent 1) regular pixels wrongly classified as occludees 2) occludees correctly classified 3) occludees wrongly classified as occluders 4) occluders correctly classified 5) occluders wrongly classified as regular.

Algo	Scenes from Middlebury comparative study					
	Tsukuba Head and Lamp			Sawtooth		
	Real status of pixels classified from depth map as occluders					
	occludee	occluder	regular	occludee	occluder	regular
bp [103]	**44.8**	16.3	38.9	**42.6**	3.8	53.6
bnv [8]	**50.4**	15.4	34.2	**42.6**	4.3	53.3
	Real status of pixels classified from depth map as occludees					
	occludee	occluder	regular	occludee	occluder	regular
bp [103]	15.5	5.9	**76.6**	5.5	1.1	**93.4**
bnv [8]	16.4	5.8	**77.8**	7.2	1.1	**91.7**
	Real status of pixels classified from depth map as regulars					
	occludee	occluder	regular	occludee	occluder	regular
bp [103]	1.0	**2.0**	97.0	0.5	**1.5**	98.0
bnv [8]	1.0	**2.0**	96.9	0.5	**1.5**	98.0

Table 5.1: Real (ground truth) status in percentages of pixels according to their classification. Examples from the Middlebury comparative study [94]. In bold are the misclassifications favored by the overestimation of the disparity of occluded pixels.

history mask. Instead, we introduce a pseudo-visibility

$$V'(\mathbf{q}, f) = (V'_1(\mathbf{q}, f), \ldots, V'_N(\mathbf{q}, f))$$

which compensates for the bias by labeling both occluders and occludees as invisible. An obvious consequence of this definition is the fact that

$$V'_i(\mathbf{p}|f(\mathbf{p}), f) \leq V_i(\mathbf{p}|f(\mathbf{p}), f) \quad \forall \mathbf{p} \in \mathcal{P}, \quad 1 \leq i \leq N.$$

The ordering constraint simply states that when scanning an epipolar line, the order in which we encounter two different objects visible in two images of a stereo pair must be the same in the two images (see Fig 5.3-left). This constraint holds for most scenes (see Fig 5.3-right) [65]. While this constraint is broken in some rare cases, it remains a powerful tool when dealing with occlusion and outliers. If we represent the depth map as an opaque mesh, we are guaranteed to preserve the ordering constraint

between the reference and any supporting camera for any point visible from them. If a set of pixels $\mathcal{O}$ breaks the ordering constraint between the reference camera and some supporting image i at iteration t, then according to our definition of pseudo-visibility (and using an opaque mesh), the history mask is updated to $H_i(\mathbf{p}|f^{t+1}(\mathbf{p}),t) = 0$ for all $\mathbf{p}$ in $\mathcal{O}$. After convergence for the final configuration f^T we have for all $\mathbf{p}$ $H(\mathbf{p}|f^{T+1}(\mathbf{p}),T) = H(\mathbf{p}|f^T(\mathbf{p}),T-1)$. In particular $H_i(\mathbf{p}|f^{T+1}(\mathbf{p}),T) = 0$ for all $\mathbf{p} \in \mathcal{O}$. Since the offending camera i was not used to compute the final solution, the ordering constraint is respected between the reference camera and the supporting camera i.

The pseudo-visibility masks V_i' are computed by using rendering techniques. Two renderings of the current depth map f are done from the point of view of each supporting camera i: one with an ordinary Z-buffer and one with a reverse Z-buffer test. Two depth maps L_i^f and G_i^f are thus obtained and contains minimal and maximal depth observed by the camera. By comparing them, we can detect when two points of the mesh project to the same location for a given supporting camera. When using rectified images, this rendering process can be greatly sped up and simplified by replacing it by a line drawing using depth buffers. The pseudo-visibility function $V_i'(\mathbf{q},f)$ can therefore be computed as

$$V_i'(\mathbf{q},f) = \delta\left(L_i^f(\mathbf{T}_i\mathbf{q}) - G_i^f(\mathbf{T}_i\mathbf{q})\right)$$

where δ is 1 at 0 and 0 elsewhere.

It is possible for a voxel to have all its cameras removed, i.e. $H(\mathbf{p}|z,t-1) = \mathbf{0}$ even if $V(\mathbf{p}|z,t-1) \neq \mathbf{0}$ In practice, when this happens, we replace $e(\mathbf{p},z,H(\mathbf{p}|z,t-1))$ by $e(\mathbf{p},f^{t'+1}(p)),H(\mathbf{p}|z,t')$ in the minimization process that computes f^t (see Eq. 5.6), where t' is the largest index such that $H(\mathbf{p}|z,t') \neq 0$. It this case, depth is assigned only from the neighborhood through smoothing.

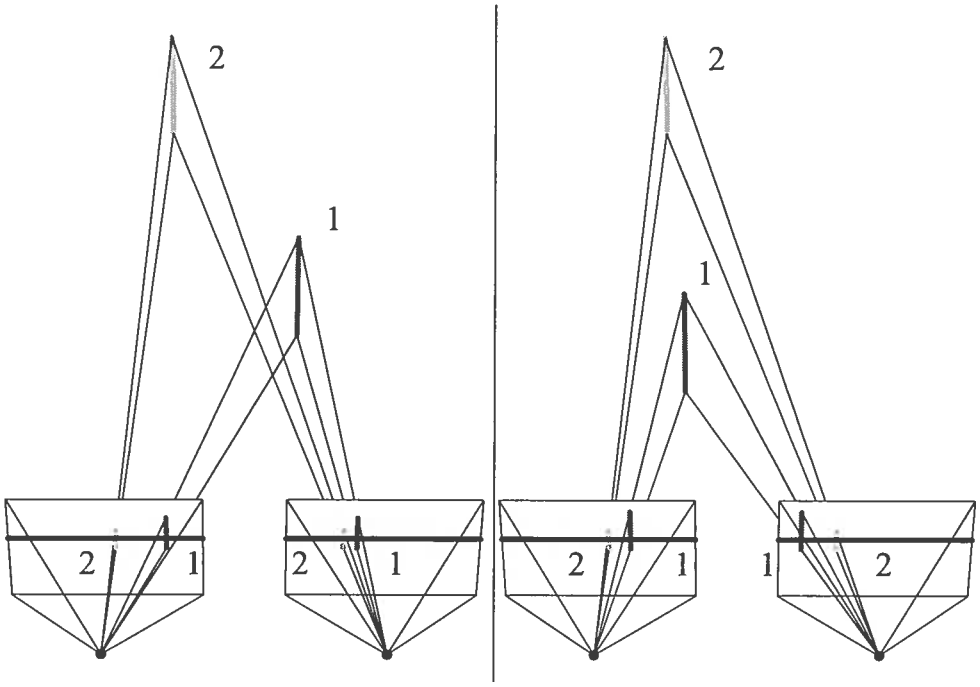

Figure 5.3: Ordering constraint Left) Ordering constraint is satisfied. In this camera configuration, the epipolar lines are parallel to the X-axis. The line 2 is located to the left of the line 1 in both images. Right) Ordering constraint is broken, the line 2 appears to the left of the line 1 in one image and to the right in the other.

5.5 *Experimental results*

In all our experiments, the matching cost function was the same for all algorithms, that of [59] which is based on [4]. We used color images but only the reference images in gray scale are shown here. As for the smoothing term, we used the experimentally defined smoothing function that also comes from [59]:

$$s(\mathbf{p}, \mathbf{r}, f(\mathbf{p}), f(\mathbf{r})) = \lambda\, h(\mathbf{p}, \mathbf{r})\, l(f(\mathbf{p}), f(\mathbf{r}))$$

where h is defined as

$$h(\mathbf{p}, \mathbf{r}) = \begin{cases} 3 & \textit{if } |I_{ref}(M_{ref}\mathbf{p}) - I_{ref}(M_{ref}\mathbf{q})| < 5 \\ 1 & \textit{otherwise} \end{cases}$$

with $l(f(\mathbf{p}), f(\mathbf{r})) = |f(\mathbf{p}) - f(\mathbf{r})|$ for the maximum flow [89] and $l(f(\mathbf{p}), f(\mathbf{r})) = \delta(f(\mathbf{p}) - f(\mathbf{r}))$ for graph cut formulation [8]. The parameter λ is user-defined. For each depth map computation, we chose the λ that achieved the best performance. A pixel disparity is considered erroneous if it differs by more than one disparity step from the ground truth. This error measurement is compatible with the one used in two comparative studies for 2-camera stereo [105, 94, 59].

When minimizing Eq. 5.6, a visibility mask must be kept for every voxel of the reconstruction volume, that is, for each $\mathbf{p} \in \mathcal{P}$ and $z \in \mathcal{Z}$. To reduce memory requirements and the number of iterations, we kept a single visibility history for each pixel p regardless of the disparity z, i.e. (5.5) becomes $H_i(\mathbf{p}, t) = \prod_{0 \leq k \leq t} V_i(\mathbf{p}|f^k(\mathbf{p}), f^k)$. This saves a lot of memory but the convergence is no longer guaranteed. We simply stop iterating when $H(\mathbf{p}, t) = H(\mathbf{p}, t-1)$ for all $p \in \mathcal{P}$. We observed that running the algorithm any longer only produce minor modifications to f^t. However, the number of pixels with final zero masks increases, usually in regions where the ordering constraint is broken. Pixels with zero masks are more prone to error, therefore we tried to improve results by adding a second step that reintroduces eliminated cameras. This step consisted in fixing to their final values the depth labels of the pixels with

Algorithm	Scenes from Middlebury					
	Barn1	Barn2	Bull	Poster	Venus	Sawtooth
FULL-BNV	3.5 %	3.1 %	0.7 %	3.7 %	3.4 %	3.3%
FULL-MF	4.0 %	5.4 %	0.7 %	3.4 %	4.4 %	3.8 %
GEO-BNV	**0.8 %**	**0.6 %**	0.4 %	1.1 %	**2.4 %**	**1.1 %**
GEO-MF	1.5 %	0.9 %	**0.3 %**	1.4 %	3.4 %	1.5 %
KAN-BNV	1.4 %	1.5 %	0.9 %	1.1 %	4.0 %	1.5%
KAN-MF	1.1 %	1.2 %	**0.3 %**	**0.9 %**	5.8 %	2.2 %

Table 5.2: Error percentages for the different scenes of the Middlebury data set. The best performance for each image set is highlighted.

non-zero final camera masks. The history of the others was discarded and the volumetric visibility recomputed, considering only occlusion caused by the fixed pixels. Finally, an additional minimization was run to produce a better depth map.

5.5.1 Middlebury

This datasets from Middlebury [95] consists of 6 series of 9 images of size 434×383. We used images 0 to 7 in our experiments. The disparities between images 2 and 6 range from 0 to 19 pixels and 20 disparity steps were used. Since the ground truth was available for this dataset, we used it to compute error percentages when using the second image as the reference. We compared our method against Nakamura's [79] with a special choice of masks: either all the cameras to the left of the reference are visible or all the cameras to the right are. This specialized version of Nakamura is described in [53, 92]. The abbreviation used for this method is KAN. Our method is denoted by GEO. The results of GEO after one iteration are also shown under the label FULL, since this is a case where no occlusion modeling is made. We used 2 different stereo matchers: maximum flow [89] (MF) and graph cuts [8] (BNV). Results are shown in Table 5.2. While KAN's modeling of occlusion achieves impressive results, our approach using the BNV stereo matcher perform better in 4 of the 6 sequences of

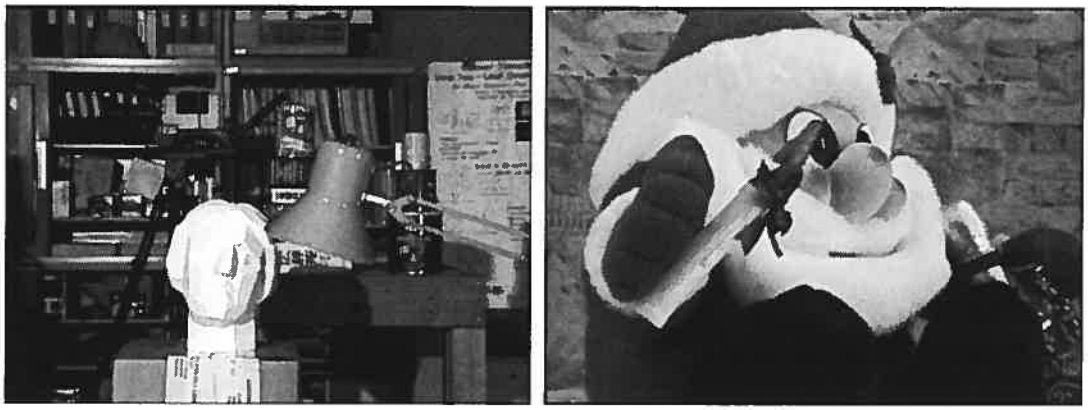

Figure 5.4: Reference images for the Head and Lamp scene (left) and the Santa scene (right) from the Multiview Images database of the University of Tsukuba.

images and were close to KAN in the other two. Oddly enough, in the Venus scene, KAN had a higher error rate than FULL, even though FULL is a simplified version of KAN (a single mask with all the cameras). Our algorithm takes an average of 8 iterations to converge, the improvement after just 4 is minimal.

5.5.2 Tsukuba Head and Lamp

This dataset is from the Multiview Image Database from the University of Tsukuba (see Figure 5.4). It is composed of a 5×5 image grid. Each image has a resolution of 384×288. The search interval is between 0 and 15 pixels and we used 16 disparity steps. We only used 5 images for each depth map computation. The reference image is the center one and the 4 supporting images are at an equal distance from it, arranged in a cross shape. In addition to those of GEO-BNV and GEO-MF, the results of GEO-BNV when using the recovery method described in section 5.5 are shown under the label "GEO-BNV pt". Some depth maps are shown in figure 5.5 and error percentages are shown in table 5.3. The entry KZ1 of the table comes directly from [59]. This method achieved a very low error rate. However, as the authors

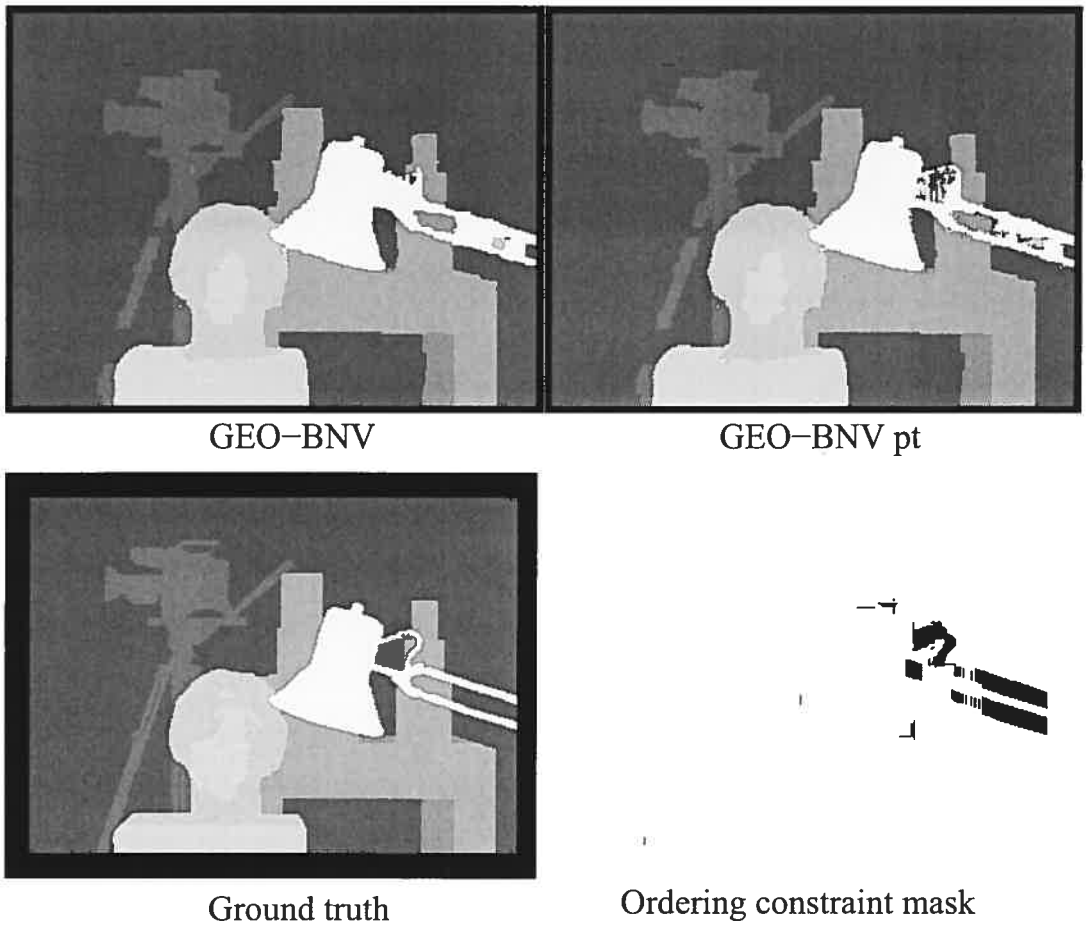

Figure 5.5: Depth maps for the Head and Lamp scene (Multiview Images database of the University of Tsukuba). Note for GEO-BNV how the errors are concentrated in regions breaking the ordering constraint. A mask of pixels breaking the ordering constraint for the smallest baseline is also shown.

Algorithm	Baseline	Error (whole image)	Error (mask)
GEO-BNV pt	1x	2.23%	1.53%
KZ1	1x	2.30%	2.01%
GEO-BNV	1x	2.46%	1.64%
GEO-MF	1x	3.42%	2.52%
GEO-BNV	2x	2.69%	2.11%
GEO-MF	2x	2.62%	1.28%

Table 5.3: Percentages of error of the different algorithms for Head and Lamp scene, using 5 images. The right column contains the amount of error computed after the removal of the pixels breaking the ordering constraint, the left shows it for all the pixels.

mentioned, the algorithm has trouble with low textured regions (the top right corner for instance), therefore the error is somewhat underestimated by the removal of an 18 pixel border in the ground truth. We also computed the error after removing the pixels breaking the ordering constraint, in particular part of the arm of the lamp. The mask was determined by re-projecting the ground truth in each supporting camera, hence it differs for the two baselines.

GEO-BNV almost performed as well as KZ1; when removing pixels breaking the ordering constraint, it achieved a slightly lower error rate. For some algorithms, the error rate decreased for the larger baseline. This counter-intuitive behavior is explained by the fact that the matching cost function in the lamp region is less ambiguous when the baseline is larger. Table 5.4 shows the stability to changes of the smoothing parameter of our algorithm using graph cuts, giving the error percentage for 6 values of this parameter.

5.5.3 Baseline test

As the baseline increases, the amount of occlusion in the scene increases as well. A stereo matcher not affected by occlusion would give identical depth maps for different baselines. To measure the level of resistance to change of the baseline, for

Algorithm	Smoothness parameter					
	1/30	1/10	1	2	3	4
GEO-BNV 1x	2.61	2.67	2.66	2.55	3.53	4.12

Table 5.4: Resistance to change of the smoothing parameter for the Head and Lamp scene. The smoothing parameter increases by a factor of 120, while the error rate varies by less than 1.6% for the small baseline.

the different occlusion overcoming strategies, we introduce the notion of depth map incompatibility. A pixel $\mathbf{p}$ is incompatible in two depth maps i and j if

$$|f_i(\mathbf{p}) - f_j(\mathbf{p})| > 1$$

(a difference of 1 is meaningless as it could be the result of discretization errors). It is important to mention that a low incompatibility level is not necessarily a sign of low error level in the depth map. But the amount of occlusion increases with the baseline, and so should the error and incompatibility levels for stereo matchers that do not model occlusion. To test the stability of our algorithm, we used the Santa scenes from the Multiview Image Database of the University of Tsukuba (see figure 5.4). This dataset contains 81 images in a 9×9 grid and the focal distance of the camera was 10 mm with successive baselines of 20, 40, 60 and 80 mm. We only used 5 images in a cross shape configuration. Images were reduced by a factor of 2 to achieve a resolution of 320×240. Each depth map was computed using 23 disparity steps. Note the details on the right side of the hat and on the candle. Again, for each depth map, the smoothing parameter was adjusted to obtain the best possible performance. Since no ground truth was available, the choice was made by visual inspection of every depth maps.

The figure 5.7 contains bar charts of the percentages of pixels incompatible between the depth maps obtained for two baselines. In addition to GEO-MF and KZ1, results from the Nakamura approach [79, 92] using maximum flow (NAKA-MF) and

graph cuts (NAKA-BNV) were also included. GEO-MF is twice as stable as NAKA-MF and yields less noisy depth maps. KZ1 and NAKA-BNV are less stable by a factor of 5 and more. The results for FULL-MF are again given. We can see in Figure 5.6 that GEO-MF achieves the best results for the third baselines. For the first baseline, KZ1, NAKA-MF and GEO-MF performed similarly. The running times for GEO-MF and GEO-BNV are respectively less than 5 and 9 minutes on a 2.0 GHz AMD Athlon(tm) XP 2600+.

5.6 Conclusion

We have presented a new framework to model occlusion in stereo by introducing geo-consistency. We also provided a way to apply this framework to add occlusion modeling to standard stereo algorithms. Rather than explicitly model occlusion, our iterative approach relies on geo-consistency of depth maps to determine visibility of cameras and to aggressively remove them to adjust the matching cost function to the scene structure and to the bias in the type of error committed by the stereo matcher. One of the main characteristics of our approach is that we do not discriminate between occluders and occludees. Our implicit occlusion model is successful in obtaining sharp and well-located depth discontinuities and allows the use of efficient standard stereo matching algorithms. Moreover, our framework does not add any parameter or constraint to the matching process. The validity of our framework has been demonstrated on standard datasets with ground truth and was compared to other state of the art occlusion models for multiple view stereo. Our approach was also tested on increasingly wider baselines in order to demonstrate its stability to increasing amount of occlusion in the scene. While the validity of our framework has been demonstrated using two stereo matching algorithms, it is general enough to be applied to others. It is not limited to regular grids of cameras and also works with other camera configurations.

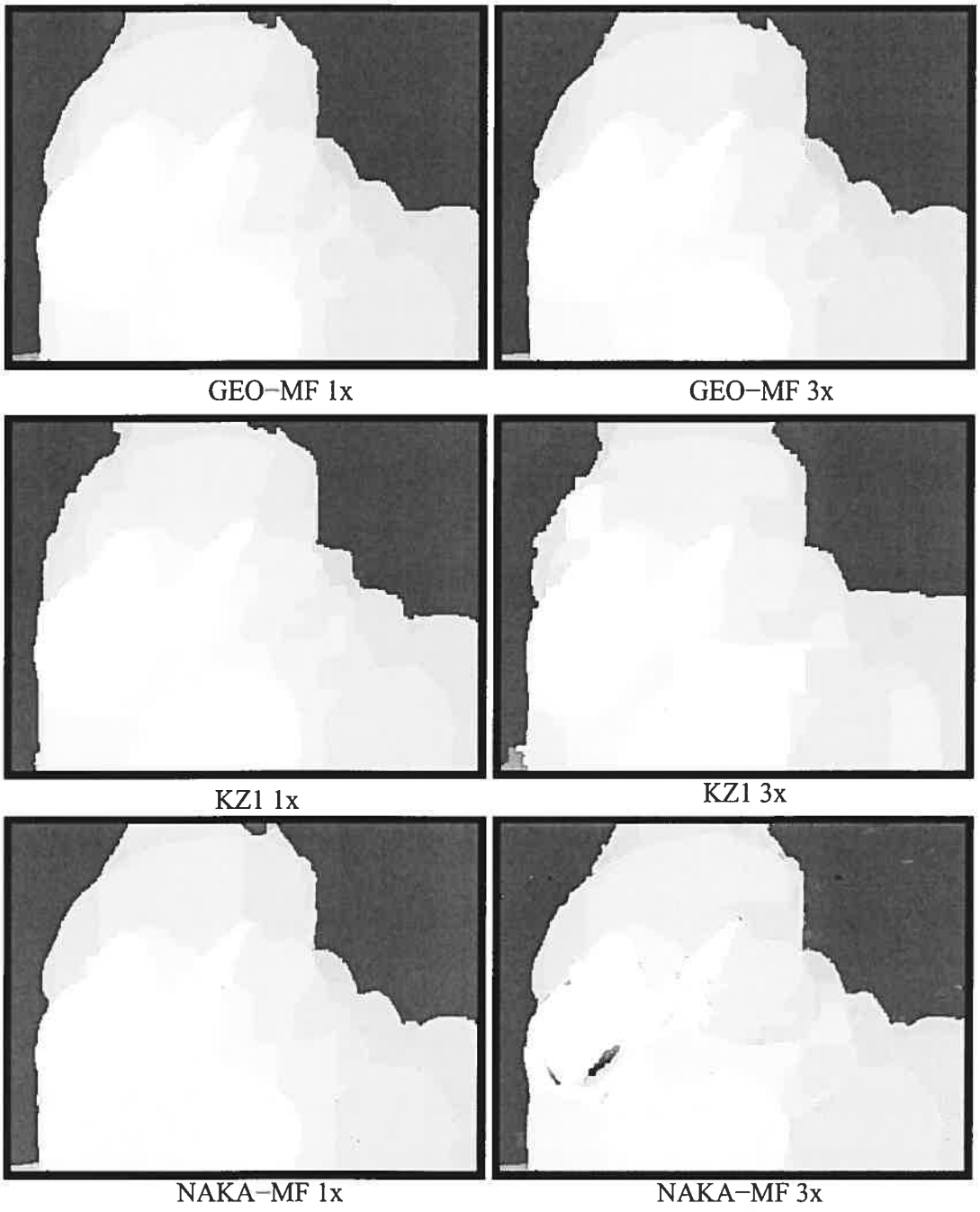

Figure 5.6: Santa scene (Multiview Image Database of the University of Tsukuba). Depth maps obtained by 3 algorithms for 2 different baselines (1x and 3x).

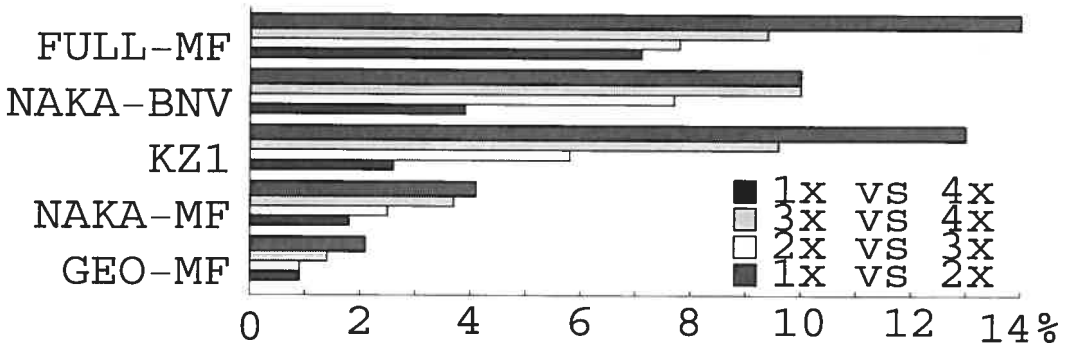

Figure 5.7: Resistance to baseline change for 5 algorithms for the Santa scene (Multiview Image Database of the University of Tsukuba); each bar represents a percentage of incompatible pixels between depth maps obtained for two different baselines.

As for future work, better approach to recover from error in scene breaking the ordering constraint should be investigated. Also, the extension of this occlusion model to full volumetric reconstruction, where occlusion becomes the dominant problem, should be investigated.

5.7 Acknowledgment

This work was made possible by NSERC (Canada) and NATEQ (Québec) grants.

Chapitre 6

FAST MULTIPLE-BASELINE STEREO WITH OCCLUSION

Cet article [15] a été publié comme l'indique la référence bibliographique.

Marc-Antoine Drouin, Martin Trudeau and Sébastien Roy, Fast Multiple-baseline Stereo with Occlusion, *The 6th International Conference on 3-D Digital Imaging and Modeling (3DIM'05)*, Ottawa, Canada, June 2005, pages 540-547

Abstract

This paper presents a new and fast algorithm for multi-baseline stereo designed to handle the occlusion problem. The algorithm is a hybrid between fast heuristic occlusion overcoming algorithms that precompute an approximate visibility and slower methods that use correct visibility handling. Our approach is based on iterative dynamic programming and computes simultaneously disparity and camera visibility. Interestingly, dynamic programming makes it possible to compute exactly part of the visibility information. The remainder is obtained through heuristics. The validity of our scheme is established using real imagery with ground truth and compares favorably with other state-of-the-art multi-baseline stereo algorithms.

6.1 Introduction

The goal of multi-baseline stereo is to reconstruct the 3D structure of a scene from multiple views with optical centers located in a two-dimensional grid configuration. In this paper, we used a configuration of five rectified images, equally spaced and

arranged in a cross (Fig. 6.1). The disparity map is reconstructed from the point of view of the center camera which we call the reference. Occlusion occurs when part of a scene is visible in the reference but not in some supporting camera (Fig. 6.1). The difficulty of detecting occlusion comes from the fact that it is induced by the 3D structure of the scene, which is unknown until the correspondence is established. This paper proposes a novel multiple-baseline stereo algorithm that computes simultaneously the disparity and part of the visibility information. The remaining visibility information cannot be computed and is found using heuristics. The proposed approach uses iterative dynamic programming (IDP) [68], a fast method for computing disparity maps. When applied to ordinary stereo, it minimizes the same energy function as Graph Cut [8] but obtains slightly higher error rates.

We use a unique property of dynamic programming that allows the application of IDP to mutiple-baseline stereo in a way that is impossible to do with Graph Cut. In this paper, we assume that the images are already rectified. For more details about image rectification, see [110].

To compute a disparity map for a reference image, we use a set of reference pixels $\mathcal{P}$ and a set of disparity labels $\mathcal{D}$. A $\mathcal{D}$-configuration $f : \mathcal{P} \mapsto \mathcal{D}$ associates a disparity label to every pixel. When occlusion is not modeled, the energy function to minimize typically is

$$E(f) = \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} e(\mathbf{p}, f(\mathbf{p}))}_{\substack{\textit{pointwise} \\ \textit{likelihood}}} + \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} \sum_{\mathbf{r}\in\mathcal{N}_\mathbf{p}} s(\mathbf{p}, \mathbf{r}, f(\mathbf{p}), f(\mathbf{r}))}_{\textit{smoothing}} \tag{6.1}$$

where e uses all the cameras for each pixel and $\mathcal{N}_\mathbf{p}$ is a neighborhood of pixel $\mathbf{p}$. This can be solved because the likelihood term $e(\mathbf{p}, f(\mathbf{p}))$ is pointwise independent and the smoothing uses a 2-site clique form.

To model occlusion, we must compute the volumetric visibility $V_i(\mathbf{q}, d, f)$ of a reference pixel $\mathbf{q}$ located at disparity d from the point of view of a camera i, given

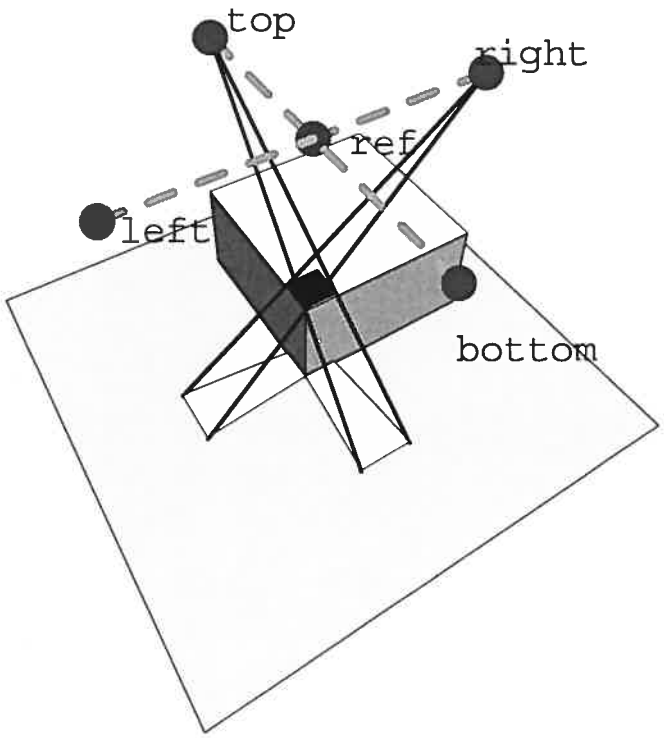

Figure 6.1: A cross-shaped camera configuration. The reference and the supporting cameras are labeled *ref*, *left*, *right*, *top* and *bottom* respectively. An example of occluder and occluded pixels are shown in black and white respectively.

a disparity configuration f defined for all other pixels. It is set to 1 if the point is visible, and 0 otherwise. The visibility information is collected into a vector as a *visibility mask*

$$V(\mathbf{q}, d, f) = (V_1(\mathbf{q}, d, f), \ldots, V_N(\mathbf{q}, d, f))$$

where N is the number of cameras outside the reference; a vector $(1, \ldots, 1)$ means that the 3D point is visible in all supporting cameras and $(0, \ldots, 0)$ means that it is invisible. We call $\mathcal{M}$ the set of all possible visibility masks; an $\mathcal{M}$-configuration $g : \mathcal{P} \mapsto \mathcal{M}$ associates a mask to every pixel of the reference image. Using this, we

transform Eq. 6.1 into an occlusion-aware energy function

$$E(f,g) = \sum_{\mathbf{p}\in\mathcal{P}} e(\mathbf{p}, f(\mathbf{p}), g(\mathbf{p})) + \textit{smoothing}. \tag{6.2}$$

Ideally, we would like to use $g(\mathbf{p}) = V(\mathbf{p}, f(\mathbf{p}), f)$ everywhere. Since this introduces a dependency between f and g, thus making the problem too hard, we will use instead for some cases a *correct* visibility, i.e. a g satisfying

$$g(\mathbf{p}) \leq V(\mathbf{p}, f(\mathbf{p}), f) \tag{6.3}$$

for each component of these vectors and all $\mathbf{p} \in \mathcal{P}$. For the others, we will say that the handling of visibility is *heuristic*. Typically, we define the pointwise likelihood

$$e(\mathbf{p}, d, \mathbf{m}) = \frac{\mathbf{m} \cdot C(\mathbf{p}, d)}{|\mathbf{m}|} \quad \text{for } \mathbf{p} \in \mathcal{P},\ d \in \mathcal{D},\ \mathbf{m} \in \mathcal{M} \tag{6.4}$$

where $C(\mathbf{q}, d) = (c_1(\mathbf{q}, d), \ldots, c_N(\mathbf{q}, d))$ is the vector of matching costs of the pixel $\mathbf{q}$ at disparity d for each camera. We use $|\mathbf{m}|$ to represent the l_1-norm which is just the number of cameras visible from $\mathbf{q}$ at d. In a multi-baseline configuration, we can make the hypothesis that every point of the reference is seen by at least one supporting camera. So the case where $|\mathbf{m}| = 0$ cannot occur. For example, a simple cost function would be $c_i(\mathbf{q}, d) = (I_{ref}(\mathbf{q}) - I_i(T_i(\mathbf{q}, d)))^2$ where I_{ref} and I_i are respectively the reference and the supporting image i. $\mathbf{T}_i$ transforms pixel $\mathbf{q}$ at disparity d into the corresponding pixel of image i. In order to simplify the discussion, we will always consider the disparity as a positive value independently of the supporting camera used, and the T_i's take this into account.

The rest of this paper is divided as follows: in Section 6.2, previous work is presented. Section 6.3 describes our algorithm. Experimental results are presented in Section 6.4.

6.2 Previous work

In a recent empirical comparison of strategies to overcome occlusion for 2 cameras, Egnal [20] enumerates 5 basic ones: left-right checking, bimodality test, goodness Jumps

constraint, duality of depth discontinuity and occlusion, and uniqueness constraint. Some algorithms that have been proposed rely on one or more of the these strategies, and are often based on varying a correlation window position or size [52, 31, 112, 53]. Other algorithms use dynamic programming [80, 47, 2, 14] because of its ability to efficiently solve more complex matching costs and smoothing terms. Two methods using graph theoretical approaches [49, 57] have been proposed, but again they do not generalize well to multiple-camera configurations. Okutomi and Kanade have proposed a matching cost function designed to reduce ambiguity in stereo with multiple cameras having collinear optical centers [82]. However, their approach does not model occlusion.

When extending binocular stereo to multiple baselines, the amount of occlusion increases since each pixel of the reference camera can be hidden in more than one supporting camera. Some researchers have proposed specially designed algorithms based on pre-computed visibility masks to cope with this. A subset $\mathcal{M}_h$ of the most likely visibility masks of $\mathcal{M}$ is selected based on knowledge of the camera configuration. In order to determine the mask for a pixel $\mathbf{p}$ at disparity $f(\mathbf{p})$, the most photo-consistent one $g_f^*(\mathbf{p})$ is selected, that is

$$g_f^*(\mathbf{p}) = \arg \min_{m \in \mathcal{M}_h} e(\mathbf{p}, f(\mathbf{p}), m)\, w(m)$$

where $w(m)$ is a weight function favoring certain masks over others [79]. The problem thus becomes the minimization of $E(f, g_f^*)$ in f. Since e is pointwise independent, the new problem is reduced to the original formulation of Eq. 6.1 and is easily solved using standard algorithms. This technique is used in [79, 92, 86, 53]. Since the selected masks and the disparity map do not always respect Eq. 6.3, these methods are *heuristic*. A survey paper by Scharstein and Szeliski compares various binocular standard algorithms [94].

Other approaches try to minimize directly Eq. 6.2 in f and g, subject to the constraint of Eq. 6.3. Such *correct* methods have to solve a substantially more difficult

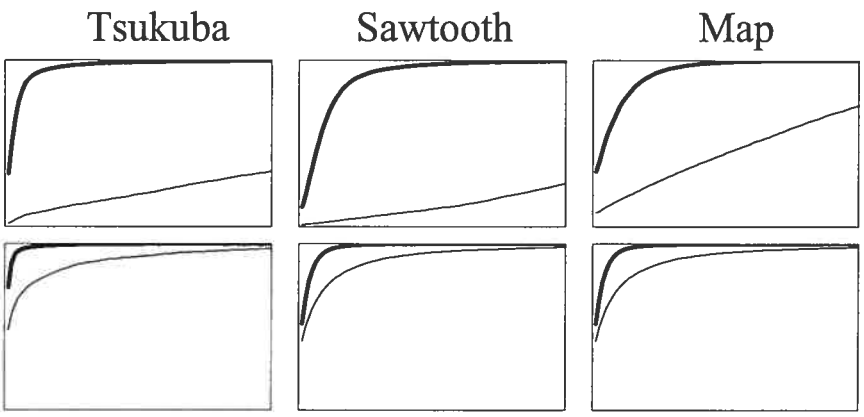

Figure 6.2: Cumulative histograms of matching cost values of occluded (thin line) and non occluded (thick line) pixels for 3 of the 4 test sequences of the Middlebury comparative study [94], for the **top**) depths from ground truth **bottom**) most likely depths (direct search). The x-axis' range is $[0, 80]$ and the y-axis' is $[0, 1]$.

problem than *heuristic* ones. In [66, 98], visibility-based methods are introduced. The matching cost incorporates the visibility information as a photo-consistency matching criteria, thereby implicitly modeling occlusion in the reconstruction process. Space carving can be seen as a greedy algorithm that minimizes Eq. 6.2 without smoothing. Similarly, a level-set method [23] uses the visibility information from the evolving reconstructed surface to explicitly model occlusion. In this case, depths are continuous and the problem is difficult to cast in the discrete setting of Eq. 6.2. Nevertheless, the idea is similar. In [59], a stereo algorithm based on graph cuts is presented. It strictly enforces visibility constraints to guide the matching process and ensures that all visibility masks are consistent with the recovered surface. This algorithm jumps from one configuration respecting Eq. 6.3 to another. The formulation imposes strict constraints on the form of the smoothing term, constraints that will not apply to our method.

6.3 A hybrid algorithm

Heuristic approaches rely on the hypothesis that photo-consistency implies visibility. The figure 6.2 suggests that this is not always true. Using the matching cost function and images from the Middlebury comparative study [94], we computed the cumulative histograms of cost values for pixels classified as occluded and non occluded, based first on the ground truth and then on the computed disparity maps using direct search. The histograms are very different when the ground truth is used, but not when a direct search is. This indicates that many occluded pixels have a low cost and illustrates the fact that photo-consistency does not imply visibility. Ideally, we would like to benefit from the speed and simplicity of heuristics without being affected by the similarity between the matching cost distribution of occluded and non occluded pixels.

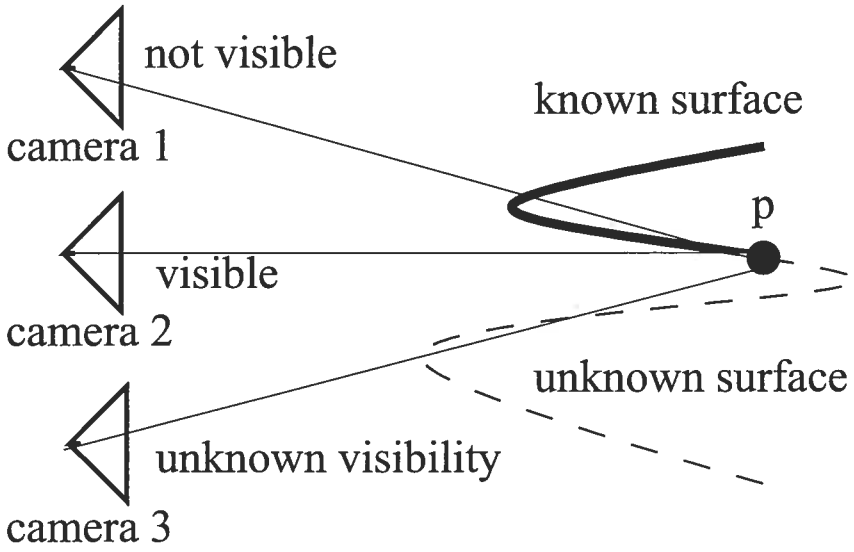

Figure 6.3: Example of the use of partial disparity map information. It is known that the point p is not visible from camera 1 but it is visible from camera 2. For camera 3, the partial disparity map does not allow us to know the visibility.

The algorithm we propose goes through the reference image pixel by pixel, building potential disparity maps for all the pixels up to the current one. At all time, the algorithm has access to the correct visibility information for a subset $\mathcal{C}_c$ of the set $\mathcal{C}$ of all supporting cameras. This visibility information comes from the partial knowledge of the disparity map (Fig. 6.3).

We can build a set $\mathcal{M}_c$ of masks using only cameras in $\mathcal{C}_c$. Given that each camera can be used or not, and discarding the empty mask, we are left with $2^{\#\mathcal{C}_c} - 1$ masks (# denotes the cardinality as usual). A mask from $\mathcal{M}_c$ is *correct*, independently of the visibility status of the cameras not in $\mathcal{C}_c$ (it has been noticed that removing a camera that is visible is far less damaging than keeping a camera that is not visible [79]). For the cases where no camera in $\mathcal{C}_c$ is visible, we must select a mask in another set $\mathcal{M}_h$ that only uses cameras in the subset $\mathcal{C}_h = \mathcal{C} - \mathcal{C}_c$. We contruct $\mathcal{M}_h$ so it only contains masks with one supporting camera. We thus have $\#\mathcal{M}_h = \#\mathcal{C}_h$. Since the set $\mathcal{M}_h$ can contain more than one mask, we use the heuristic that photo-consistency implies visibility to select the visibility mask. When a mask in $\mathcal{M}_h$ is selected, it is known that all cameras in the subset $\mathcal{C}_c$ are not visible. Since they are not used in $\mathcal{M}_h$, the heuristic mask is likely to be in fact correct. We expect the choice between $\mathcal{M}_c$ and $\mathcal{M}_h$ to be spacially coherent, we can thus add a visibility smoothing term that penalizes the use of masks belonging to different sets for adjacent pixels.

Explicitly, for a pixel $\mathbf{p}$ and a certain disparity map f constructed up to the previous pixel, if $\mathbf{p}$ is visible by at least one camera in $\mathcal{C}_c$, the mask of $\mathbf{p}$ at d is set to a partial visibility $V'(\mathbf{p}, d, f)$ with each component defined as

$$V'_i(\mathbf{p}, d, f) = \begin{cases} V_i(\mathbf{p}, d, f) & \text{if } i \in \mathcal{C}_c \\ 0 & \text{otherwise.} \end{cases}$$

If $\mathbf{p}$ is not visible by any camera in $\mathcal{C}_c$, its mask is defined as $\arg\min_{m \in \mathcal{M}_h} e(\mathbf{p}, d, m)$. Note that the mask which minimizes the previous energy function would also minimize it if other masks containing more than one camera were added to the set $\mathcal{M}_h$. This

comes from the matching cost function of Eq. 6.4 and the fact that the mean of multiple values is always greater than the smallest of these values.

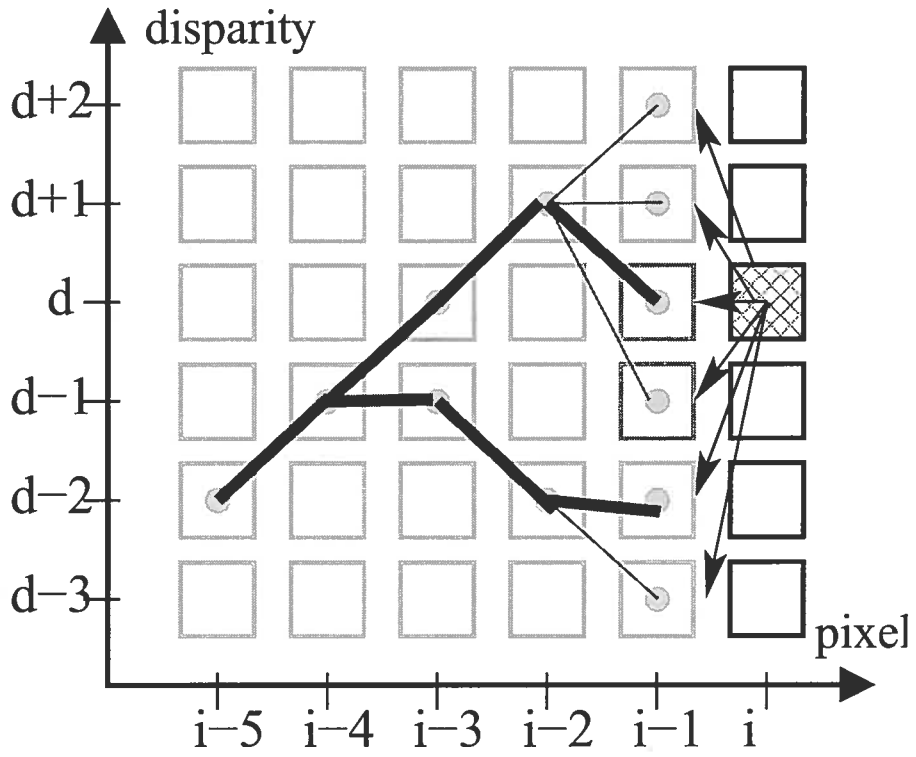

Figure 6.4: DP matching process. To determine the best disparity map up to pixel i with disparity d, for the different disparity values of $i-1$, we look at the best solution up to $i-1$, available by construction.

Our algorithm finds an f and a g having a low energy according to Eq. 6.2 (not necessarily a global minimum), respecting the constraint

$$g(\mathbf{p}) = \begin{cases} V'(\mathbf{p}, f(\mathbf{p}), f) & \text{if } \exists i \in \mathcal{C}_c : V_i(\mathbf{p}, f(\mathbf{p}), f) = 1 \\ \arg\min_{m \in \mathcal{M}_h} e(\mathbf{p}, f(\mathbf{p}), m) & \text{otherwise.} \end{cases}$$

Disparity and visibility will be solved simultaneously using dynamic programming, taking into account long range visibility interactions.

6.3.1 Optimizing disparity and visibility

The stereo matching proceeds using Dynamic Programming (DP) applied along epipolar lines, which can be horizontal or vertical for our camera configuration. We illustrate the process for a left-right epipolar line, with a center reference image and left and right supporting images. When dynamic programming proceeds along, the computation of the disparity at pixel i can rely on knowledge of the disparities of all preceding pixels (Fig. 6.4). In the following discussion, a left to right order is assumed, but a right to left one is possible as well. Because of this, the visibility between any camera to the left of the reference and the 3D point formed by pixel i at disparity d is also known (Fig. 6.3). A similar strategy for binocular stereo was presented in [2]. When going from left to right, $\mathcal{C}_c$ consists of the left camera and $\mathcal{C}_h$ of the right one. Consequently, $\mathcal{M}_c$ contains only the mask consisting of the left camera, namely $(1,0)$. Similarly, $\mathcal{M}_h$ is simply $\{(0,1)\}$. When solving the correspondence problem along an epipolar line, two 2-dimensional tables t and t' are filled out; $t(i,d)$ is the lowest energy of all disparity maps of pixels 0 to i with pixel i at disparity d; $t'(i,d)$ is the disparity of pixel $i-1$ given by this map of lowest energy, denoted $f_{i,d}(i')$. Two sample disparity maps, $f_{i-1,d}$ and $f_{i-1,d-2}$, are highlighted in Fig. 6.4. The table t' is used to compute the different $f_{i,d}$'s.

Explicitly, the tables t and t' are defined inductively as

$$
\begin{aligned}
t(0,d) &= e(0,d,(1,0)) \\
t'(0,d) &= d \\
t(i,d) &= \min_{d' \in \mathcal{D}} \begin{pmatrix} e_v(i,d,d') \\ + \; s(i-1,i,d',d) \\ + \; t(i-1,d') \end{pmatrix} \\
t'(i,d) &\quad \text{is the index of the minimum in the above formula}
\end{aligned}
$$

where

$$e_v(i,d,d') = \begin{cases} e(i,d,(1,0)) & \text{if } O(i,d,d') < 0 \\ e(i,d,(0,1)) & \text{otherwise} \end{cases}$$

and $O(i,d,d')$ is a visibility function that is smaller than 0 iff the left camera is visible. It only requires the knowledge of $f_{i-1,d'}(j)$ for $j < i$. The $f_{i,d}$'s can be computed with the relations

$$\begin{aligned} f_{i,d}(i) &= d \\ f_{i,d}(j) &= t'(j+1, f_{i,d}(j+1)) \quad \text{for} \quad 0 \leq j < i. \end{aligned}$$

It is thus possible to compute $f_{i-1,d'}(j)$ for all $j < i$ and $d' \in \mathcal{D}$. This allows us to compute visibility $O(i,d,d')$ for all d' and finally $t(i,d)$. Note that if for some $j \leq i' \leq i$ and $d, d' \in \mathcal{D}$ we have $f_{i,d}(j) = f_{i',d'}(j)$ then $f_{i,d}(k) = f_{i',d'}(k)$ for all $k \leq j$. Moreover, the likelihood term e of a pixel i is not pointwise independent but depends on every pixel located to its left. As mentioned before, s may include visibility as well as disparity smoothing.

6.3.2 Computing visibility

When computing the left visibility function, the disparity map representation has an impact. We can consider a disparity map as a series of disconnected 3D points or as a continuous mesh. We define $O(i,d,d')$ as the visibility of pixel i at disparity d for the best solution with pixel $i-1$ at disparity d'. In the discontinuous case, O is defined as

$$O(i,d,d') = \begin{cases} 0 & \text{if } i + d = j + f_{i-1,d'}(j) \\ & \text{for some } j < i \\ -1 & \text{otherwise.} \end{cases} \tag{6.5}$$

It takes the value 0 when occlusion occurs and -1 when the left camera is visible. In the continuous case, we can lower the complexity by introducing the function O'

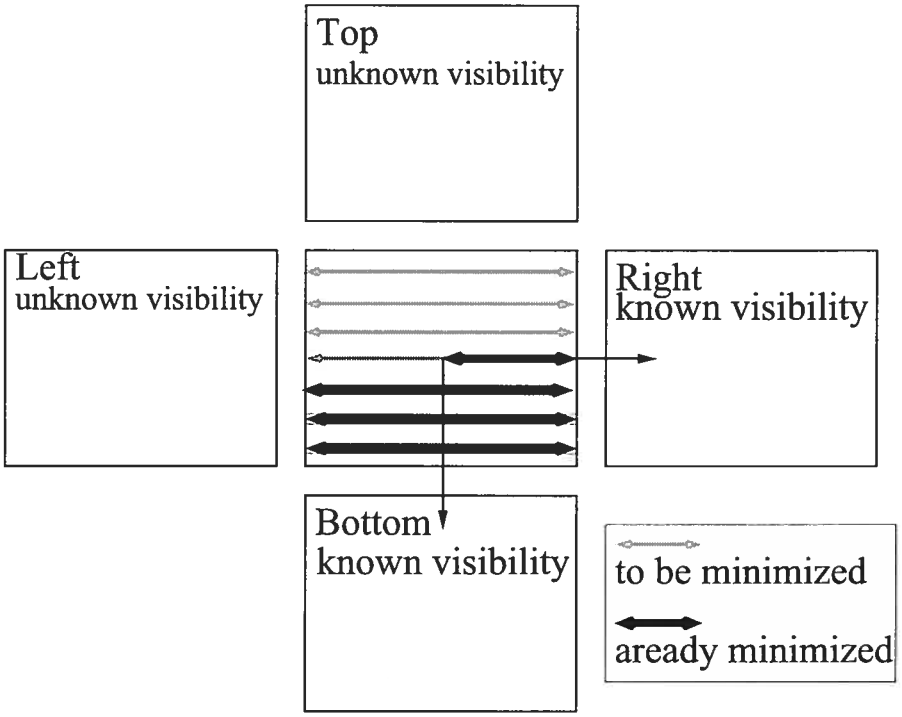

Figure 6.5: Optimization of a line: the visibility information is available for the right and the bottom cameras. For the current line, the right camera visibility is computed simultaneously with the disparity. The bottom camera visibility depends on the disparity of the previous lines which is fixed.

defined as

$$O'(j, d') = \max_{0 \le k \le j} (k + f_{j,d'}(k)).$$

This function can be computed inductively using the relations

$$O'(0, d') = d'$$

$$O'(j, d') = \max\{O'(j-1, f_{j,d'}(j-1)), j + d'\}$$

for $j > 0$.

Now O can be simply defined as

$$O(i, d, d') = O'(i-1, d') - (i + d) \tag{6.6}$$

which is negative when the left camera is visible. Using a continuous mesh is equivalent to imposing the ordering constraint [20] in the selection of visibility masks, but not on the disparity map itself. Note that it is much faster to compute the continuous case. Moreover, as will be shown in section 6.4, always treating disparity maps as continuous does not affect the quality of the reconstruction in a scene that breaks the ordering constraint. The relations for right to left, top to bottom and bottom to top minimization are obtained similarly.

6.3.3 Visibility-aware iterative dynamic programming

Optimization	Mask	Visibility
PIX right to left	$\mathcal{M}_c$ = { (**0,1,0,0**),(**0,0,0,1**),(**0,1,0,1**) }	correct
LINE bottom to top	$\mathcal{M}_h$ = { (**1,0,0,0**),(**0,0,1,0**) }	heuristic
PIX bottom to top	$\mathcal{M}_c$ = { (**1,0,0,0**),(**0,0,0,1**),(**1,0,0,1**) }	correct
LINE left to right	$\mathcal{M}_h$ = { (**0,1,0,0**),(**0,0,1,0**) }	heuristic
PIX left to right	$\mathcal{M}_c$ = { (**1,0,0,0**),(**0,0,0,1**),(**1,0,0,1**) }	correct
LINE bottom to top	$\mathcal{M}_h$ = { (**0,1,0,0**),(**0,0,1,0**) }	heuristic
PIX top to bottom	$\mathcal{M}_c$ = { (**1,0,0,0**),(**0,0,1,0**),(**1,0,1,0**) }	correct
LINE left to right	$\mathcal{M}_h$ = { (**0,1,0,0**),(**0,0,0,1**) }	heuristic

Table 6.1: Visibility masks and their status depending on the current step. PIX refers to the order inside the line being solved,while LINE refers to the order in which lines are processed. In bold are the cameras belonging to $\mathcal{C}_c$. The camera order in a mask is left,right,top and bottom.

In the previous section, we discussed visibility computation along an epipolar line. When working with a 5-camera cross-shape configuration, illustrated in Fig. 6.5, we can use the solution of the previous lines to compute the visibility of one of the cameras perpendicular to the line being processed. The visibility function O for such a camera is computed similarly as that of the camera with *correct* visibility along the current line. There is an important difference between the visibility information

coming from the disparity maps of the previous lines and that from the line currently being processed: for latter, the disparity map is part of the minimization process, for the former it is fixed (Fig. 6.5).

In order to apply smoothing across epipolar lines, we use Iterative Dynamic Programming (IDP) proposed by Leung *et al.* [68]. For binocular stereo, they proceed in two steps: first they solve along horizontal lines and then along vertical lines. They repeat these two steps until a certain convergence criteria is met. The spatial smoothing term is not limited to a single line, but uses the last disparity information obtained from previous lines, step or iteration. We use the same smoothing strategy, but proceed in four steps when solving for lines and columns. In a first step, illustrated in Fig. 6.5, we start solving for horizontal lines from bottom to top, applying dynamic programming (DP) from right to left inside each line. The visibility computation relies on the disparity map obtained for the lower lines. In a second step, we solve for vertical lines from left to right, applying DP from bottom to top inside each line. Once again, solutions to previous lines are used for visibility. In the third step, we solve for horizontal lines from bottom to top, applying DP from right to left; for the fourth and last step, we solve for vertical lines from left to right, applying DP from top to bottom. Note that $\mathcal{C}_c$ and $\mathcal{M}_c$ vary from one step to the next. Table 6.1 shows the different masks for which the visibility is either *correct* or *heuristic* depending on the current step.

We also propose a different initialization; in [68], the disparity is initialized to a constant value. However, we do not use any prior disparity solution, the spatial smoothing is constrained to the current line during the first step of the algorithm. For subsequent steps, smoothing is performed along and across lines based on previously obtained solutions.

An iteration consists of the four steps described above. After the first iteration, every camera in $\mathcal{C}$ has been in $\mathcal{C}_c$ at least once. With a 5-camera cross configuration, in each step there is exactly one camera along the current epipolar line for which

we have *correct* visibility at all time. For this reason, this configuration performs particularly well.

We can iterate to improve the disparity map. Our algorithm does not necessarily converge, as it is possible for the process to cycle. In practice, we stop after 1 to 8 iterations since changes are minimal after that. After one iteration, the algorithm already provides high quality disparity maps.

When using the visibility function of Eq. 6.6 and hence representing the disparity map as a continuous mesh, the asymptotic complexity of the algorithm remains the same as for ordinary IDP, that is $\Theta(\#\mathcal{P}\,\#\mathcal{D}^2)$ where $\#\mathcal{P}$ is the number of reference pixels and $\#\mathcal{D}$ the number of disparity labels. When using the visibility function of Eq. 6.5, the asymptotic complexity increases to $\Theta(\#\mathcal{P}\,\#\mathcal{D}^3)$. In our experiments, a continuous mesh representation was always used.

Algorithm	**Error**
BNV-Truth	1.01%
DP-Hybrid (4 iterations, $\gamma = 19$)	1.67%
BNV-heuristic	1.77%
DP-Hybrid (1 iteration, $\gamma = 19$)	1.82%
DP-Hybrid (12 iterations, $\gamma = 0$)	2.01%
KZ1	2.30%
DP-Heuristic (12 iterations)	2.35%
DP-Heuristic (1 iteration)	2.63%
DP-Hybrid (1 iteration,$\gamma = 0$)	2.77%

Table 6.2: Percentages of error of the different algorithms for the Head and Lamp scene, using 5 images. All algorithms use the same matching cost and smoothing function.

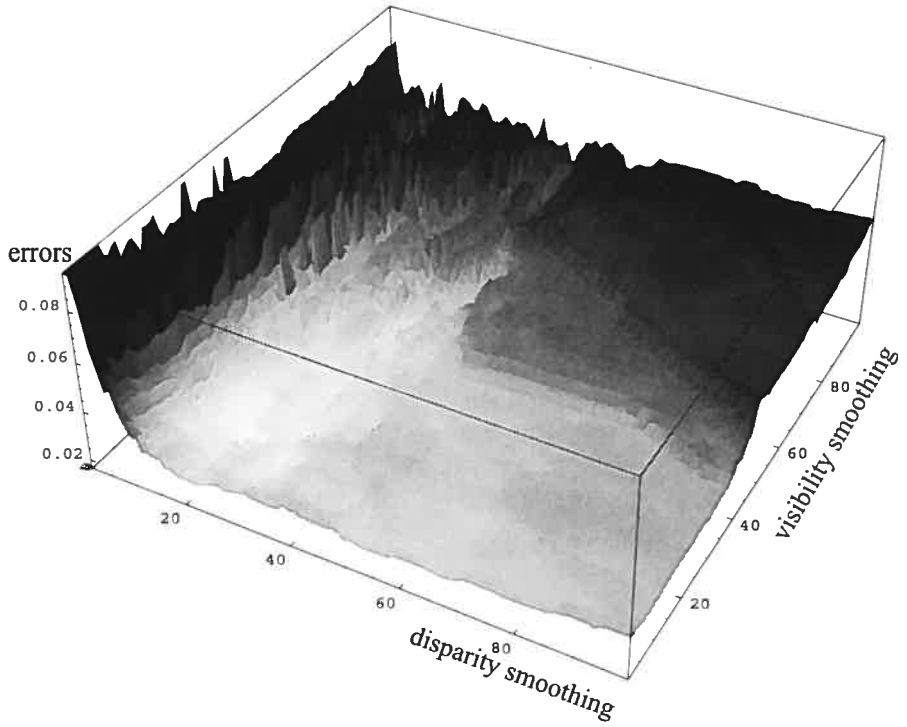

Figure 6.6: Resistance to change of the smoothing parameter for the Head and Lamp scene for one iteration. Both smoothing parameters increase by a factor of more than 100.

6.4 Experimental results

In all our experiments, the matching cost function was the same for all algorithms, that of [59] which is based on [4]. In our experiments we used color images; only the reference images in gray scale are shown here. For the smoothing term, we used the experimentally derived smoothing function that also comes from [59]:

$$s(\mathbf{p}, \mathbf{r}, f(\mathbf{p}), f(\mathbf{r})) = \lambda\, t(\mathbf{p}, \mathbf{r})\, \delta(f(\mathbf{p}) - f(\mathbf{r}))$$

where δ is 1 at 0 and 0 elsewhere and t is defined as

$$t(\mathbf{p},\mathbf{r}) = \begin{cases} 3 & \text{if } |I_{ref}(\mathbf{p}) - I_{ref}(\mathbf{r})| < 5 \\ 1 & \text{otherwise} \end{cases} .$$

While we chose the Potts model for smoothing, dynamic programming can in fact use any model. As previously mentioned, we added to our method a visibility smoothing, taking the value 0 if the mask of the current pixel and that of its neighbor are both in $\mathcal{M}_c$ or both in $\mathcal{M}_h$, and the value γ if they are not. The details are similar *mutatis mutandis* as for the usual disparity smoothing (see [68]). The parameters λ and γ are user-defined smoothness levels. For each disparity map computation, we chose the λ and γ that achieved the best performance. A pixel disparity is considered erroneous if it differs by more than one disparity step from the ground truth. This error measurement is compatible with the one used in two comparative studies for 2-camera stereo [105, 94, 59].

6.4.1 Tsukuba Head and Lamp scene

This dataset is from the Multiview Image Database from the University of Tsukuba (Fig. 6.7). It is composed of a 5×5 image grid. Each image has a resolution of 384×288. The search interval is between 0 and 15 pixels and we used 16 disparity steps. The reference image is the center one and the four supporting images are the closest to it, forming a cross.

In order to show the validity of our algorithm, we compared our method with other state-of-the-art multi-baseline algorithms. Results from Graph Cut using Nakamura's visibility masks are labeled BNV-Heuristic. This is an adaptation of [92] where the maximum flow formulation of [89] was used with the precomputed visibility masks of [79]. We replaced the maximum flow by the Graph Cut algorithm of [8] (ranked the best stereo matcher in two comparative studies [105, 94]), having observed that, for this scene, it achieves a lower error rate. We also ran a version of the previous

algorithm using IDP instead of Graph Cut as the optimization method (labeled DP-Heuristic), with 1 and 12 iterations. For the two versions, we tried different sets of masks $\mathcal{M}_h$ and picked the one achieving the best performance, namely the one that uses only 2 supporting cameras in each mask. This set has a total of 6 visibility masks.

Results of our method are shown under the label DP-Hybrid, with 1 and 4 iterations when using visibility smoothing, with 1 and 12 iterations when not. The only difference between DP-Heuristic and DP-Hybrid is the occlusion model. We also compared our algorithm with that of [59], a method with *correct* visibility handling (labeled KZ1). Its error measurement was taken directly from the same article. Finally, we computed the disparity maps using Graph Cut with the *exact* visibility masks computed in advance from the ground truth. We labeled this method BNV-Truth.

Some disparity maps are shown in Fig. 6.7; all the error percentages are shown in Table 6.2. Our method with 4 iterations achieved the lowest error rate after BNV-Truth. Even after one iteration, the error rate is low and requires less than 4 seconds of running time on a 2.0 GHz Athlon 64 with a non optimized implementation. Table 6.2 also shows the minimal impact of subsequent iterations after the first when using visibility smoothing. The error goes down with additional iterations, but only by a small amount. This figure also shows the impact of visibility smoothing for our algorithm.

Figure 6.6 shows a stability analysis of the smoothing parameter for our algorithm (DP-Hybrid), giving the error percentages over a broad range of values.

There are pixels for which the ordering constraint is broken, in particular in the arm of the lamp (Fig. 6.7). They were identified by re-projecting the ground truth in each supporting camera. They did not affect our algorithm even if our visibility computation makes the hypothesis that the ordering constraint is respected.

6.4.2 Others scenes

We used the Plant and the Santa scenes from the Multiview Image Database of the University of Tsukuba (Fig. 6.8). These datasets contain 81 images in a 9×9 grid taken with a camera having a 10 mm focal length. We only used 5 images in our usual cross configuration. Images were reduced by a factor of 2 to achieve a resolution of 320×240. Each disparity map was computed using 24 disparity steps. For these datasets, the value of γ did not have a significant impact. We display the results for $\gamma = 0$.

The Plant scene features a lot of occlusion coming from very thin objects and constitutes a very good test for the validity of any occlusion model. The red flower is located at 59cm of the reference camera, the back blue ones at 92cm and the background at 184cm. The baseline is 20 mm. The disparity map obtained after only one iteration is shown in Fig. 6.8. The running time was under 4 seconds. The results for 8 iterations are similar.

The second scene features a Santa doll. The hand is located at 59 cm of the reference camera and the background at 184 cm. The disparity map obtained after 8 iterations is shown in Fig. 6.8 (it is slightly better than the one obtained after one iteration). Note the details on the right side of the hat and on the candle.

6.5 Conclusion

We have presented a new stereo matching algorithm for multiple-baseline stereo. Our approach is a hybrid between the fast methods that use photo-consistency to approximate correct visibility and slower methods that use correct visibility. In this paper, we used a Potts model, but the proposed algorithm is flexible enough to be used with any type of smoothing term. It is fast and also succeeds in obtaining sharp and well-located depth discontinuities. The validity of our framework has been demonstrated on standard datasets with ground truth and compares favorably with

other state-of-the-art occlusion models for multiple-view stereo.

As for future work, we would like to build a real-time implementation using dedicated hardware such as FPGA's. In addition, the extension of this occlusion model to arbitrary multi-camera configurations and volumetric reconstruction should be explored.

6.6 Acknowledgment

This work was made possible by NSERC (Canada) and NATEQ (Québec) grants.

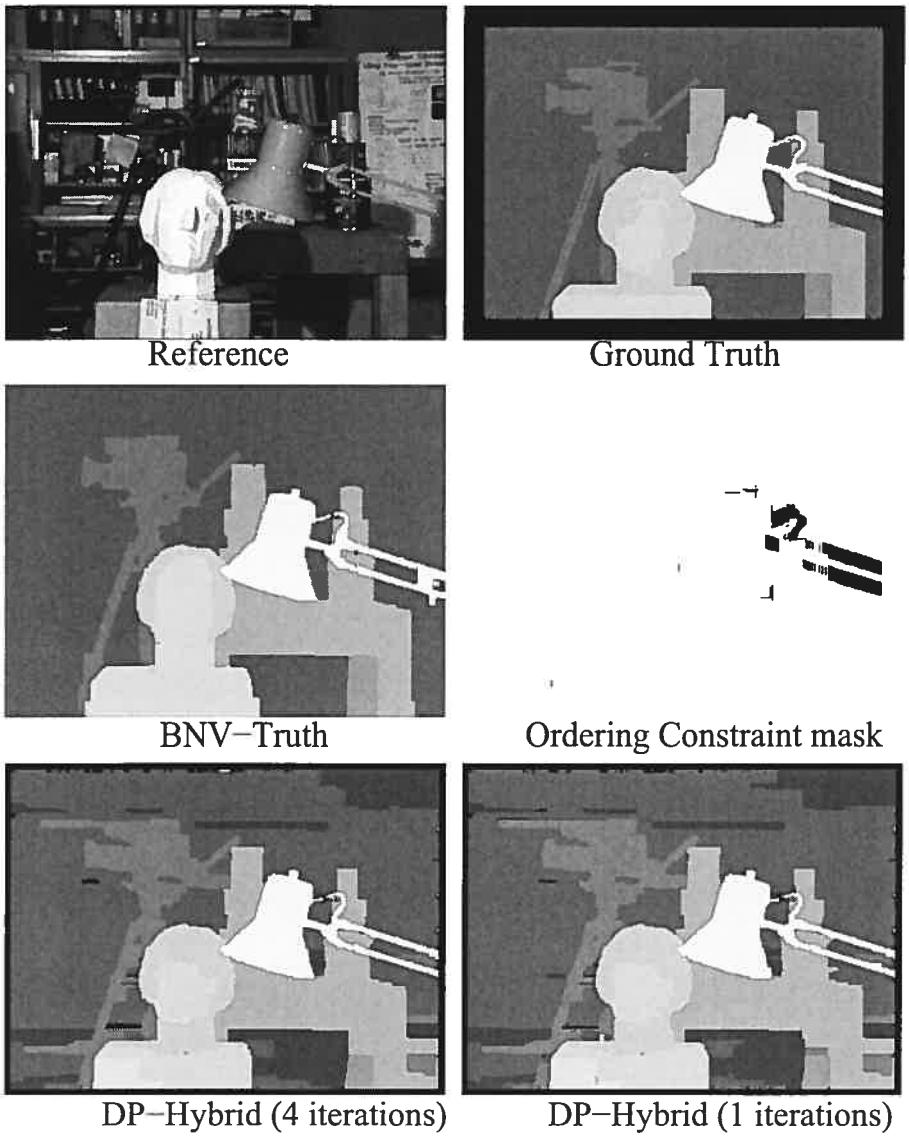

Figure 6.7: Disparity maps for various algorithms for the Head and Lamp scene (Multiview Images database of the University of Tsukuba). DP-Hybrid was run with a value of $\gamma = 19$. A linear mapping of disparities to gray levels was used. The mask of pixels breaking the ordering constraint in at least one supporting camera is also shown.

Figure 6.8: Disparity maps for the Plant and Santa scene using DP-Hybrid (Multiview Image Database of the University of Tsukuba). A linear mapping of disparities to gray levels was used.

Chapitre 7

IMPROVING BORDER LOCALIZATION OF MULTI-BASELINE STEREO USING BORDER-CUT

Cet article [18] a été publié comme l'indique la référence bibliographique.

Marc-Antoine Drouin, Martin Trudeau and Sébastien Roy, Improving Border Localization of Multi-Baseline Stereo Using Border-Cut, *International Conference on Computer Vision and Pattern Recognition (CVPR'06)*, New York, USA, June 2006, pages 511-518

Abstract

This paper presents a novel algorithm that improves the localization of disparity discontinuities of disparity maps obtained by multi-baseline stereo. Rather than associating a disparity label to every pixel of a disparity map, it associates a position to every disparity discontinuity. This formulation allows us to find an approximate solution to a 2D labeling problem with robust smoothing term by minimizing multiple 1D problems, thus making possible the use of dynamic programming. Dynamic programming allows the efficient computation of the visibility of most of the cameras during the minimization. Whilst the proposed minimization strategy is particularly suitable for stereo with occlusion, it may be used with other labeling problems.

7.1 Introduction

The goal of binocular stereo is to reconstruct the 3D structure of a scene from two views. Occlusion occurs when part of a scene is visible in the reference but not in

the supporting camera. The difficulty of detecting occlusion comes from the fact that it is induced by the 3D structure of the scene, which is unknown until the correspondence is established. When extending binocular stereo to multiple cameras, the amount of occlusion increases since each pixel of the reference camera can be hidden in more than one supporting camera. This is particularly true when going from a single to a multiple-baseline configuration. We propose a novel algorithm that improves the localization of disparity discontinuities of disparity maps obtained by multi-baseline stereo. For some applications such as augmented reality, well-localized borders are indeed very important. The algorithm computes simultaneously the border localization and most of the visibility information. Heuristics may be required to establish camera visibility for a small subset of the cameras.

The rest of this paper is divided as follows: in Section 7.2, previous work is presented. Section 7.3 describes our algorithm. Visibility is discussed in Section 7.4. Experimental results are presented in Section 7.5.

7.2 Previous work

In Egnal [20], five basic strategies to overcome occlusion for two cameras are presented: left-right checking, bimodality test, goodness Jumps constraint, duality of depth discontinuity and occlusion, and uniqueness constraint. Some algorithms rely on one or more of the these strategies, and are often based on varying a correlation window position or size [52, 31, 112, 53]. Other algorithms use dynamic programming [80, 47, 14]. Two methods using graphs [49, 57] have been proposed. In [102], visibility and disparity are iteratively minimized using belief propagation. Most of these methods are binocular in nature and do not generalize well to the case of multiple cameras. Some researchers have proposed specially designed algorithms to cope with occlusion in multiple camera configurations. They can be coarsely divided into three categories. Some approaches are based on the heuristic that a low matching cost

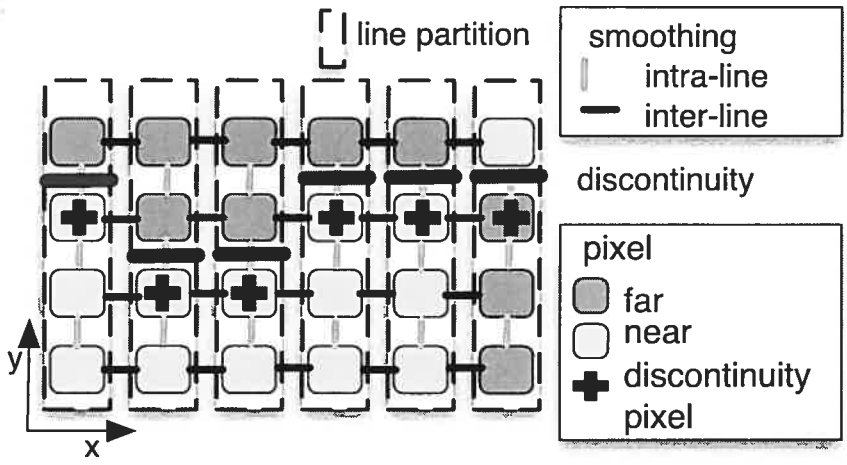

Figure 7.1: Representation of a disparity map with vertical line partitioning.

function implies the absence of occlusion [53, 79, 92, 87]. Others guarantee a solution that is geo-consistent [17, 66, 98, 23, 59]. These approaches preserve the consistency between the recovered visibility and the geometry [17]. Finally, some algorithms are mixes between heuristic and geo-consistent algorithms [15, 119, 36].

Many discrete optimization methods have been applied to stereo [8, 68, 103]. Our minimization strategy is similar to [8, 68]. All these methods find iteratively a disparity for a subset of the pixels while keeping the disparity of the others fixed. Nevertheless, our approach does not work directly on the disparity map; it works on the localization of the discontinuities. The two main differences between our approach and active contour ones are the discrete formulation and the absence of constraints on the smoothing term [83].

7.3 Formulation

We have a set $\mathcal{P}$ of reference pixels, for which we want to compute disparity, and a set $\mathcal{D}$ of disparity labels. A $\mathcal{D}$-configuration $f : \mathcal{P} \mapsto \mathcal{D}$ associates a disparity label

to every pixel. When occlusion is not modeled, the energy function to minimize is

$$E(f) = \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} e(\mathbf{p}, f(\mathbf{p}))}_{\substack{\text{pixel} \\ \text{likelihood}}} + \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} \sum_{\mathbf{p}'\in\mathcal{N}_\mathbf{p}} s(\mathbf{p}, \mathbf{p}', f(\mathbf{p}), f(\mathbf{p}'))}_{\text{pixel smoothing}} \tag{7.1}$$

where $\mathcal{N}_\mathbf{p}$ is pixel $\mathbf{p}$'s neighborhood, which can be 4- or 8-connected. When the smoothing term is robust, the minimization of Eq. 7.1 is an NP-hard problem [8].

We partition the reference image into the set $\mathcal{L}$ of its vertical or horizontal lines, where each line λ is a vector of pixels $(p_1^\lambda, \ldots, p_N^\lambda)$, N being the number of pixels on each line. Two lines are neighbors if two of their pixels are. The neighborhood of line λ is denoted $\mathcal{N}_\lambda$ and is 2-connected when $\mathcal{N}_\mathbf{p}$ is 4- or 8-connected. If there are only 2 different disparity values and a single disparity discontinuity in a line then there is only one *discontinuity* pixel p_d^λ in the line λ preceding a pixel with a different disparity (see Fig. 7.1). A $\mathcal{B}$-configuration $b : \mathcal{L} \mapsto \mathcal{B}$ associates a *discontinuity* pixel to every line. The set $\mathcal{B}$ is simply {1,2,...,N} and represents the index of the *discontinuity* pixel. There are $N-1$ possible discontinuity locations and it is possible for a line to have no discontinuity at all, which correspond to the label N.

We define $\mathcal{E}$ to be the set containing the two end pixels of every line. If we have an endpoint disparity map f' that associates a disparity to every pixel in $\mathcal{E}$ and a $\mathcal{B}$-configuration b, then we can define $F_{b,f'}$ to be the corresponding $\mathcal{D}$-configuration. Similarly, B_f is the $\mathcal{B}$-configuration associated with f (since f may have more than one discontinuity on one of its lines, B_f is not always well-defined). For an endpoint disparity map f', allowing only one discontinuity per line, the energy to minimize is

$$E_{f'}(b) = \underbrace{\sum_{\lambda\in\mathcal{L}} e_{f'}(\lambda, b(\lambda))}_{\substack{\text{discontinuity} \\ \text{likelihood}}} + \underbrace{\sum_{\lambda\in\mathcal{L}} \sum_{\lambda'\in\mathcal{N}_\lambda} s_{f'}(\lambda, \lambda', b(\lambda), b(\lambda'))}_{\text{inter-line smoothing}} \tag{7.2}$$

BORDER-CUT(f)

```
1  repeat
2        change ← 0
3        for R ⊂ P such that B_{f|R} is well-defined
4        do  Compute f' from f|R
5            E ← min_b [E_{f'}(b) + R_R(f|_{R^c}, F_{b,f'})]
6            b* ← the minimum in the above formula
7            if E < E_{f'}(B_{f|R}) + R_R(f|_{R^c}, f|R)
8              then change ← 1
9            Update f using F_{b*,f'}
10   until change = 0
11 return f
```

Figure 7.2: Overview of the Border-Cut algorithm

with respect to b, where $e_{f'}$ contains the likelihood of pixels one the same line and intra-line smoothing, i.e. the smoothing of neighboring pixels belonging to the same line (see Fig. 7.1). The inter-line smoothing (see Fig. 7.1) is the pixel smoothing of neighboring pixels belonging to different lines (see Fig. 7.1). Note that $E_{f'}(b) = E(F_{b,f'})$. The neighborhood used in Eq. 7.2 is 2-connected making the minimization easy assuming f' known and B_f well-defined. Standard stereo algorithms find sharp discontinuities but often they are not well localized in areas with occlusion [15]. Nevertheless, disparities on each side of these discontinuities are generally well recovered and provide reasonable endpoint disparity maps f' for the minimization of $E_{f'}(b)$ to yield a good global disparity map.

7.3.1 The Border Cut Algorithm

In a disparity map, there is generally more than one discontinuity on a line of pixels. Our algorithm does not work on whole lines but only on segments containing one *discontinuity* and centered around it. Two segments are neighbors if two of their pixels are and if they belong to different lines. A line sweeping strategy is used to choose the segments whose pixels will form the *active* subset $\mathcal{R}$ of the set of pixels $\mathcal{P}$, for which the localization of borders will be improved. This is done while keeping the disparity of other pixels (called *passive*) fixed. The smoothing between *active* and *passive* pixels is denoted $R_{\mathcal{R}}(f|_{\mathcal{R}^C}, F_{b,f'})$, where $\mathcal{R}^C$ is the complement of $\mathcal{R}$, i.e. $\mathcal{P} - \mathcal{R}$. Let us describe this process for horizontal lines swept from left to right. We first pick a disparity δ and only consider *discontinuities* between a pixel with disparity smaller than δ and one with disparity greater than or equal to δ. We call such a *discontinuity* a δ-discontinuity. We find the leftmost horizontal δ-discontinuity. We then recursively collect neighboring segments with δ-discontinuities to form $\mathcal{R}$, making sure there is at most one segment on each line. Having an initial disparity map f, we can minimize Eq. 7.2 on this set. This minimization, which we call *border move*, occurs on line 5 of the algorithm outlined in Fig. 7.2. The disparity map is then updated (line 9) and we start the above procedure again from the next leftmost unprocessed δ-discontinuity. This is repeated until all δ-discontinuities have been processed. This line sweeping strategy is also repeated in the other three directions with the same δ. Finally, this process is done for all values of δ.

We call the execution of the outer loop (line 1 to 10) a *cycle*. Note that each time we update f on line 9, the energy E_f, decreases or remains the same. At each *cycle* before the last, the energy decreases by at least the minimum cost of changing the label of a pixel, thus ensuring convergence. In practice, convergence is obtained after only a few cycles. After the last, a local minimum with respect to a *border move* is found.

To compute the value of the *discontinuity* likelihood term for all possible discontinuity locations, we only need to scan all the pixels on a line once. Since only two pixels have their disparity changed when the discontinuity is moved by one pixel, the likelihood term e_f can be efficiently recursively computed. The same technique can be used for the computation of the smoothing term. Consequently, each *border move* is in $\Theta(\#\mathcal{L} \cdot N^2)$ when Eq. 7.2 is minimized using dynamic programming. In our tests, we used segments of up to 19 pixels.

The algorithm may remove disparity discontinuities, but never adds new ones. Discussion about initialization is postponed until section 7.5.1.

7.4 Visibility

To model occlusion, we must compute the volumetric visibility $V_c(\mathbf{p}, \delta, f)$ of a 3D reference point formed from the pixel $\mathbf{p}$ at disparity δ from the point of view of a camera c, given a disparity configuration f. It is set to 1 if the point is visible, and 0 otherwise. The visibility information is collected into a vector, the *visibility mask*

$$V(\mathbf{p}, \delta, f) = (V_1(\mathbf{p}, \delta, f), \ldots, V_C(\mathbf{p}, \delta, f))$$

where C is the number of cameras outside the reference; a vector $(1, \ldots, 1)$ means that the 3D point is visible in all supporting cameras. We call $\mathcal{M}$ the set of all possible visibility masks; an $\mathcal{M}$-configuration $g : \mathcal{P} \mapsto \mathcal{M}$ associates a mask to every pixel. The visibility masks are *correct* when g satisfies

$$g(\mathbf{p}) = V(\mathbf{p}, f(\mathbf{p}), f) \tag{7.3}$$

for all $\mathbf{p} \in \mathcal{P}$. As in [17], we transform Eq. 7.1 into an energy function with masks

$$E(f, g) = \sum_{\mathbf{p} \in \mathcal{P}} e(\mathbf{p}, f(\mathbf{p}), g(\mathbf{p})) + \text{smoothing}$$

and Eq. 7.2 into

$$E_{f'}(b, g) = \sum_{\lambda \in \mathcal{L}} e_{f'}(\lambda, b(\lambda), g|_\lambda) + \text{smoothing}. \tag{7.4}$$

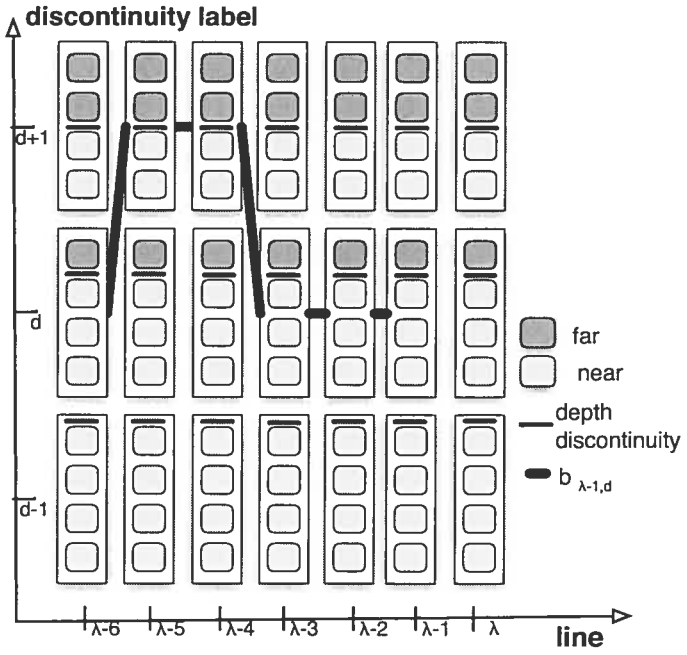

Figure 7.3: DP matching process. To determine the best discontinuity map up to line λ with discontinuity at d, for the various discontinuity locations on $\lambda - 1$, we look at the best solution up to $\lambda - 1$, available by construction. An example of a $b_{\lambda-1,d}$ is illustrated.

Mutatis mutandis, our new $e_{f'}$ is the *discontinuity* likelihood term that uses the pixel likelihood with masks. Since we expect masks belonging to adjacent pixels to be spacially coherent, we can add a visibility smoothing term that penalizes the change of visibility status between adjacent pixels. However, no visibility smoothing was used in the results presented in this paper.

7.4.1 Optimizing discontinuity and visibility

The algorithm works by using dynamic programming (DP) applied perpendicularly to the orientation of the lines in $\mathcal{L}$. For instance, if the lines are vertical then DP proceeds horizontally either from left to right or right to left. We illustrate this for

vertical lines processed from left to right. The cameras are assumed to be in a cross-shaped configuration, with the reference in the center and the 4 supporting images at an equal distance from it. The order of the cameras in the mask is left, right, top and bottom.

When dynamic programming attempts to compute the location of the discontinuity of line λ, it can rely on knowledge of the location of the discontinuities of all preceding lines (Fig. 7.3). Because of this, the visibility between any camera to the left of the reference and the 3D point formed by pixels $\mathbf{p}_i^\lambda$ at any disparity is also known (Fig. 7.4). Two similar strategies for binocular and multi-camera stereo were presented in [2, 15] respectively. These approaches minimize Eq. 7.1 directly, whilst we minimize Eq. 7.2, allowing the use of much more visibility information. Since the disparity of *passive* pixels is fixed during the minimization process, the visibility of the bottom and top cameras is known for each pixel of the line being examined. Figure 7.4 illustrates the visibility information available when processing a disparity map with our Border-Cut algorithm.

When solving the discontinuity localization problem, two 2-dimensional tables t and t' are filled out; $t(\lambda, d)$ is the lowest energy of all discontinuity maps of lines 0 to λ with line λ having its *discontinuity* at d; $t'(\lambda, d)$ is the index of the *discontinuity* of $\lambda - 1$ given by this map of lowest energy, denoted $b_{\lambda,d}$. A sample discontinuity map $b_{\lambda-1,d}$ is highlighted in Fig. 7.3 and is also shown in Fig. 7.4. The table t' is used to compute the different $b_{\lambda,d}$'s.

Explicitly, the tables t and t' are defined inductively as

$$\begin{aligned} t(0,d) &= e_{f'}(0,d,g|_{\lambda_0}) \\ t'(0,d) &= d \end{aligned}$$

and for $\lambda > 0$

$$t(\lambda, d) = \min_{d' \in \mathcal{B}} \begin{pmatrix} & e_{f'}(\lambda, d, g|_\lambda) \\ + & s_{f'}(\lambda - 1, \lambda, d', d) \\ + & r_{\mathcal{R},f'}(\lambda, d) \\ + & t(\lambda - 1, d') \end{pmatrix}$$

$t'(\lambda, d)$ is the index of the minimum in the above formula

where $r_{\mathcal{R},f'}(\lambda, d)$ is the smoothing between the pixels of line λ having its *discontinuity* at d and *passive* pixels surrounding it. The visibility masks are

$$g(\mathbf{p}_i^\lambda) = (O_1(i, \lambda, d, d'), \ldots, O_4(i, \lambda, d, d'))$$

where $O_c(i, \lambda, d, d')$ is a visibility function that is equal to 1 iff the camera c is visible from pixel $\mathbf{p}_i^\lambda$ (see next section). For most c's, it only requires the knowledge of $b_{\lambda-1,d'}(\lambda')$ for $\lambda' < \lambda$ and of *passive* pixels. The $b_{\lambda,d}$'s can be computed with the relations

$$\begin{aligned} b_{\lambda,d}(\lambda) &= d \\ b_{\lambda,d}(\lambda') &= t'(\lambda' + 1, b_{\lambda,d}(\lambda' + 1)) \quad \text{for} \quad 0 \leq \lambda' < \lambda. \end{aligned}$$

It is thus possible to compute $b_{\lambda-1,d'}(\lambda')$ for all $\lambda' < \lambda$ and $d' \in \mathcal{B}$. This allows us to compute visibility $O_c(i, \lambda, d, d')$ for all d' and for most of the c's and finally $t(\lambda, d)$.

7.4.2 Computing visibility

We define $O_c(i, \lambda, d, d')$ as the visibility of camera c for pixel $\mathbf{p}_i^\lambda$, for the *discontinuity* map defined on lines 0 to λ having its *discontinuity* on line λ at d and having the lowest energy amongst those with *discontinuity* d' on $\lambda - 1$, i.e. the disparity map $b_{\lambda',d'}$ extended to λ by placing its *discontinuity* at d. We show how to compute the

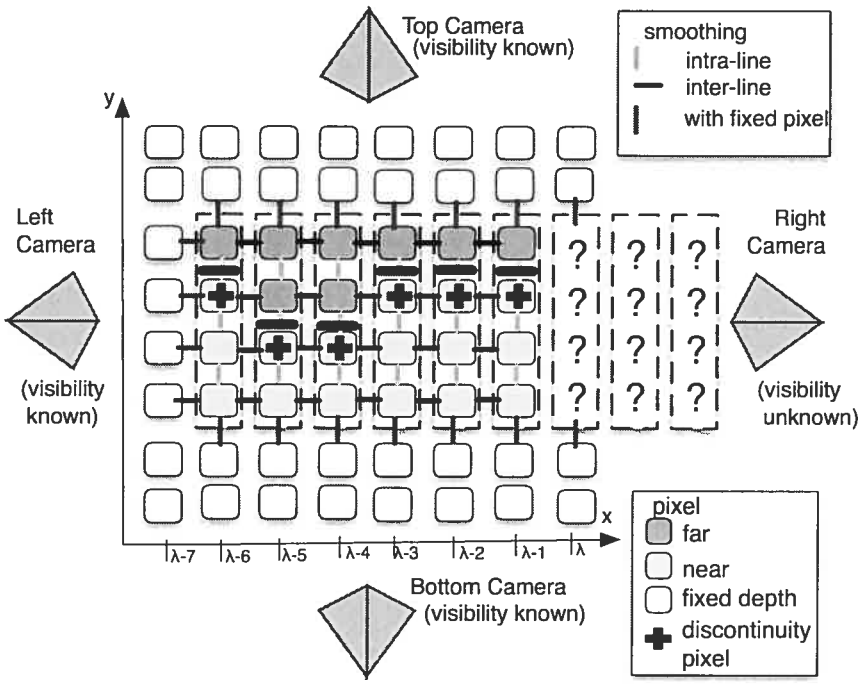

Figure 7.4: Visibility information available when searching for the location of the *discontinuity* pixel of vertical line λ using DP from left to right. The visibility of the left camera is available by construction whilst right visibility is unknown since border localization for lines to the rigth of λ is not yet computed. Top and bottom visibilities are always available since they only depend on disparity of pixels on the line being examined and of *passive* pixels.

visibility function O_1 for the left camera assuming that all segments are perfectly aligned, i.e. there is no *passive* pixel in the bounding rectangle of the region $\mathcal{R}$.

We first introduce the auxiliary function $S_1(i, \lambda, d)$ that represents how far the *shadow* of all the pixels to the left of $\mathbf{p}_i^\lambda$ and $\mathbf{p}_i^\lambda$ itself goes, for the discontinuity map of lowest energy defined on lines up to λ with a discontinuity on λ at d. Explicitly,

$$S_1(i, 0, d) = \max\left\{\max_{\lambda'<0}\left(f|_{\mathcal{R}^C}(\mathbf{p}_i^{\lambda'}) + \lambda'\right), F_{b_{0,d},f'}(\mathbf{p}_i^0)\right\}$$

$$S_1(i, \lambda, d) = \max\left\{\begin{array}{l} S_1(i, \lambda - 1, b_{\lambda,d}(\lambda - 1)), \\ F_{b_{\lambda,d},f'}(\mathbf{p}_i^\lambda) + \lambda \end{array}\right\} \text{ for } \lambda > 0$$

where the negative λ''s are the lines of *passive* pixels to the left of the first (zeroth!) active one (of course $F_{b_{\lambda,d},f'}$ is only defined on lines 0 to λ). When the segments are not aligned, the disparity of *passive* pixels can be needed to compute S_1 even for λ greater than 0. We leave the modifications to the reader. This recursive computation of S_1 used a continuous mesh representation of the disparity map. A consequence of this is that the ordering constraint is applied in the selection of visibility masks, but not on the disparity map itself. Now, O_1 is simply defined as

$$O_1(i, \lambda, d, d') = \begin{cases} 1 & \text{if } \delta_{f'}(i, \lambda, d) + \lambda > S_1(i, \lambda - 1, d') \\ 0 & \text{otherwise} \end{cases}$$

where $\delta_{f'}(i, \lambda, d)$ is the disparity of $\mathbf{p}_i^\lambda$ when the discontinuity of λ is at d, given f'. O_1 is equal to 1 when the left camera is visible. The corresponding relations for the top and bottom ones are obtained similarly except that they only use disparity of the pixels of the current line and of *passive* pixels. For the right camera, no visibility information is available. Because it has been noticed that removing a camera that is visible is far less damaging than keeping a camera that is not [79], the right camera is only used when no others are. In multi-baseline stereo, it is reasonable to expect every pixel of the reference to be visible by at least one supporting camera. This use of a heuristic can make the Border-Cut algorithm oscillate, but after one or two cycles, changes are negligible.

Note that when the cameras are located on only 2 non opposite sides of the reference (for example top and left), the direction for which DP processes horizontal and vertical lines can always be chosen so that the visibility is handled *correctly*. With these camera configurations, the Border-Cut algorithm finds a local minimum, with respect to a *border move*, to energy function 7.4 respecting the constraint 7.3. The pixel likelihood term takes a user-defined occlusion cost when no camera is visible. Note that during a *border move*, the masks of *passive* pixels may change. This should be considered in the term $r_{\mathcal{R},f'}$. However in our tests, ignoring this did not have an impact on the quality of the solutions. The explanation comes from the fact that close objects are generally enlarged by stereo matchers [17]. Since *border moves* usually correct this, *passive* pixels have a tendency to regain cameras rather than the opposite, a phenomenon that has little impact as mentioned above [79].

7.5 Experimental results

In all our experiments, the matching cost function used came from [59] which is based on [4]. We used color images but only the references in gray scale are shown here. As for the smoothing term, we used the experimentally defined smoothing function that also comes from [59]:

$$s(\mathbf{p}, \mathbf{p}', f(\mathbf{p}), f(\mathbf{p}')) = \gamma\, h(\mathbf{p}, \mathbf{p}')\, l(f(\mathbf{p}) - f(\mathbf{p}'))$$

where h is defined as

$$h(\mathbf{p}, \mathbf{p}') = \begin{cases} 3 & \text{if } |I_{ref}(\mathbf{p}) - I_{ref}(\mathbf{p}')| < 5 \\ 1 & \text{otherwise} \end{cases}$$

where l is 1 at 0 and 0 elsewhere. $I_{ref}(\mathbf{p})$ is the intensity of pixel $\mathbf{p}$ in the reference image. The parameter γ is user-defined. A pixel disparity is considered erroneous if it differs by more than one step from the ground truth. This error measurement was used in two comparative studies [105, 94].

7.5.1 Tsukuba Head and Lamp

This dataset is from the Multiview Image Database from the University of Tsukuba (see Fig. 7.5). It is composed of a 5×5 image grid. Each image has a resolution of 384×288. The search interval was between 0 and 15 pixels and we used 16 disparity steps. The 5-camera configuration is a cross with 4 cameras equidistant to the center one (the reference). The 3-camera configuration is obtained from the previous one by removing the top and bottom cameras. Some disparity maps are shown in Fig. 7.5 and error percentages are given in Table 7.1. The entries ASYM-KZ1 and KZ1' come directly from [119]. HYBRID-IDP and NAKA-BNV come from [15]. The disparity map for the latter was not available, so we could not use it to initialize our algorithm. GEO-BNV and REL-DP come from [17] and [36] respectively. All these occlusion modeling methods are multi-baseline and used 5 cameras. Our algorithm lowered the error rate with all initializations. We also initialized with the current best two algorithms (SEG+VIS [5], SYMBP+OCC [102]) of the Middlebury Stereo Evaluation - (Version 2) [93]. The fast TREEDP based on dynamic programming was used as well [113]. We provide results obtained with a fixed smoothing parameter for all the different initializations, with the 3- and 5-camera configurations. We also provide those computed with the best parameter for each disparity map computation. Of interest is the fact that results are comparable no matter which initial stereo matcher was used, even when this matcher only utilized two cameras without occlusion modeling. Only one *cycle* was performed with segments of up to 11 pixels; the running time was below 19 seconds on an AMD Athlon(tm) 64 Processor 3500+, even though we did not compute $e_{f'}$ and $s_{f'}$ recursively as described at the end of section 7.3.1. Note in Fig. 7.5 how a few pixels with correct disparity between the branches of the arm of the lamp were enough to allow Border-Cut to recover the disparities of this region correctly. When using three cameras with the fixed parameter, we could not reduce the error rate of SEG+VIS and KZ1'. Nevertheless, we saw a significant improvement

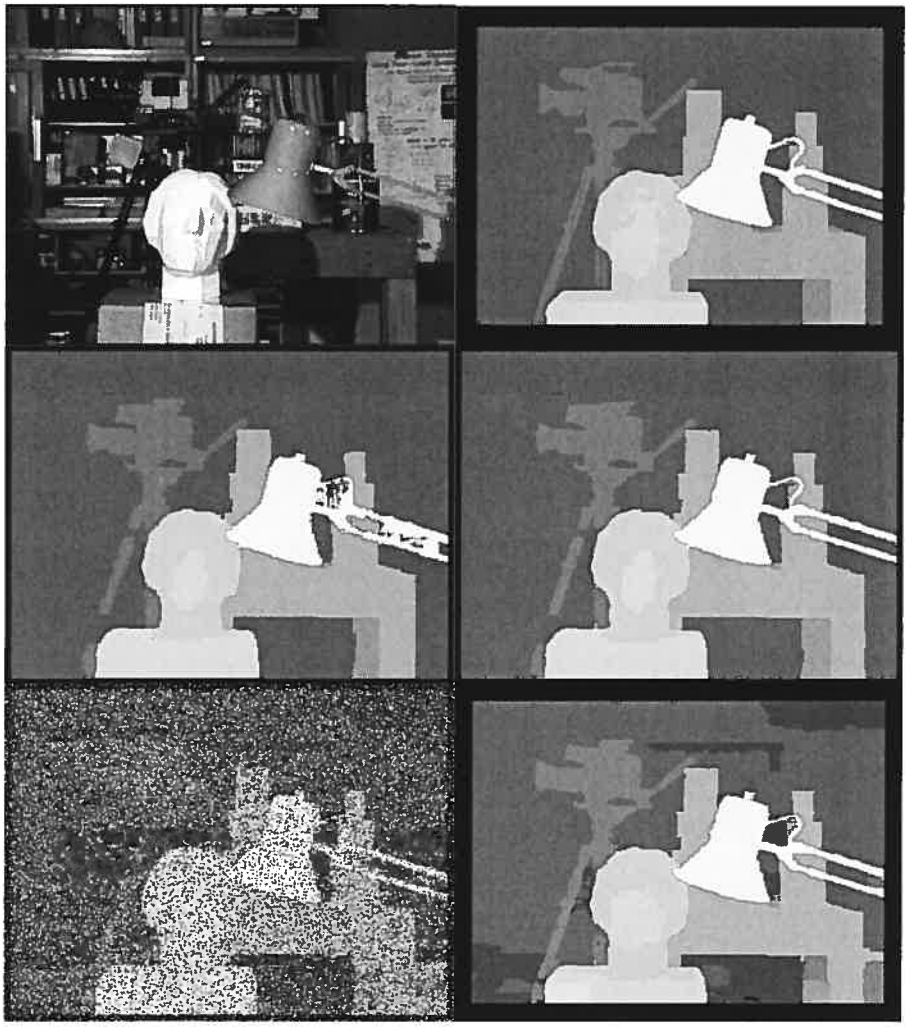

Figure 7.5: Reference images for the Head and Lamp (top left) and ground truth (top right) from the Multiview Images database of the University of Tsukuba. Disparity map obtained from GEO-BNV (middle left) and the improvement obtained after applying our algorithm (middle right). Disparity map of Hybrid-IDP with 36 % corruption (bottom left) and result after applying our algorithm (bottom right).

Initialization algorithm with number of cameras used	Before B-C	After B-C			
		5 cameras		3 cameras	
		fixed γ	best γ	fixed γ	best γ
NAKA-BNV(5 cam's)	1.70	-	-	-	-
GEO-BNV(5 cam's)	2.23	1.17	1.07	1.48	1.44
HYBRID-IDP(5 cam's)	1.67	1.09	1.09	1.17	1.17
REL-DP(5 cam's)	1.86	1.13	1.07	1.24	1.11
KZ1'(5 cam's)	1.28	1.06	1.04	1.77	1.19
ASYM-KZ1(5 cam's)	1.30	1.11	1.08	1.09	1.08
SEG+VIS(2 cam's)	1.57	1.11	1.07	1.69	1.08
SYMBP+OCC(2 cam's)	1.75	0.96	0.96	1.02	1.02
TREEDP(2 cam's)	2.84	1.04	1.04	1.05	0.96
Average	-	1.08	1.05	1.31	1.13

Table 7.1: Percentage of error in disparity for all pixels of the different algorithms for Head and Lamp scene, before and after applying Border-Cut, using 3 and 5 images.

when the best parameter was used. On this dataset, the sensitivity to initialization and to change of the smoothing parameter is reduced when the number of cameras increases from three to five. The reduced sensitivity of multi-baseline stereo to change of the smoothing parameter was also mentioned in [119].

Figure 7.6 shows the sensitivity to change of the smoothing parameter and to the quality of the initial disparity map of our algorithm, using HYBRID-IDP for initialization. The probability of a pixel to be *corrupted* (i.e. having its disparity changed to one picked at random from a uniform distribution) ranges from 0 to 80%. When used with multi-baseline camera configurations, our algorithm is very stable to change of the smoothing parameter. Without *corruption*, the error stays below 3.7% for γ ranging from 0 to 100. Moreover, even when pixels have a 36% chance of being *corrupted*, the error rate remains below 1.40% (see Fig. 7.5). It is thus very resistant to *corruption* of the initial disparity map even when a single *cycle* is performed. Since our approach does not add new discontinuities but may remove existing ones and since it is robust to *corruption*, we can start with an under-smoothed initial disparity map

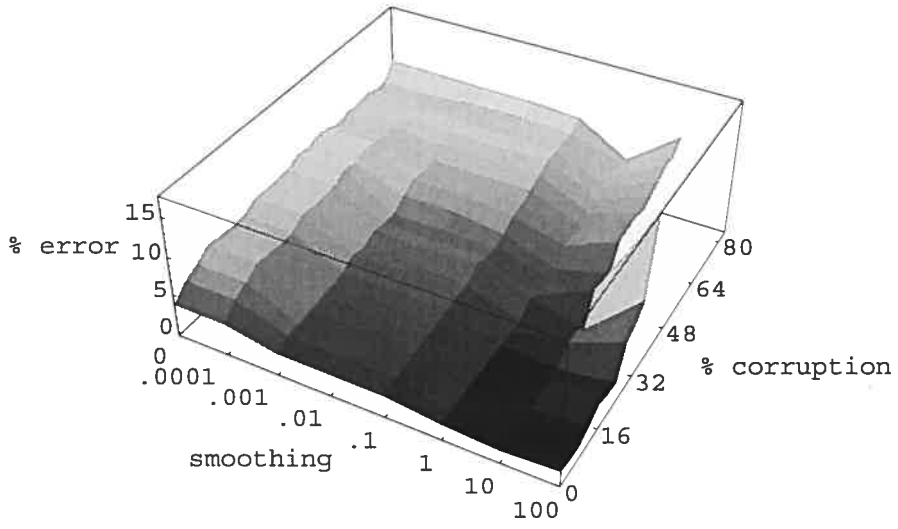

Figure 7.6: Variation of the error rate as a function of smoothing and pixel *corruption*.

in order to avoid missing small objects. In our tests, this was not necessary.

7.5.2 City scene

This dataset from the Multiview Image Database of the University of Tsukuba (Fig. 7.7) contains 81 images of 640×480 in a 9×9 grid. We only used 5 images in a cross configuration. Each disparity map was computed using 44 disparity steps and the search interval was between 0 and 43 pixels. Two *cycles* were performed with segments of up to 11 pixels; the running time was below 92 seconds on an AMD Athlon(tm) 64 Processor 3500+. Some disparity discontinuities present in the reference were manually segmented to create a partial *discontinuity* ground truth. A discontinuity location is considered erroneous if it differs by more than one pixel from the ground truth.

This is motivated by the fact that discontinuities in a scene are not perfectly aligned with the pixel grid. The different 5-camera algorithms used to initialize Border-Cut come from [17] and [15], whilst the 2-camera algorithms BNV and MF came from [8] and [89] respectively. The results are presented in Table 7.2 and some disparity maps are shown in Fig. 7.7. The percentage of pixels with a change in disparity greater than one for GEO-MF before and after Border-Cut is only 0.71%. Nevertheless, the improvement in border localization after applying Border-Cut is significant. Note that a correct discontinuity location does not imply that the disparities on both sides are correct as well.

7.5.3 Middlebury Binocular Comparative Study

To improve border localization, our Border-Cut algorithm is more appropriate for multi-baseline stereo configurations, since in binocular stereo, occlusion affects the localization of depth discontinuities on only one side of objects. Nevertheless, we provide results for the stereo pairs of the new Middlebury comparative study. We use as initialization the results obtained from BNV-OCC, KZ1, BNV and TREEDP coming from [57],[59], [94]-c and [113] respectively. The first two methods model occlusion, whilst the others do not. They all use the same smoothing model and similar cost function as that of our Border-Cut algorithm. Results are shown in Table. 7.3. Border-Cut, when used on binocular stereo, is more sensitive to change of the smoothing parameter and occlusion cost (the latter is not even used in multi-baseline Border-Cut). The algorithm is also more sensitive to the initialization. Nevertheless, the average error reduction for BNV-OCC and KZ1 are repectively 0.48 and 0.21. For BNV and TREEDP, the error reduction is 1.61 and 1.88 respectively. As expected, the error reduction is more significant for algorithms that do not model occlusion. For TREEDP the error in disparity in discontinuity regions (as defined by the Middlebury Stereo Evaluation - Version 2) for the Venus scene was reduced from 7.74% to 2.22% (see Fig. 7.8). The improvement in discontinuity regions is even greater

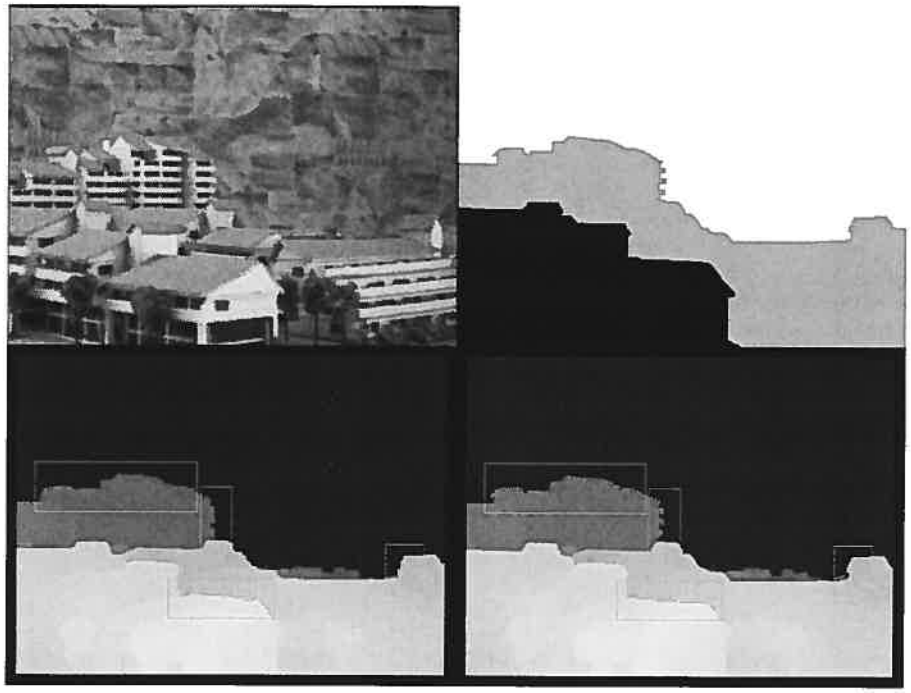

Figure 7.7: Reference image for the City scene (top left) from the Multiview Images database of the University of Tsukuba. Some manually segmented disparity discontinuities (top right). Disparity map obtained from GEO-MF (bottom left) and improved by Border-Cut (bottom right). Regions of interest are framed (see high resolution images in the electronic version of this paper for small details).

Algorithm	Before B-C (best γ)	After B-C (fixed γ)
FULL-MF (5 cam's)	22.3	11.7
FULL-BNV (5 cam's)	19.8	11.1
GEO-MF (5 cam's)	23.2	10.4
GEO-BNV (5 cam's)	15.4	11.4
Hybrid-IDP (5 cam's)	28.2	14.7
BNV (2 cam's)	21.4	9.8
MF (2 cam's)	30.5	9.8

Table 7.2: Percentage of error in the discontinuity location of the different algorithms for City scene, before and after Border-Cut, using 5 images.

for algorithms known for having trouble with them, such as the one of [89]. For this particular case, the error rate was reduced from 21.5% to 4.61%. The increase of the error rate on the Tsukuba scene for KZ1 is caused by the fact that the same smoothing parameter was used for all four scenes, the one minimizing the average error.

7.6 Conclusion

We have presented a new algorithm to improve discontinuity localization of the disparity maps obtained from multi-baseline stereo. From an initial disparity map, the algorithm associates a position to every disparity discontinuity, instead of a disparity label to every pixel. This formulation allows to find an approximated solution to a 2D labeling problem with a robust smoothing term by minimizing multiple 1D problems. We showed how to include efficiently visibility computation into each 1D minimization using dynamic programming. Our method obtains sharp and well-located disparity discontinuities starting from the output of a wide range of stereo matchers. For binocular algorithms and those that do not model occlusion, the improvement after applying *Border-Cut* is significant to the point where the final results are indistinguishable from those obtained after initializing with multi-baseline meth-

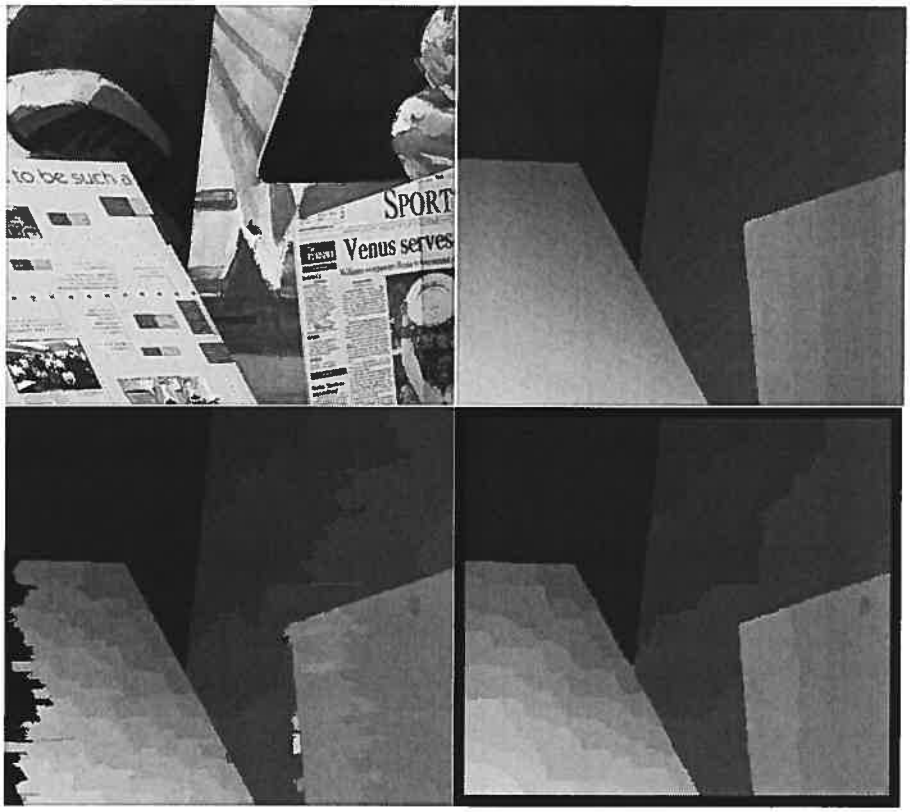

Figure 7.8: Reference image for the Venus scene (top left) from the Middlebury Comparative study. Ground truth (top right). Disparity map obtained by TREEDP (bottom left) and improved by Border-Cut using 2 cameras (bottom right).

ods with occlusion modeling. Moreover, our framework, when used with certain camera configurations, guarantees the *correct* visibility. The validity of our framework was demonstrated on standard datasets with ground truth and was compared to other state-of-the-art multi-baseline stereo matchers. For future work, better ways of defining and extracting borders should be found. Applications of the Border-Cut algorithm to other vision problems should also be investigated.

Algorithm	Tsukuba		Venus		Teddy		Cones	
	B	A	B	A	B	A	B	A
BNV-OCC	2.01	1.97	2.19	1.52	17.4	17.1	12.4	11.5
KZ1	1.99	2.31	3.13	3.09	17.6	17.6	11.8	10.7
BNV	4.12	2.62	4.33	2.81	25.0	23.9	18.2	15.9
TREEDP	2.84	1.92	2.10	0.81	23.9	20.9	18.3	16.0

Table 7.3: Percentage of error in disparity for all pixels of the different algorithms from the new Middlebury comparative study, before (B) and after (A) Border-Cut, using 2 images and fixed parameters.

7.7 Acknowledgment

This work was made possible by NSERC (Canada) and NATEQ (Québec) grants.

Chapitre 8

INTRODUCTION À LA LUMIÈRE STRUCTURÉE

La stéréoscopie et la lumière structurée sont des moyens d'établir la correspondance entre les pixels de deux vues différentes d'une même scène. Dans le premier cas les deux vues sont des caméras, alors que dans le second une caméra est remplacée par un projecteur. La possibilité de *modifier* la scène avec le projecteur permet de réduire considérablement l'ambiguïté lors de la mise en correspondance. L'utilisation de lissage tel que fait en stéréoscopie n'est généralement pas nécessaire. Comme il sera démontré dans le chapitre 9, l'application de lissage peut même réduire la qualité des mises en correspondance. Un système de lumière structurée classique composé d'un projecteur et d'une caméra est illustré à la figure 8.1. Le projecteur modifie l'apparence de la scène en projetant des motifs de lumière connus dans le but d'établir la mise en correspondance entre les pixels du projecteur et ceux de la caméra. Par exemple, dans la figure 8.1, les pixels rouges de la caméra doivent venir nécessairement d'un objet de la scène illuminé par l'un des deux pixels rouges du projecteur. Afin de déterminer quel pixel rouge du projecteur est vu par un pixel de la caméra, il suffit de projeter un second motif dans lequel l'un des pixels rouges du projecteur est éteint. Il devient alors possible d'établir la correspondance de manière unique. Évidemment, la scène ne doit pas être de couleur rouge. En lumière structurée, il est généralement préférable d'avoir une scène de couleur unie et pâle afin de distinguer les motifs projetés. La lumière structurée contraste avec la stéréoscopie où de riches textures facilitent la mise en correspondance alors que les zones unies la rende plus difficile. Comme en stéréoscopie, la mise en correspondance en lumière structurée est simplifiée lorsque la matrice fondamentale est connue. La figure 8.2 illustre un système à lumière structurée quand la géométrie épipolaire est connue. Puisque la

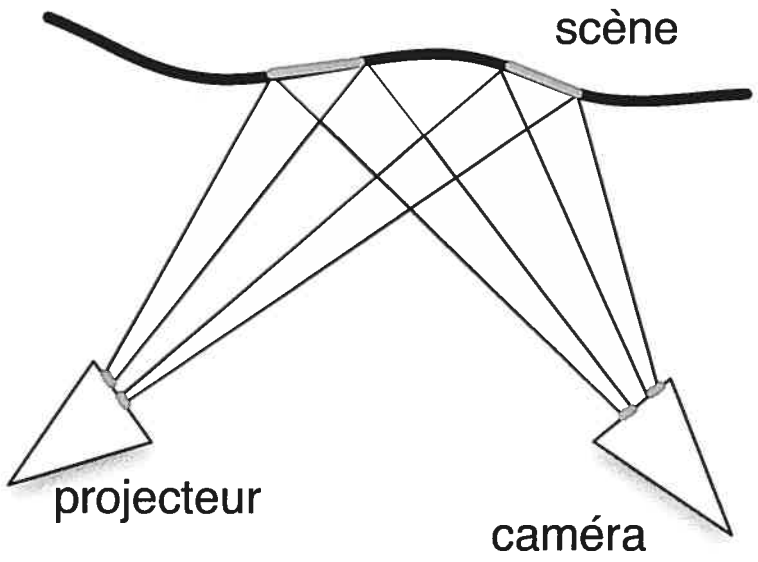

FIG. 8.1 – Système à lumière structurée.

ligne épipolaire sur laquelle se trouve le pixel $\mathbf{q}_2$ de la caméra est connue, il suffit de projeter des motifs permettant de différencier les différentes colonnes du projecteur. Connaissant la ligne épipolaire dans le plan image du projecteur et la colonne du projecteur correspondant au point $\mathbf{Q}$, on peut donc intercepter ces deux lignes afin d'obtenir une mise en correspondance. Évidemment, si les colonnes du projecteur sont parallèles aux lignes épipolaires, il faudra projeter des motifs permettant de différencier les lignes du projecteur afin d'obtenir une mise en correspondance. Dans la prochaine section, nous présenterons brièvement les différentes classes de méthodes de lumière structurée.

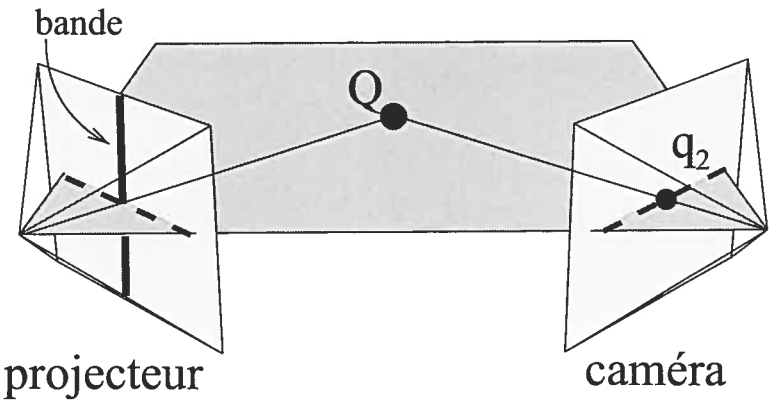

FIG. 8.2 – Système à lumière structurée calibrée. Un motif comprenant une seule bande verticale permet d'établir la correspondance entre le point q_2 et un pixel du projecteur.

8.1 Classification

Un article publié en 2004 par Salvi *et al.* [91] présente une classification des différentes méthodes de lumière structurée. Nous décrirons brièvement les 3 grandes classes qui ont été établies et nous fournirons un exemple de codification pour chacune.

8.1.1 Codification directe

La codification directe permet d'établir la mise en correspondance en projetant une seule image. La figure 8.3 contient un exemple simplifié de codification directe. L'image du projecteur a une résolution de 256 colonnes. Chaque colonne a une intensité unique. L'alphabet du code comporte 256 symboles distincts. Le code permet d'enlever l'ambiguïté sur un seul des axes du plan image du projecteur parce que tous les pixels de la même colonne ont le même code. Il est évidemment possible de créer un code permettant de rendre les lignes non ambiguës. Un des problèmes de cette codification est le suivant : les projecteurs et les caméras distordent le signal de

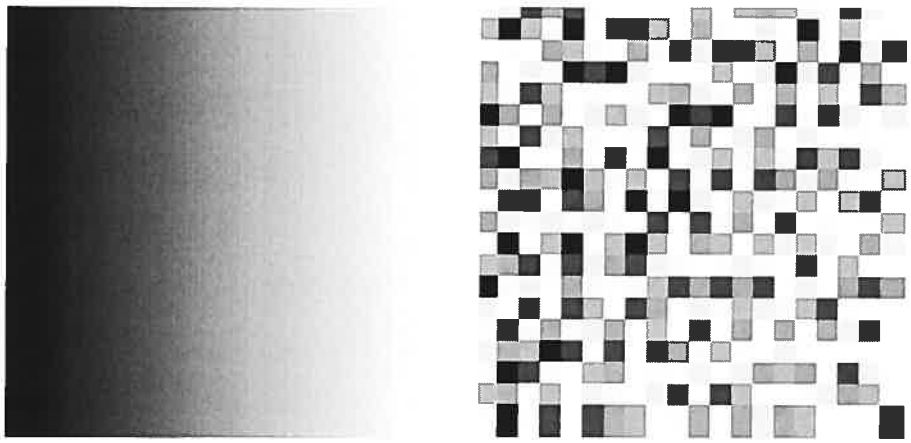

FIG. 8.3 – Exemples de codifications directe et spatiale.

manière non-linéaire. Il faut donc connaître les courbes de réponses des caméras et des projecteurs. Ce type de méthode est également très sensible au bruit.

8.1.2 Codification spatiale

Dans le codage spatial, la résolution spatiale du projecteur est sacrifiée afin d'encoder en une seule image et avec un alphabet limité suffisamment d'information. Ce qui permet un décodage sans ambiguïté. La figure 8.3 contient un exemple de codification spatiale. Le motif utilise 8 couleurs (symboles de l'aphabet) et chacun des blocs de 2×2 du motif est unique. Il est important de noter que ce motif n'est pas invariable en rotation. Ce type motif est particulièrement utile lorsque la matrice fondamentale est inconnue. Il encode de l'information permettant de décoder sans ambiguïté les lignes et les colonnes du projecteur.

8.1.3 Codification par multiplexage temporel

Ce type de codification permet d'établir la mise en correspondance avec un alphabet limité. Il utilise plusieurs motifs projetés successivement sans sacrifier la résolution

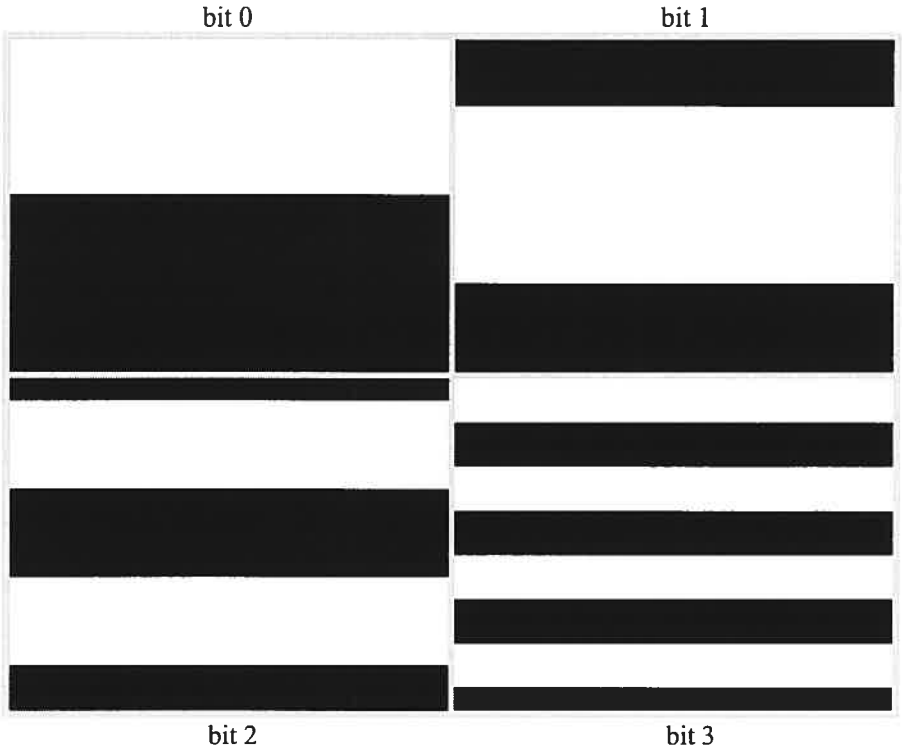

FIG. 8.4 – Exemple de codification par multiplexage temporel. Quatre motifs du code de Gray sont illustrés.

spatiale du projecteur. Avec un alphabet de M symboles différents et N différents pixels à distinguer, il faut $\log_M N$ images de lumière structurée. Le codage direct est un cas particulier où $M = N$. Dans la prochaine section un codage temporel extrêmement répandu avec un alphabet de 2 symboles sera présenté. Il s'agit du code de Gray que nous utiliserons au chapitre 9. Le code permet de rendre non ambigu un seul des axes puisque tous les pixels de la même colonne (ou ligne) ont le même code.

8.2 Code de Gray

Le code de Gray est défini par *National Institute of Standards and Technology* comme étant une suite de nombres binaires dans lequel seulement un bit change

entre deux éléments successifs de la suite. Il existe plusieurs suites qui respectent cette définition. Lw tableau 8.1 contient deux suites représentant un code de Gray de 4 bits (16 éléments).

Le premier usage technologique du code de Gray remonte à 1878. Il a été utilisé dans une application de transmission télégraphique par Émile Baudot. C'est en 1953, qu'il a été breveté au nom de Frank Gray du *Bell Labs* [38]. Un système de lumière structurée utilisant le code de Gray a été présenté en 1984 [45]. On utilise généralement des bandes noires et blanches afin de représenter les deux symboles de l'alphabet. La figure 8.4 illustre 4 motifs du code de Gray.

8.2.1 Algorithme

Puisqu'il peut exister plus d'une suite qui respecte la définition du code de Gray, il peut donc exister aussi plus d'un algorithme qui génère une suite valide. Nous présenterons un algorithme permettant de produire la suite gauche du tableau 8.1. Il s'agit de l'algorithme qui a été utilisé. Il est une adaptation de l'algorithme proposé par Dan T. Abell et est disponible au `http://www.faqs.org/faqs/ai-faq/genetic/part6/section-1.html`. La figure 8.5 contient une version en pseudo-code des algorithmes d'encodage et de décodage.

Dans les applications de lumière structurée, il arrive fréquemment que les motifs contenant les bits de poids faibles soient rendus indécodables par les différentes sources de bruits. La suite générée par l'algorithme de la figure 8.5 possède la caractéristique suivante : lorsque les N motifs correspondant au bit de poids faibles sont absents, les valeurs binaires possibles forment un seul intervalle de 2^N valeurs consécutives.

8.3 Applications

La lumière structurée permet d'établir la correspondance entre les pixels d'une caméra et ceux d'un projecteur. Il existe une multitude d'applications qui utilisent

Suite 1		Suite 2	
Binaire	Décimal	Binaire	Décimal
0000	0	0110	6
0001	1	0100	4
0011	3	0101	5
0010	2	0111	7
0110	6	0011	3
0111	7	0010	2
0101	5	0000	0
0100	4	0001	1
0110	12	1001	9
0111	13	1000	15
1111	15	1010	10
1110	14	1011	11
1010	10	1111	15
1011	11	1101	13
1001	9	1100	12
1000	8	1110	14

TAB. 8.1 – Exemple du code de Gray. Deux suites respectant la définition du code de Gray tel que proposé par *National Institute of Standards and Technology.*

```
CONVERSION GRAY VERS BINAIRE(gray[0...n])
1  bin[0]←gray[0]
2  for i ← 1; i ≤ n; i++
3  do bin[i]←bin[i-1] XOR gray[i]
4  return bin

CONVERSION BINAIRE VERS GRAY(bin[0...n])
1  gray[0]←bin[0]
2  for i ← 1; i ≤ n; i++
3  do gray[i]←gray[i-1] XOR bin[i]
4  return gray
```

FIG. 8.5 – Pseudo-code permettant la conversion du code de Gray vers le code binaire et vice-versa.

les mises en correspondance obtenues par la lumière structurée. Nous en présenterons deux : le calibrage et la reconstruction 3D.

Calibrage

Dans le chapitre 9, nous présenterons une procédure permettant de calibrer un projecteur à l'aide d'un système de lumière structurée utilisant le code de Gray et projetant des bandes verticales et horizontales. Une fois qu'un système projecteur et caméra est calibré, il devient possible d'effectuer des reconstructions 3D.

Reconstruction 3D

Lorsque la matrice fondamentale entre le projecteur et la caméra est connue, il est possible d'utiliser la lumière structurée afin d'établir la correspondance entre un pixel du projecteur et un pixel de la caméra. Pour cela, il s'agit d'utiliser une seule orien-

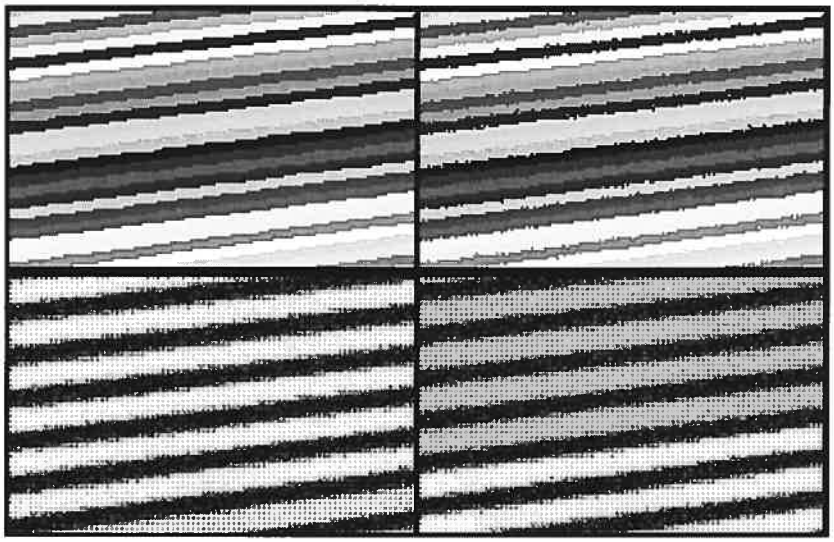

FIG. 8.6 – Résultats de la méthode active proposée. Haut gauche) méthode proposée ; Haut droite) décodage conventionnel ; Bas) images normalisées du motif. Les vraies intensités varient de 0 à 23.

tation de la bande. Il faut cependant s'assurer que l'orientation choisie ne se confond pas à celle de la ligne épipolaire. Une fois que la mise en correspondance est établie, il suffit de trianguler pour obtenir un modèle 3D. Lorsque les paramètres internes du projecteur et de la caméra sont connus on obtient une reconstruction euclidienne à un facteur d'échelle. Lorsqu'on connaît uniquement la matrice fondamentale, on doit se contenter d'une reconstruction projective.

8.4 Contribution

Une méthode basée sur la minimisation d'énergie et qui fonctionne en temps réel sur un processeur graphique programmable sera présentée au chapitre 9. La méthode a la capacité d'effectuer le décodage du code de Gray sur n'importe lequel des plans du monde. Ceci est possible lorsque l'homographie entre la caméra et le plan est connue. De plus ça permet de généraliser la méthode à une configuration

multi-caméras, utile lors du calibrage d'un projecteur à l'aide de la méthode de calibration plan. La figure 8.6 illustre le résultat du décodage d'un code de Gray en utilisant la méthode proposée dans des conditions d'acquisition difficile. La scène est un plan incliné. Puisque la linéarité est préservée par une transformation projective, les différentes lignes de lumière structurée devraient demeurer des lignes dans le plan image de la caméra. Notre méthode préserve la linéarité, mais elle ne l'est pas dans le décodage conventionnel.

Chapitre 9

REAL-TIME GEOMETRICALLY PLAUSIBLE ENERGY-BASED STRUCTURED LIGHT MATCHING

Marc-Antoine Drouin, Guy Godin and Sébastien Roy, Real-Time Geometrically Plausible Energy-Based Structured Light Matching,soumis

Marc-Antoine Drouin	Conseil National de Recherches Canada
	Université de Montréal
Guy Godin	Conseil National de Recherches Canada
Sébastien Roy	Université de Montréal

Abstract

This paper presents a new real-time energy-based formulation of the matching problem encountered when establishing the correspondence using a stripe-based structured light system. We provide a general framework that takes into consideration the projective displacement which is induced by the camera, projector and scene configuration. This allows the selection of a neighborhood for the smoothing term which is geometrically plausible. The minimization is performed using a GPU-oriented reformulation of the dynamic programming recurrences. We experimentally show that our approach provides significant improvements under difficult acquisition conditions. Our method is also applied to the problem of projector calibration using a planar target and multiple cameras. The results are validated using a laser tracker.

9.1 Introduction

This paper aims at providing an energy formulation of the matching problem encountered in stripe-based time-multiplexing structured light systems, when establishing the correspondence between stripes of pixels of a projector and pixels of one or several cameras. Whilst we use Gray code projective patterns, our approach is applicable to other stripe-based time-mutiplexing codes [46]. Explicitly, we want to identify, depending on the orientation of the Gray code stripe, the row or column of pixels in the projector that illuminates each pixel of the camera. We call the pixels of the camera reference sites. One of the novel aspect of our framework, is that it takes into consideration the projective displacement which is induced by the camera, the projector and the scene configuration. This allows the selection of a neighborhood for the smoothing term which is geometrically plausible. The energy function is minimized using a reformulation of the Dynamic Programming (DP) recurrences that supports very high frame rates when implemented on a GPU. Our real-time energy-based formulation improves the quality of the correspondences when working with a low signal-to-noise ratio (SNR). This allows a reduction of the total acquisition time of a system while preserving the quality of the results when the scene contains surfaces with low reflectance. Our method is very useful for projector calibration when using plane-based calibration techniques [125, 100]. The correspondence between the calibration plane and the projector is established on the former. Points forming a regular lattice on the calibration plane are defined as the reference sites. The projector pixel coordinate that illuminates each reference site is identified, and the correspondences are used to compute the homography between the projector and the calibration plane. This naturally scales to multiple cameras and allows a significant improvement of the accuracy of the calibration.

The remainder of this paper is divided as follows: in Section 9.2, previous work is presented; Section 9.3 describes our formulation; the GPU-based DP is described in

Section 9.4; experimental results are presented in Section 9.5.

9.2 Previous work

A recent survey by Salvi *et al.* [91] classified structured light techniques in 3 categories: time-multiplexing, neighborhood codification and direct codification. The Gray code is probably the best known time-multiplexing technique in structured light systems [46]. Some methods use minimization to remove ambiguity in their code [12, 122, 62]. In contrast, we use it to remove ambiguities introduced by the structured light system and environment. In [63] an energy minimization *phase shift* method that adapts the patterns to the scene geometry is proposed. In [108] an off-line ICM-based method is presented. Ours differs by its real-time nature and is also geometrically more plausible*.

A recent survey by Szeliski *et al.* evaluates different minimization algorithms [106]. Energy formulation used for structured light matching may have a large number of labels, making the minimization difficult [108]. A special belief propagation algorithm was proposed to cope with a large number of labels [60]. In stereo, many approches for reducing the search space have been proposed [115]. Nevertheless, these would not allow real-time decoding.

9.3 Formulation of the matching problem

Our algorithm takes as input two sets of images G and R, the first one contains the images G_b^c of camera c viewing the scene while the Gray code pattern representing bit b is projected on the scene. The images R_b^c of the other set are the image G_b^c where white and black stripes of the Gray code are reversed. Depending on the orientation of the Gray code stripes used, the algorithm outputs the position of the row or column of pixels in the projector that illuminated each reference site. The different

* ICM stands for Iterative Conditional Modes.

noise sources present in a projector-camera system make the matching error-prone. For example, the limited depth of field and chromatic aberration of the projector lens affect the detectability of spatial black/white transitions. We use the smoothing term of an energy formulation to reduce the impact of noise.

9.3.1 Energy function

We have a set $\mathcal{P}$ of sites on a reference plane. As seen in section 9.1, it can be the set of camera pixels or points on a calibration plane. For each reference site we want to compute the corresponding row (or column) from the projector image plane. The set $\mathcal{L}$ contains either the rows or columns of pixels in the projector, depending on the orientation of the stripes used. A $\mathcal{L}$-configuration $f : \mathcal{P} \mapsto \mathcal{L}$ associates a row or column of pixels to every site. The energy function is $E(f) =$

$$\sum_{b<B}\sum_{\mathbf{p}\in\mathcal{P}}\Big(\underbrace{e_b(\mathbf{p}, g_b(f(\mathbf{p})))}_{likelihood} + \underbrace{\sum_{\mathbf{q}\in\mathcal{N}_b(\mathbf{p})} s(g_b(f(\mathbf{p})), g_b(f(\mathbf{q})))}_{\text{smoothing}}\Big) \tag{9.1}$$

where B is the number of patterns (bits) of the Gray Code and $\mathcal{N}_b(\mathbf{p})$ is site $\mathbf{p}$'s neighborhood which may vary with b and is described in section 9.3.3. The term $g_b(d)$ is the b-bit of the Gray code representation of the index of row (or column) d. Note that the structure of the energy function of Eq. 9.1 allows the minimization of B functions of 2 labels (for each pattern, the light stripe illuminating a site is either white or black) rather than a large one having 2^B labels as in [108]. The likelihood term is

$$e_b(\mathbf{p}, d) = \begin{cases} \sum_{c<C} e^{\frac{G_b^c(T^c(\mathbf{p})) - R_b^c(T^c(\mathbf{p}))}{\theta}} & \text{if } d = 0 \\ \sum_{c<C} e^{\frac{R_b^c(T^c(\mathbf{p})) - G_b^c(T^c(\mathbf{p}))}{\theta}} & \text{otherwise} \end{cases}$$

where C is the number of cameras and the constant θ is set to 255 in our experiments. When there is only one camera and the reference plane is that of the camera the function T^c is simply the identity function. When the sites in $\mathcal{P}$ correspond to points on the calibration plane, many cameras can be used and T^c is the homography

mapping a site on the calibration plane onto a pixel in camera c. In that case, the scene is composed of a single plane, that is the calibration plane. Our formulation does not allow multiple cameras and an arbitrary scene. The smoothing term is described in the next section and it is similar to the one used in stereo [94].

9.3.2 Projective displacement

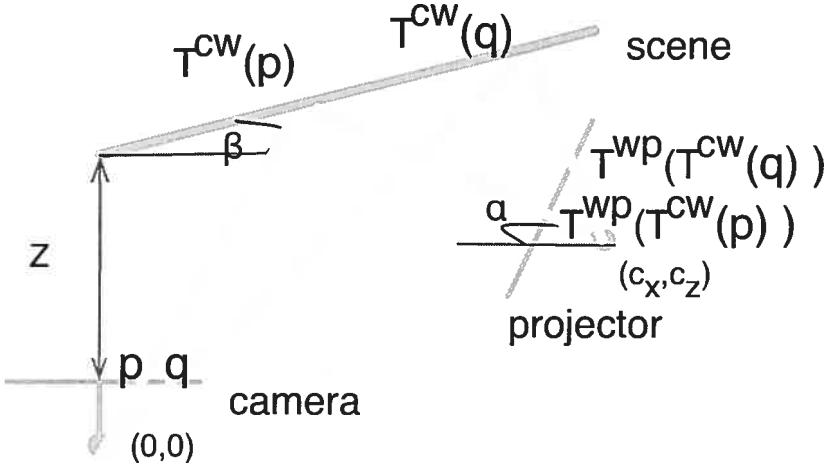

Figure 9.1: Representation of the structured light system. The distance along the optical axis between the camera and the plane is Z. The angle between the camera's image plane and world plane is β and α is the angle between the projector's optical axes and the camera's image plane.

The projective displacement is induced by the scene geometry and the internal and external parameters of the projector-camera system. Figure 9.1 illustrates a 2D projector-camera system. The projective displacement ΔD is simply

$$\Delta D(\mathbf{p}, \mathbf{q}) = T^{cw}(T^{wp}(\mathbf{p})) - T^{cw}(T^{wp}(\mathbf{q})). \tag{9.2}$$

where T^{cw} is the homography between the camera plane and world plane and T^{wp} is the homography between the world plane and the projector plane. When using a calibration plane as reference, the projective displacement is defined as:

$$\Delta D(\mathbf{p}, \mathbf{q}) = T^{wp}(\mathbf{p}) - T^{wp}(\mathbf{q}) \tag{9.3}$$

and is independent of the parameters of the camera. Figure 9.2 shows, for a 2D camera-projector system, the value of the projective displacement ΔD for varying angles α and β when the camera plane holds the reference sites.

The projective displacement will not be included in the smoothing term, but it will be used to select the neighborhood structure used in the energy function of Eq. 9.1. A smoothing penalty between two sites is applied only when the projective displacement is expected to be very close to zero. Further discussion about the neighborhood structure is postponed until Section 9.3.3. This is motivated by the discrete nature of the set of label $\mathcal{L}$ and the continuous nature of the projective displacement that may be difficult to combine into a smoothing term. The smoothing term is defined as

$$s(\lambda, \lambda') = \gamma\delta(\lambda - \lambda') \tag{9.4}$$

where δ is 0 at 0 and 1 elsewhere and γ is a user-defined parameter. Further discussion about γ is postponed until Section 9.5. The total smoothing penalty in Eq. 9.1 between two neighbor sites $\mathbf{p}$ and $\mathbf{q}$, that is $\sum_{b<B} s(g_b(f(\mathbf{p})), g_b(f(\mathbf{q})))$, is thus proportional to the number of bit that are different in the Gray code representation of the index of rows (or columns) $f(\mathbf{p})$ and $f(\mathbf{q})$.

9.3.3 Neighborhood

The neighborhood structure will first be described for the case where the reference sites are in the camera image ($\mathcal{P}$ contains the camera's pixels). For now, we will assume that the camera and projector planes are rectified and that the scene is a single plane which is fronto-parallel to the structured light system. In this case, the

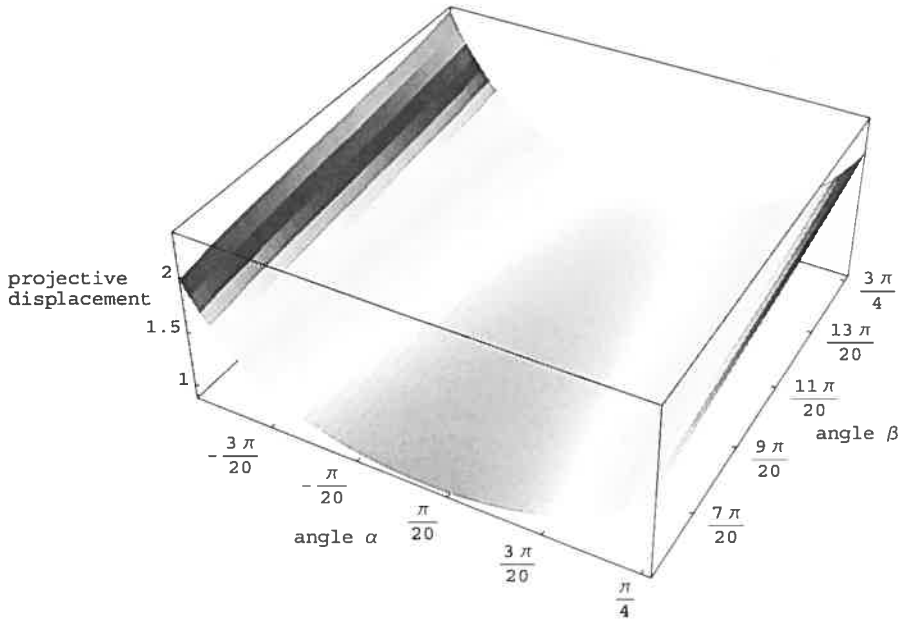

Figure 9.2: Projective displacement in function of angle α and β for a projector and camera having a focal length 5 and a sensor element size of 0.1. The distance along the optical axis between the camera and the plane is 200, The projector is located at (20,0) and the camera is located at the origin (see supplemental material for an animation).

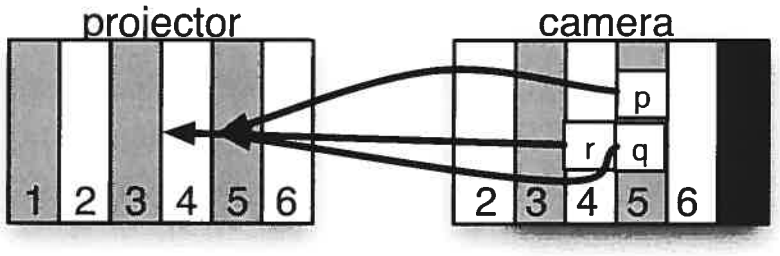

Figure 9.3: Mapping camera pixel to projector column for a rectified pair and a fronto-parallel scene.

image viewed by the camera is that of the projector shifted along the horizontal axe. The magnitude of this displacement - that is, the disparity - depends on the depth of the plane in the scene. Figure 9.3 shows the camera and projector images corresponding to such a scene when the elements of $\mathcal{L}$ are the columns of the projector. The sites $\mathbf{p}$ and $\mathbf{q}$ on the camera image have the same label (5) and $\Delta D(p,q)$ is $(0,0)$, whilst pixels $\mathbf{q}$ and $\mathbf{r}$ have labels that correspond to two adjacent columns (4 and 5) in the projector and $\Delta D(q,r) = (-1,0)$. When a smoothing term that penalizes for a difference of label is computed for $\mathbf{p}$ and $\mathbf{q}$ the presence of a depth discontinuity (and thereby occlusion) is penalized. However, when the same smoothing is applied between $\mathbf{q}$ and $\mathbf{r}$, the smoothing term penalizes the absence of occlusion. Algorithms that use a 4-connected neighborhood with a smoothing term that penalizes for the difference of label, such as [108, 63], promote occlusion in one axe and penalize it on the other. We propose a more coherent approach where the smoothing is applied in a way that never penalizes the absence of occlusion. This is achieved by using a 2-connected neighborhood. For vertical stripes ($\mathcal{L}$ is the set of column of the projector), two pixels are neighbors if they are adjacent and belong to the same column of pixel of the camera. For horizontal stripes, two pixels are neighbors if they are adjacent and belong to the same row of camera pixels. We will show in section 9.5.2 that our neighborhood structure significantly improves the results over 4-connected energy formulations and traditional decoding. Note that the projector and camera views do not need to be rectified; what is needed is that the projector's stripe orientation remains the same in the camera. As will be shown in Section ??, the use of our formulation with a non-rectified configuration provides significant improvement. Also, the 2-connected neighborhood can be used when a calibration plane is the reference as long as the orientation of the stripes remain the same in the calibration plane coordinate system. Note that it is important when the camera plane is used as the reference that the rigid transformation between the camera and the projector does not include a rotation around the optical axis.

Few transitions are present in the structured light patterns associated with the higher order bits of the Gray code since the stripes are several pixels wide. It is therefore safe to assume that the projective displacement is very close to zero, and consequently a 4-connected neighborhood can be used for those bits. Note that solving for those is similar to increasing the size of the projector pixels, thereby making the projective displacement closer to zero. Here, the optimal solution when solving for one (4-connected) pattern could be obtained using Graph Cut [39]. In our experiment we observed that there is little ambiguity in these Bits, resulting in ICM's energy being close to the optimum. In the experimental section of this paper, we use a 2-connected neighborhood for all the bits. Section 9.4 describes a novel real-time GPU-based DP method. If a 4-connected neighborhood is required for the high order bits, GPU implementations of ICM already exist and could be added to our system [50] †.

9.3.4 Sampling of the calibration plane

When a camera image plane is the reference, it is natural to use each pixel of the camera as a site in $\mathcal{P}$. When using our framework for calibration, the calibration plane is the reference and the sites in $\mathcal{P}$ are points forming a regular lattice on the calibration plane. Moreover, those point must lie on the region of the calibration plane illuminated by the projector and a sampling rate λ must be specified. The relation between a site $\mathbf{p} = (x, y)$ in $\mathcal{P}$, that is a point on the calibration plane, and a pixel $X^c = (x^c, y^c)$ belonging to camera c is

$$x = \lambda \frac{\mathbf{h}_1^{wc} \cdot (x^c, y^c, 1)}{\mathbf{h}_3^{wc} \cdot (x^c, y^c, 1)} \quad \text{and} \quad y = \lambda \frac{\mathbf{h}_2^{wc} \cdot (x^c, y^c, 1)}{\mathbf{h}_3^{wc} \cdot (x^c, y^c, 1)}$$

where $\mathbf{h}_i^{wc}$ is the i^{th} row of the matrix representing the homography that maps a point from the camera c into the calibration plane w. When this transformation preserves

† This was not needed in our experiments.

the area, the determinant of the Jacobian

$$\frac{\partial(x,y)}{\partial(x^c,y^c)} = \begin{pmatrix} \frac{\partial x}{\partial x^c} & \frac{\partial y}{\partial x^c} \\ \frac{\partial x}{\partial y^c} & \frac{\partial y}{\partial y^c} \end{pmatrix}$$

is equal to 1. The sampling rate λ is thus obtained by minimizing

$$\sum_{c<C} \sum_{X^c \in G_b^c} (det \frac{\partial(x,y)}{\partial(x^c,y^c)} - \omega)^2 \tag{9.5}$$

where the parameter ω is user-defined and is the desired mean magnification factor between the pixels in the different images G_b^c and the sites in $\mathcal{P}$ (i.e. ω controls the amount of sub-sampling that occurs in the images). The energy function of Eq. 9.5 is linear in λ^2. Note that it would be possible for the user to specify directly λ; however, in our experiments, it was much more intuitive to specify ω. The choice of ω is discussed later in Section 9.5.3.

9.4 GPU-based dynamic programming

Dynamic Programming when a Potts smoothing model is used has a complexity of $\Theta(N.L)$ where N is the number of sites and L is the number of labels [36]. The depth of the recurrence relation is linear in the number of sites. A GPU-based Smith-Waterman DP was presented to exploit the parallelism already present in the standard recursive formulation [69]. We propose to use a divide-and-conquer strategy and to reduce the depth of the recurrence relation in order to increase the amount of parallel computation. Figure 9.4 illustrates the concept on a small problem of 4 sites and 2 labels. On the left, we see two steps of the usual relation. On the right, in the first step, the energies for all the combinations of pairs of sites $(0,1)$ and $(2,3)$ is computed. Four values are thus computed for each pair of sites. In the next step, the lowest energy for the combinations of labels of sites (0,3) are computed. In general, this binary tree structure significantly reduces the number of steps required to process all the sites. The table $t(m_{i,j}, n_{i,j}, k, l)$ is the lowest energy of all maps of sites in the

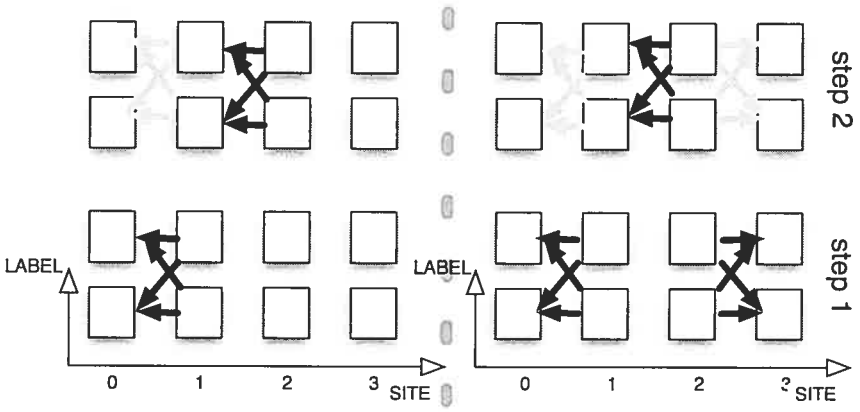

Figure 9.4: Parallelization of the energy computation phase of DP. Left) ordinary recurrence relation. Right) proposed one. The gray arrows represent the smoothing cost already being taken into account, whilst the black ones represent the smoothing cost computed at the current step.

interval $[m_{i,j}, n_{i,j}]$ with site $m_{i,j}$ and $n_{i,j}$ at label k and l respectively. We define $m_{i,j} = i2^j$ and $n_{i,j} = m_{i+1,j} - 1$. Until Section 9.4.1 we will assume that the number of sites is a power of two. Explicitly, the table t is defined inductively as

$$t(m_{i,1}, n_{i,1}, k, l) = \begin{array}{l} e(m_{i,1}, k) + e(n_{i,1}, l) \\ +s(m_{i,1}, n_{i,1}, k, l) \end{array}$$

and for $j > 1$

$$t(m_{i,j}, n_{i,j}, k, l) = \min_{k', l' \in \mathcal{D}} t'(m_{i,j}, n_{i,j}, k, l, k', l')$$

with

$$t'(m_{i,j}, n_{i,j}, k, l, k', l') = \begin{array}{l} t(m_{i,j}, m'_{i,j}, k, k') \\ +t(n'_{i,j}, n_{i,j}, l', l) \\ +s(m'_{i,j}, n'_{i,j}, k', l') \end{array}$$

and where $n'_{i,j} = (2i+1)2^{j-1}$ and $m'_{i,j} = n'_{i,j} - 1$. The minimal energy is $\arg\min_{k,l} t(0, N-1, k, l)$ where $N-1$ is the index of the last site. Note that we could easily add smoothing between the first and last site. This would correspond to reconstructing a label map on a cylinder. The entry of t can be evaluated in $\Theta(N.L^2)$ operations, which is more than for ordinary DP. The depth of the relation is now logarithmic and the minimum can thus be computed in $\Theta(logN)$ rendering passes on a GPU. Note that for our problem the number of labels is two, thereby limiting the increase in computation while taking advantage of the parallelization. Figure 9.5 shows the recovery

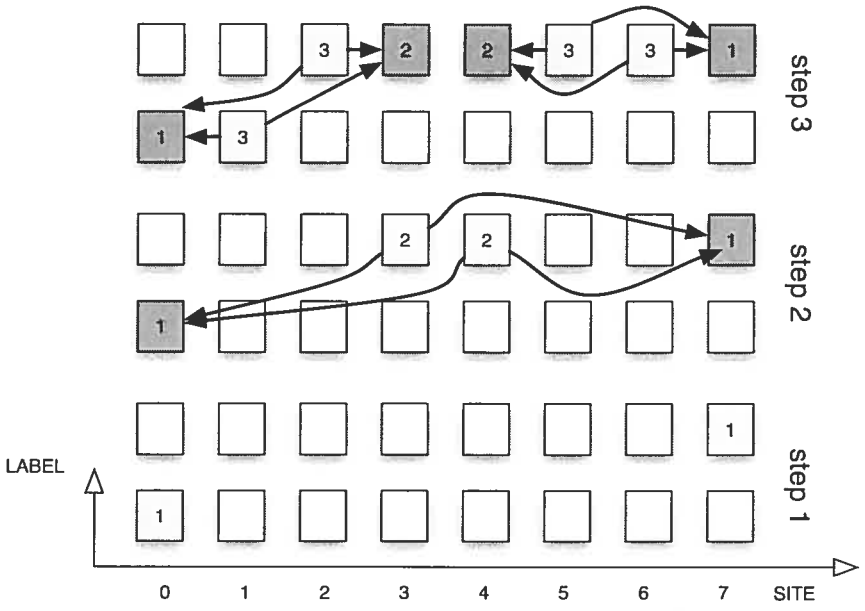

Figure 9.5: Parallel computation of the recovery phase of DP for a problem containing 8 sites and 2 labels. The light gray boxes represent sites for which the label is fixed in the current step. The dark gray boxes represent sites for which the label has been fixed in the previous step. The number on a site is the step at which the label is determined. The arrows indicate the dependency between sites. As an example, the labels of sites 0 and 3 must be known before computation of the label of sites 1 and 2.

process of the label map with the minimum energy for a 2-label problem containing 8 sites. At step 1, the labels of sites 0 and 7 are computed by looking only at the minimum entry of table t for the pair of sites $(0, 7)$. At step 2, the labels of site 3 and 4 are found by using t for the pair $(0, 3)$ and $(4, 7)$ combined with the result of step 1. The process is similar for step 3, except that the results of all previous steps are needed. The label map $f_{i,j}^{k,l}$ is the lowest energy map for all sites in the interval $[m_{i,j}, n_{i,j}]$ having sites $m_{i,j}$ and $n_{i,j}$ at label k and l respectively. Explicitly, the label map $f_{i,j}^{k,l}$ is defined inductively as

$$(f_{i,j}^{k,l}(m_{i,j}), f_{i,j}^{k,l}(n_{i,j})) = (k, l)$$

and for $\{m'_{i'j'}, n'_{i'j'}\} \subseteq]m_{i,j}, n_{i,j}[$

$$(f_{i,j}^{k,l}(m'_{i'j'}), f_{i,j}^{k,l}(n'_{i'j'})) = \arg\min_{k',l' \in \mathcal{D}} t'(m_{i',j'}, n_{i',j'}, f_{i,j}^{k,l}(m_{i',j'}), f_{i,j}^{k,l}(n_{i',j'}), k', l').$$

The recovery phase can be accomplished in $\Theta(\log N)$ rendering passes. Note that similar relations exist to recover the min marginal for each site and the joint min marginal for two neighbor sites. This could be useful for belief propagation or parallel reweighted message passing [117] on the GPU. The combined usage of our GPU-based DP with the iterated dynamic programming of [68] should also be explored.

In our structured light approach, all the rows of pixels in an image can be processed simultaneously. The total complexity of the algorithm is now $\Theta(B \log N)$ rendering passes where N is again the length of a DP line and B is the number of patterns used. Our bitwise GPU-based algorithm scales very well with the increase in resolution of the camera and projector. The decoding of a 10-bit structured light pattern on a 1024×768 camera is accomplished at a rate of more than 12 Gray codes per second on an NVIDIA 6800 GPU. Our GPU-based DP is sufficiently general to be applied to other vision problems and is particularly suitable for applications that use the label map for display.

9.4.1 Implementation remarks

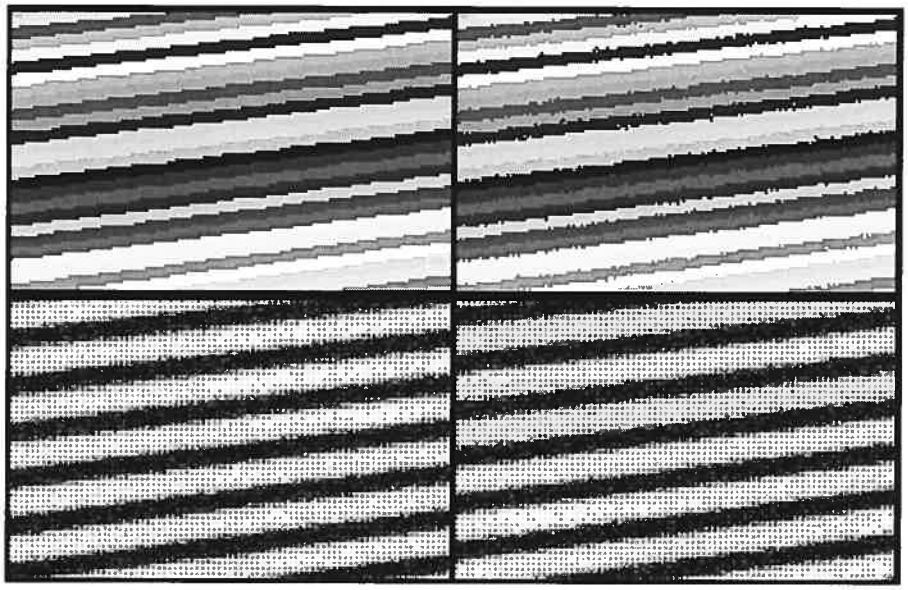

Figure 9.6: Crop of the vertical decoded structured light pattern. Labels are assigned random colors. Top left) results of our algorithm. Top Right) results of standard decoding. Bottom) equalized images of the camera with a Bayer matrix viewing the pattern containing the low order bit. The real intensity varies between 0 and 23.

The recurrences can be computed using different combinations of fragment program and depth test. Their efficency is GPU-dependent and will not be discussed further. Since the energy is not required, we can subtract a constant when computing the entry $t(m_{i,j}, n_{i,j}, k, l)$. The constant for each pair $(m_{i,j}, n_{i,j})$ is $\min_{k,l} t(m_{i,j}, n_{i,j}, k, l)$. This normalization allows the use of 16-bit arithmetics, thereby reducing the amount of storage and speeding up computation. It is also possible to reduce the number of buffers. As an example, when minimizing a problem having 3 labels, 9 entries in table t are required for each pair $(m_{i,j}, n_{i,j})$. Normalizing such that $t(m_{i,j}, n_{i,j}, 0, 0) = 0$ allows the use of two RGBA buffers. The number of sites on a DP line can be expanded to a power of two, by setting the likelihood term of dummy sites to a constant value.

This can be efficiently done using the texture's clamping mode. The result of each minimization can be kept in the GPU until all bits have been processed and then converted in binary before being sent back to the CPU.

9.5 Experimental evaluation

In this section, we will first evaluate the behavior of our method when there is departure from the geometric model. Then, the impact of neglecting the projective displacement will be investigated. Finally, we will use our method for projector calibration.

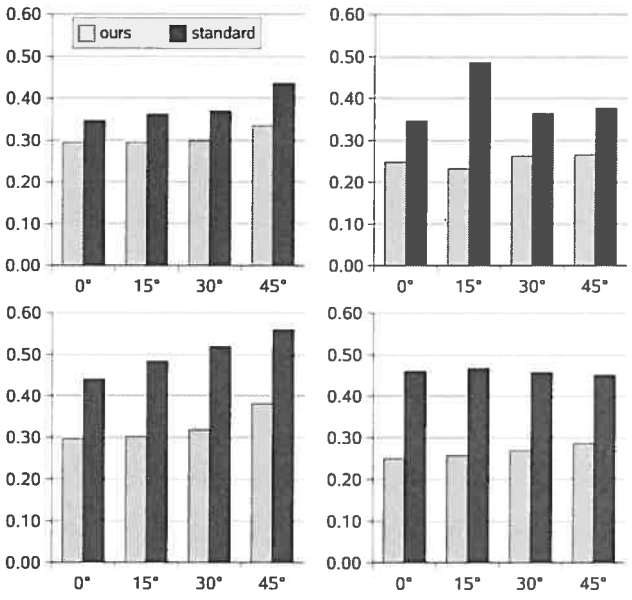

Figure 9.7: Least squared error (in pixels) of line fitting. The smoothing parameter is fixed for the 4 angles. Top left) smallest aperture horizontal code. Top right) smallest aperture vertical code. Bottom left) largest aperture horizontal code. Bottom right) largest aperture vertical code.

approximate plane angle	Departure from the model					
	Vertical stripes			Horizontal stripes		
	min	mean	max	min	mean	max
0	0.0	0.5	1.5	0.0	1.4	2.1
15	0.0	1.9	4.2	0.0	1.4	1.8
30	0.1	4.0	7.8	0.0	1.3	1.7
45	1.7	6.8	11.9	0.0	1.0	1.6

Table 9.1: Angle in degree between the measured reference lines and the expected ones according to the neighborhood structured.

9.5.1 Departure from the model

A planar surface was imaged at an approximate angle of 0, 15, 30 and 45 degrees between the projector and plane. The camera (with a Bayer matrix) and projector were installed in a convergent configuration. The experiment was repeated with two different apertures, resulting in a change in the SNR. One of the recovered mappings is shown in Figure 9.6. Note how small artifacts are cleaned up even if the SNR ratio is low. Since linearity is preserved under projective transform, for each label discontinuity in the camera a linear regression was performed. The residual error of the fit was used as a metric to compare our approach to standard decoding. Note that the camera and projector had very little radial distortion. Figure 9.7 shows the results. With standard decoding, the error rate significantly increases with the reduction of the aperture: this phenomenon did not occur with ours. As the angle increases, the amount of chromatic aberration and regions that are out of focus also increases. Our approach still provides significant improvement. As the angle increases the model breaks down for the vertical stripes, however the error does not increase more than for the horizontal one (see Fig 9.7). The departure from the model is shown in Table 9.1. Figure 9.9 shows a stability analysis of the smoothing parameter γ for our algorithm, giving the error over a broad range of values. Figure 9.8 shows the recovered mappings for a scene composed of complex curved surfaces. Our algorithm

also provides improvement for this non-planar surface.

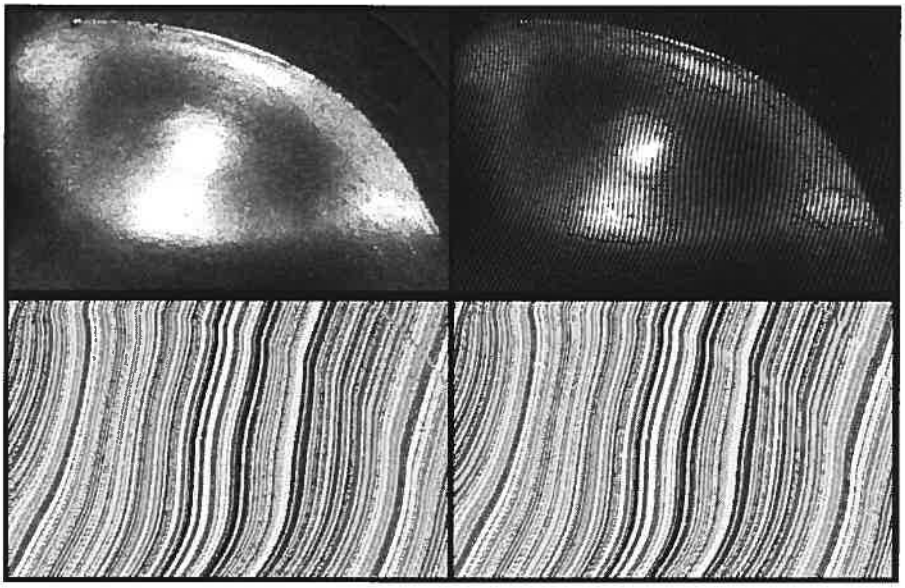

Figure 9.8: Decoded structured light pattern for horizontal stripes. Top left) color image Top right) raw image of the stripes Bottom left) standard decoding and Bottom right) our algorithm (see the small details in the high resolution images in the electronic version of the paper).

9.5.2 Discontinuity preservation test

A scene composed of two planes separated by a strong depth discontinuity was imaged by a projector-camera system. The large disparity discontinuities present in the image were manually segmented to create a *discontinuity* ground truth. The scene and acquisition condition were controlled, allowing a non-ambiguous segmentation. A discontinuity location is considered erroneous if it differs from the ground truth. The results are presented in Table 9.2, and images in Figure 9.10. Here, direct Gray code decoding already produces low error rates because of the ideal acquisition conditions. Nevertheless, the improvement after applying our algorithm is significant. In order to

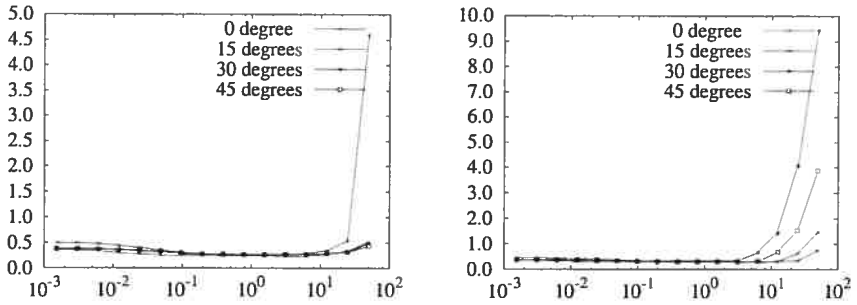

Figure 9.9: Robustness to change of the smoothing parameter Left) horizontal stripes and Right) vertical stripes. The smoothing parameter is increased by a factor of more than 32000.

show the impact of neglecting the projective displacement, we compare our method with that of Tardif et al. [108] which used a 4-connected neighborhood. Their best results are achieved with a smoothing of 0 and are identical to the ones of standard decoding. A maximum flow algorithm that yields the global minimum was used instead of ICM thereby showing that the error comes from the model [89]. We show results in Table 9.2 for a non-zero smoothing. In the first and fourth columns the high error rate comes from the smoothing term that favored occlusion by penalizing the difference of label between neighbor sites having a projective displacement that is not close to zero. The error rate is much lower in the two other columns since the projective displacement should have been close to zero. We also evaluate our minimization algorithm using a 4-connected neighborhood for all bits. Again, the best results are achieved when the smoothing is zero. Note that a correct discontinuity location does not imply that the correspondences on both sides are correct as well.

9.5.3 Projector calibration

A calibration plane was imaged by 4 cameras while vertical and horizontal Gray codes are projected on it (see Fig 9.11). The homographies between the checkerboard and

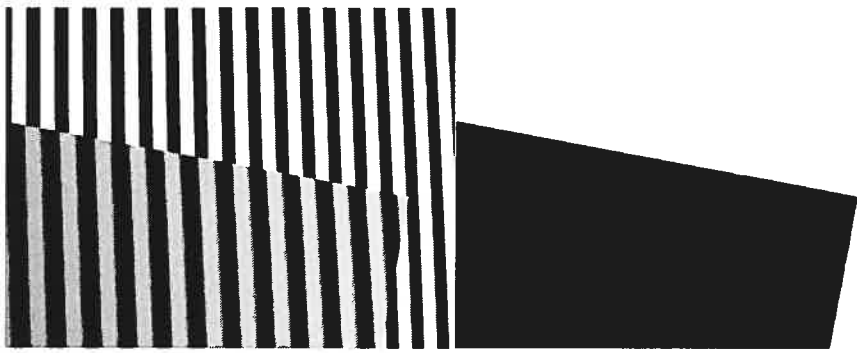

Figure 9.10: Image of the discontinuity scene with ground truth.

algorithm	error in border localization			
	Horizontal stripes		Vertical stripes	
	X and X+1	Y and Y+1	X and X+1	Y and Y+1
STD	0.045	0.052	0.006	0.003
OURS	0.027	0.043	0.003	0.003
TARDIF	6.148	0.348	0.030	3.592

Table 9.2: Percentages of error for the discontinuity scene.

camera images are automatically established using the ARTag target system [25]. The projector was fixed on two translation tables allowing 2 DOF. The displacement was independently measured by a laser tracker accurate to 25 um. In our experiment we calibrated three projectors of different models. The system setup is illustrated in Fig 9.11.

Our minimization framework was used with the calibration plane as the reference. Note that in our experiments, the lattice is aligned with the checkerboard and the distance between adjacent points of the lattice is usually less than 200 micron. The homography between the projector and the calibration plane can be established by projecting horizontal and vertical Gray code stripes and using the transition of label between points of the lattice as feature points. When many homographies are available, a planar calibration algorithm can be used [125, 100]. The homographies are obtained by changing the orientation of the plane. Moreover, 3 others were obtained

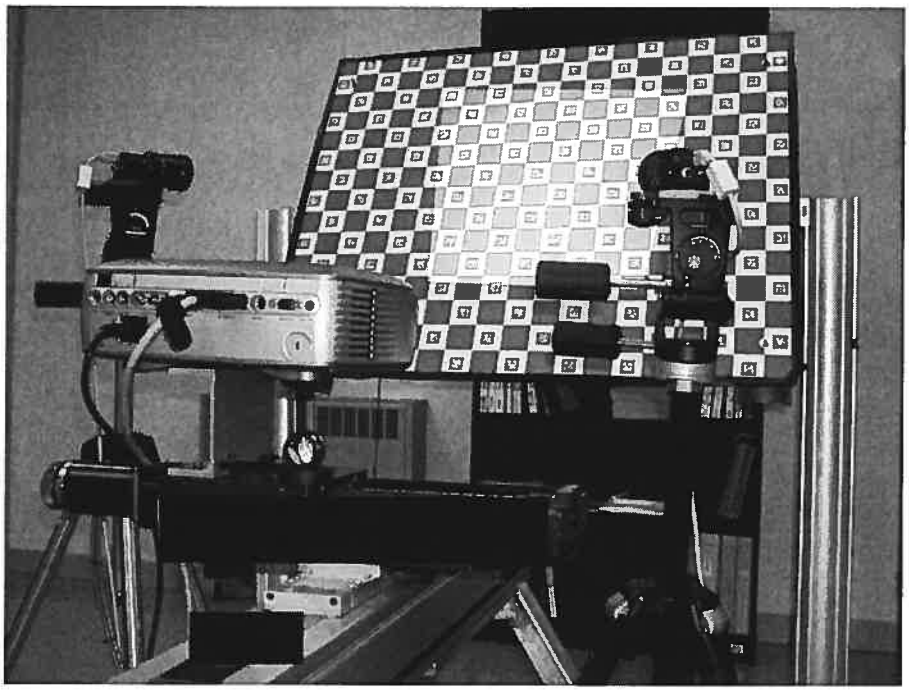

Figure 9.11: Setup used for projector calibration.

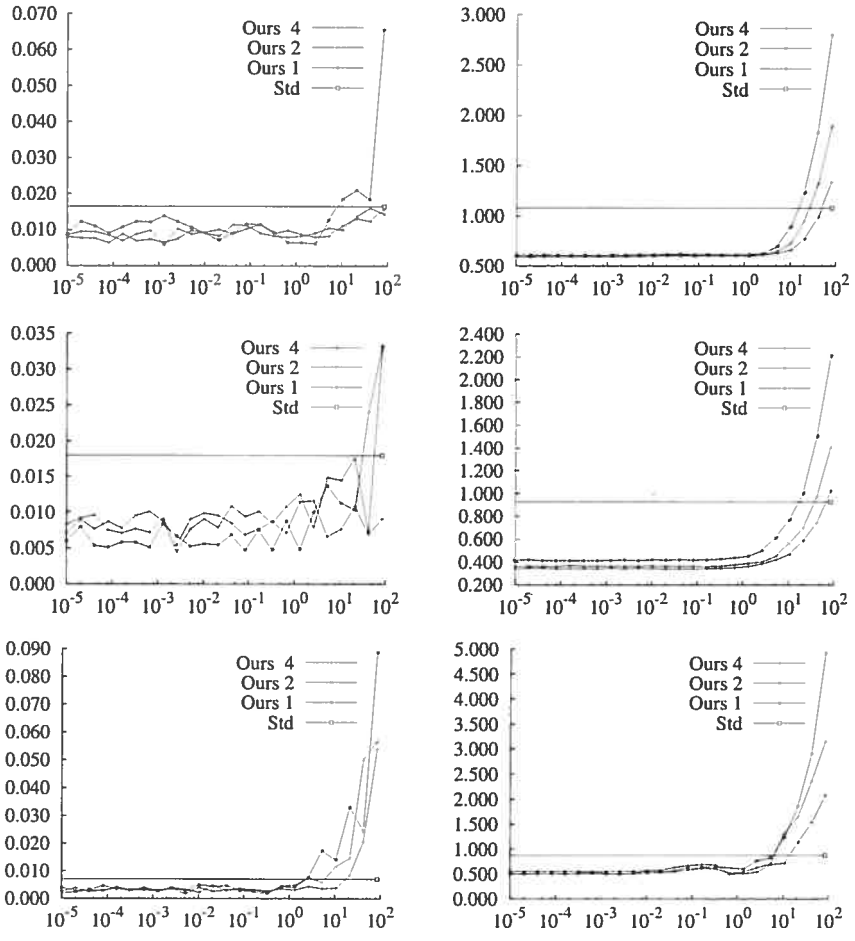

Figure 9.12: From top to bottom: projector A,B and C. Left) error ratio difference between the computed and measured displacement divided by the measured displacement. Right) reprojective error. Results are shown for 3 values of ω.

by moving the projector in a triangular path in front of the fixed calibration plane. The displacement for the projectors A, B and C is respectively 339.94, 336.28 and 344.49mm, at a projective distance of approximately 1.2 m. The relative displacement of the projector was measured by the laser tracker and could be compared to the one obtained from the homographies [125]. Note that the information obtained by the laser tracker was never used in the calibration process, but only as a validation of the results: the calibration technique relies only on off-the-shelf cameras and projectors and a low-cost target.

Figure 9.12 shows, for three values of ω, a stability analysis for the smoothing parameter γ for our algorithm, giving the error (in displacement ratio and reprojective) over a broad range of values. We compare with the standard method of finding the fiducial points of the plane in the camera's image, decoding the Gray code at those points and using then to compute the homography. In this case, the plane-projector homography is thus computed directly through the plane-camera-projector chain [81, 10, 123], while we use the camera-plane-projector chain. Our approach yields a much lower error rate even when no smoothing is used. For the projector C we could achieve an error ratio of $\frac{17}{10000}$ which is more than 4 times less than the other method.

Our low cost calibration target is specular and is also not perfectly planar: the form factor and RMS error, as measured by the laser tracker, were respectively as high as .244mm and 1.135mm. In our tests, the positioning of the camera was chosen to minimize the impact of these. Modeling the radial distortion for the projector and cameras of our system should further improve the results [40].

9.6 Conclusion

We presented a novel energy-based formulation of the matching problem encountered when establishing the correspondence using a structured light system. The corre-

spondence between projector and camera pixels can be established in the camera plane or on a calibration plane. A real-time GPU algorithm is computationally inexpensive and scales well with an increase in camera and projector resolution. Our framework allows significant quality improvement over standard Gray code decoding when working with cameras with Bayer matrix and in low SNR condition.

As for future work, the extension of our energy-based framework to multiple projectors and cameras should be explored. We should investigate the application of our GPU-based dynamic programming to other vision problems.

Chapitre 10

DISCUSSION ET CONCLUSION

Les travaux que nous avons présentés dans cette thèse se rapportent à la mise en correspondance passive et active. Nous avons utilisé des systèmes calibrés et non calibrés. Nous avons proposé trois nouveaux algorithmes passifs fonctionnant avec des systèmes calibrés. Un nouvel algorithme permettant la mise en correspondance active a également été proposé. Ce dernier a été conçu spécifiquement pour être utilisé avec les systèmes non calibrés. Il peut aussi servir avec un système calibré. Il est particulièrement efficace pour établir la mise en correspondance nécessaire au calibrage.

Les prochaines sections décriront brièvement les méthodes que nous avons proposées. Nous élaborerons les applications et les extensions possibles à chacun des algorithmes proposés.

10.1 Mise en correspondance passive

Nous avons proposé trois algorithmes permettant d'effectuer la mise en correspondance passive. Ceux-ci sont conçus spécifiquement pour gérer les occultations en stéréoscopie à caméras multiples de la deuxième famille. Dans ce type de configuration, toutes les caméras possèdent la même orientation et elles sont du même côté de la scène. Les données utilisées sont généralement obtenues à l'aide d'une caméra montée sur une table de déplacement ou sur un bras robotisé. La compagnie Point Grey Research dispose de deux systèmes stéréoscopiques à trois caméras: le Digiclops® et le Bumblebee® XB3. Le premier possède une configuration en "L". Le second système est un prototype qui devrait posséder trois caméras installées de manière

colinéaire. Les algorithmes développés devraient fonctionner avec les images produites par ces systèmes stéréoscopiques. Les deux premiers algorithmes (chap. 5 et 6) s'attaquent à la reconstruction stéréoscopique et le troisième (chap. 7) s'intéresse au post-traitement. Ce dernier nécessite une carte de disparités initiale. Le problème d'estimation de la disparité est remplacé par un problème d'estimation de la localisation des bordures. Tous les algorithmes utilisent un relâchement de la contrainte de visibilité qui se nomme contrainte de *géo-cohérence*. Une solution respecte cette contrainte lorsqu'aucune caméra non visible n'est utilisée. Il est cependant possible de ne pas utiliser une caméra lorsque celle-ci est visible. Les algorithmes présentés aux chapitres 6 et 7 relâchent cette contrainte en l'appliquant uniquement à un sous-ensemble de caméras. Lorsqu'aucune des caméras de ce sous-ensemble ne sont pas visibles, une heuristique est utilisée afin de déterminer quelles caméras seront considérées *visibles* et serviront dans le calcul du terme de vraisemblance. Dans tous nos algorithmes, l'information de visibilité pour chacun des pixels a été représentée sous forme de masque de visibilité. L'algorithme, proposé au chapitre 5, impose la contrainte d'ordre dans la solution finale. Il peut néanmoins, détecter des régions qui sont suceptibles de contenir un bris de la contrainte d'ordre. Il est alors possible d'utiliser l'algorithme présenté au chapitre 6 afin d'obtenir une nouvelle solution dans ces régions. Le temps de calcul des différentes méthodes varie de plusieurs ordres de magnitude. La méthode proposée au chapitre 5 est très robuste quant au problème de spécularité et est aussi beaucoup plus lente que les autres. La méthode présentée au chapitre 6 pourrait probablement fonctionner en temps réel en utilisant un FPGA. Elle est malheureusement plus susceptible de commettre des erreurs dans les zones peu texturées et spéculaires. L'algorithme présenté au chapitre 7 permet d'améliorer significativement les cartes de disparités produites par une vaste gamme d'algorithmes. Il s'implante bien sur des systèmes multi-processeurs.

Les algorithmes de mise en correspondance que nous avons proposés permettent d'obtenir des modèles $2D\frac{1}{2}$. Il existe plusieurs algorithmes permettant de trouver des

modèles 3D complets [97]. Ceux-ci sont des algorithmes de stéréoscopie multi-vues de classe un. Il serait particulièrement intéressant de vérifier si la méthode présentée au chapitre 5 permet d'améliorer la qualité des modèles 3D obtenus par ces algorithmes. La méthode par coupe de bordures présentée au chapitre 7 pourrait être adaptée pour améliorer la localisation des surfaces d'un modèle 3D complet. Lorsque l'on fait du lissage en reconstruction 3D, le lissage est généralement appliqué sur les trois dimensions. Notre méthode applique le lissage sur uniquement deux dimensions. Il serait probablement possible d'utiliser la stratégie de propagation du lissage utilisée au chapitre 6 afin d'adapter notre algorithme à la reconstruction 3D complète.

10.2 Mise en correspondance active

Une méthode basée sur la minimisation d'énergie et qui fonctionne en temps réel sur un processeur graphique programmable a été présentée au chapitre 9. Le terme de lissage utilisé dans notre formulation contient un modèle de correction des disparités (*disparity correction model*). Celui-ci permet au terme de lissage de demeurer géométriquement plausible. Nous avons utilisé le code de Gray, mais il serait possible d'utiliser d'autres codes. Le décodage se fait bit par bit et la méthode se caractérise par sa capacité d'effectuer le décodage du code de Gray sur n'importe lequel des plans du monde. Ceci est possible lorsque l'homographie entre la caméra et le plan est connue. La méthode peut être étendue aux configurations multi-caméras et est utile lors du calibrage d'un projecteur. La méthode utilisée est très efficace pour réduire les artefacts introduits par la matrice de Bayer. Nous avons montré que les correspondances obtenues par notre méthode, entre un plan de calibrage et un projecteur, sont fiables. Pour le vérifier, nous avons calibré plusieurs projecteurs. Lors de certaines de nos expériences, nous avons fait varier la position des projecteurs en gardant le plan de calibrage fixe. Cette façon de faire nous a permis de comparer la variation de la position des centres de projection avec celle mesurée indépendamment à l'aide

d'équipement de métrologie 3D. Nous avons obtenu un ratio d'erreur, des distances entre les centres de projection, pouvant atteindre $\frac{17}{10000}$. Ces expériences ont été réalisées avec des projecteurs multimédias standards disponibles dans les grandes chaînes d'équipements électroniques. Un plan de calibration bon marché a été utilisé.

Nous avons travaillé uniquement avec des systèmes de projecteurs non calibrés. Si le calibrage était connu, le modèle permettrait de corriger les variations de disparités (*disparity correction model*). Malheureusement, lorsque celui-ci est différent de zéro, il devient impossible de minimiser les bits un à la fois. Lorsque le calibrage des caméras et du projecteur est connu, il est possible de rectifier les images. Les systèmes projecteurs et caméras calibrés servent généralement à la reconstruction tridimensionnelle. Dans ce cas, il serait alors possible d'utiliser une seule orientation des bandes et le nombre d'étiquettes du problème de minimisation diminuerait de manière significative. Il serait également possible de fixer une disparité maximale et minimale. L'intervalle de recherche dépendrait du volume de reconstruction ainsi que de la configuration du système caméras et projecteurs. Les stratégies utilisées pour effectuer la mises en correspondance passive pourraient alors être adaptées à la mise en correspondance active. Il serait particulièrement intéressant d'adapter les méthodes de gestion des occultations développées dans la première partie de cette thèse.

L'algorithme de programmation dynamique sur GPU proposé au chapitre 9 a été utilisé pour résoudre un problème de décodage de la lumière structurée. Notre méthode est bien adaptée aux applications utilisant les cartes d'étiquettes pour l'affichage graphique qui utilisent de 2 à 3 étiquettes. Notre formulation est particulièrement bien adaptée aux problèmes possédant peu d'étiquettes. L'extension de notre méthode à la proprammation dynamique itérative et aux méthodes de passage de messages devrait être envisagée.

RÉFÉRENCES

[1] L. Alvarez, R. Deriche, J. Sanchez, and J. Weickert. Dense disparity map estimation respecting image discontinuities: a PDE and scalespace based approach. Technical Report RR-3874, INRIA, 2000.

[2] P. N. Belhumeur. A Bayesian approach to binocular stereopsis. *International Journal of Computer Vision*, 19(3):237–260, 1996.

[3] M. Berthod, L. Gabet, G. Giraudon, and J.L. Lotti. High-resolution stereo for the detection of buildings. In *Automatic Extraction of Man-Made Objects from Aerial and Space Images*, pages 135–144, 1995.

[4] S. Birchfield and C. Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):401–406, 1998.

[5] M. Bleyer and M. Gelautz. A layered stereo algorithm using image segmentation and global visibility constraints. In *IEEE International Conference on Image Processing*, pages 2997–3000, 2004.

[6] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.

[7] Y. Boykov, O. Veksler, and R. Zabih. Markov random fields with efficient approximations. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 648–654, 1998.

[8] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cut. In *International Conference on Computer Vision*, pages 377–384, 1999.

[9] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[10] M. Brown, A. Majumder, and R. Yang. Camera-based calibration techniques for seamless multiprojector displays. *IEEE Transactions on Visualization and Computer Graphics*, 11(2):193–206, 2005.

[11] A. Brunton, C. Shu, and G. Roth. Belief propagation on the GPU for stereo vision. In *Canadian Conference on Computer and Robot Vision*, pages 76–82, 2006.

[12] C. Chen, Y. Hung, C. Chiang, and J. Wu. Range acquisition using color structured lighting and stereo vision. *Image Vision Comput.*, 15(6):445–456, 1997.

[13] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. McGraw-Hill, New York, 1990.

[14] I. J. Cox, S. Hingorani, B. M. Maggs, and S. B. Rao. A maximum likelihood stereo algorithm. *Computer Vision and Image Understanding*, 63(3):542–567, 1996.

[15] M.-A. Drouin, M. Trudeau, and S. Roy. Fast multiple-baseline stereo with occlusion. In *3-D Digital Imaging and Modeling*, pages 540–548, 2005.

[16] M.-A. Drouin, M. Trudeau, and S. Roy. Geo-consistency for wide multi-camera stereo. Technical Report 1273, DIRO, Université de Montréal, Montréal, Canada, 2005.

[17] M.-A. Drouin, M. Trudeau, and S. Roy. Geo-consistency for wide multi-camera stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 351–359, 2005.

[18] M.-A. Drouin, M. Trudeau, and S. Roy. Improving border localization of multi-baseline stereo using border-cut. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 511–518, 2006.

[19] M.-A. Drouin, M. Trudeau, and S. Roy. Non-uniform hierarchical geo-consistency for multi-baseline stereo. In *Canadian Conference on Computer and Robot Vision*, 2007.

[20] G. Egnal and R. P. Wildes. Detecting binocular half-occlusions: Empirical comparisons of five approaches. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8):1127–1133, 2002.

[21] L. Falkenhagen. Hierarchical block-based disparity estimation considering neighbourhood constraints. In *International Workshop on Synthetic-Natural Hybrid Coding and Three-Dimensional Imaging*, 1997.

[22] O. Faugeras and Q.-T. Luong. *The geometry of multiple images.* MIT Press, Cambridge, 2001.

[23] O. D. Faugeras and R. Keriven. Complete dense stereovision using level set methods. In *European Conference on Computer Vision*, pages 379–393, 1998.

[24] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient belief propagation for early vision. *International Journal of Computer Vision*, 70(1):41–54, 2006.

[25] M. Fiola. ARTag revision 1. a fiducial marker system using digital techniques. Technical Report NRC 47419, National Research Council of Canada, 2004.

[26] M. Fiola. Artag a fiducial marker system using digital techniques. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 590–596, 2005.

[27] M. Fiola and C. Shu. Fully automatic camera calibration using self-identifying calibration targets. Technical Report NRC 48306, National Research Council of Canada, 2005.

[28] L. Ford and D. Fulkerson. *Flows in Networks.* Princeton University Press, 1962.

[29] M. Fradkin, M. Roux, H. Maitre, and U. M. Leloglu. Surface reconstruction from multiple aerial images in dense urban areas. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 262–267, 1999.

[30] D. Freedman and P. Drineas. Energy minimization via graph cuts: Settling what is possible. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 939–946, 2005.

[31] A. Fusiello, V. Roberto, and E. Trucco. Efficient stereo with multiple windowing. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 858–863, 1997.

[32] G. Gimel'farb and U. Lipowezky. Accuracy of the regularized dynamic programming stereo. In *International Conference on Pattern Recognition*, 2001.

[33] M. Goesele, S. M. Seitz, and B. Curless. Multi-view stereo revisited. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2402–2409, 2006.

[34] A. V. Goldberg. Recent developments in maximum flow algorithms. In *Proceedings of the 6th Skandinavian Workshop on Algorithm Theory*, pages 1–10, 1998.

[35] A. V. Goldberg and S. B. Rao. Length functions for flow computations. Technical Report 97-055, NEC Research Institute, Princeton NJ, 1997.

[36] M. Gong and Y.-H. Yang. Fast stereo matching using reliability-based dynamic programming and consistency constraint. In *International Conference on Computer Vision*, page 610, 2003.

[37] S. Graf and T. Hanning. Analytically solving radial distortion parameters. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1104–1109, 2005.

[38] F. Gray. Pulse code communication, u. s. patent 2 632 058, 17 mars, 1953.

[39] D. M. Greig, B. T. Porteous, and A. H. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society*, 51(2):271–279, 1989.

[40] R. Hartley and S. B. Kang. Parameter-free radial distortion correction with center of distortion estimation. In *International Conference on Computer Vision*, pages 1834–1841, 2005.

[41] R. Hartley and S. B. Kang. Parameter-free radial distrotion correction with center estimation. Technical Report MSR-TR-2005-42, Microsoft Research, 2005.

[42] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, second edition, 2004.

[43] C. Huang and A. Darwiche. Inference in belief networks: A procedural guide. *International Journal of Approximate Reasoning*, 15(3):225–263, 1996.

[44] Y.P. Hung, C.S. Chen, K.C. Hung, Y.S. Chen, and C.S. Fuh. Multipass hierarchical stereo matching for generation of digital terrain models from aerial images. *Machine Vision and Applications*, 10(5-6):280–291, 1998.

[45] S Inokuchi, K Sato, and Matsuda. Range imaging system for 3-d object recognition. In *International Conference on Pattern Recognition*, pages 806–808, 1984.

[46] S. Inokuchi, K. Sato, and F. Matsuda. Range imaging system for 3-d object recognition. In *International Conference on Pattern Recognition*, pages 806–808, 1984.

[47] S. Intille and A F. Bobick. Disparity-space images and large occlusion stereo. In *European Conference on Computer Vision*, pages 179–186, 2002.

[48] H. Ishikawa. Exact optimization for markov random fields with convex priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1333–1336, 2003.

[49] H. Ishikawa and D. Geiger. Occlusions, discontinuities, and epipolar lines in stereo. In *European Conference on Computer Vision*, pages 232–248, 1998.

[50] P-M Jodoin, M Mignotte, and J-P St-Amour. Markovian energy-based computer vision algorithms on graphics hardware. In *International Conference on Image Analysis and Processing*, pages 592–603, 2005.

[51] O. Juan and Y. Boykov. Active graph cuts. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1023–1029, 2006.

[52] T. Kanade and M. Okutomi. A stereo matching algorithm with an adaptive window: Theory and experiment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9):920–932, 1994.

[53] S. Kang, R. Szeliski, and J. Chai. Handling occlusions in dense multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 103–110, 2001.

[54] P. Kohli and P. H. Torr. Efficiently solving dynamic markov random fields using graph cuts. In *International Conference on Computer Vision*, pages 922–929, October 2005.

[55] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. Technical Report MSR-TR-2005-38, Microsoft Research, 2005.

[56] V. Kolmogorov and M. Wainwright. On the optimality of tree-reweighted max-product message-passing. In *Annual Conference on Uncertainty in Artificial Intelligence*, pages 316–302, 2005.

[57] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions via graph cuts. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 508–515, 2001.

[58] V. Kolmogorov and R. Zabih. What energy function can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159, 2001.

[59] V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph cuts. In *European Conference on Computer Vision*, pages 82–96, 2002.

[60] N. Komodakis. Image completion using global optimization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 442–452, 2006.

[61] N. Komodakis and G. Tziritas. A new framework for approximate labeling via graph cuts. In *International Conference on Computer Vision*, pages 1018–1025, 2005.

[62] T. P. Koninckx, I. Geys, T. Jaeggli, and L. Van Gool. A graph cut based adaptive structured light approach for real-time range acquisition. In *3D Data Processing, Visualization and Transmission*, pages 413–421, 2004.

[63] T. P. Koninckx, P. Peers, P. Dutre, and L. Van Gool. Scene-adapted structured light. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 611–618, 2005.

[64] A. Koschan and V. Rodehorst. Dense depth maps by active color illumination and image pyramids. In *Advances in Computer Vision*, pages 137–148, 1997.

[65] J.D. Krol and W.A. van der Grind. The double-nail illusion. *Perception*, 11(5):615–619, 1982.

[66] K.N. Kutulakos and S.M. Seitz. A theory of shape by space carving. *International Journal of Computer Vision*, 38(3):133–144, 2000.

[67] C. Lei, J. Selzer, and Y.-H. Yang. Region-tree based stereo using dynamic programming optimization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2378–2385, 2006.

[68] C. Leung, B. Appleton, and C. Sun. Fast stereo matching by iterated dynamic programming and quadtree subregioning. In *British Machine Vision Conference*, pages 97–106, 2004.

[69] Y. Liu, W. Huang, J. Johnson, and S. Vaidya. GPU accelerated Smith-Waterman. In *International Conference on Computational Science*, pages 188–195, 2006.

[70] J. Lotti and G. Giraudon. Adaptive window algorithm for aerial image stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 701–703, 1994.

[71] J. Lotti and G. Giraudon. Correlation algorithm with adaptive window for aerial image in stereo vision. In *Image and Signal Processing for Remote Sensing*, pages 76–87, 1994.

[72] M. I. A. Lourakis and A. A. Argyros. Is levenberg-marquardt the most efficient optimization algorithm for implementing bundle adjustment? In *International Conference on Computer Vision*, pages 1526–1531, 2005.

[73] M.I.A. Lourakis and A.A. Argyros. The design and implementation of a generic sparse bundle adjustment software package based on the levenberg-marquardt algorithm. Technical Report 340, Institute of Computer Science - FORTH, Heraklion, Crete, Greece, 2004.

[74] Y. Ma, S. Soatto, J. Kosecka, and S. Shankar Sastry. *An Invitation to 3-D Vision.* Springer, 2004.

[75] J. Magarey and A. Dick. Multiresolution stereo image matching using complex wavelets. In *International Conference on Pattern Recognition*, pages 4–7, 1998.

[76] T. Meltzer, C. Yanover, and Y. Weiss. Globally optimal solutions for energy minimization in stereo vision using reweighted belief propagation. In *International Conference on Computer Vision*, pages 428–435, 2005.

[77] E. M. Mikhail, J. S. Bethel, and J. C. McGlone. *Introduction to modern photogrametry.* Wiley, 2004.

[78] K. Murphy, Y. Weiss, and M. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Annual Conference on Uncertainty in Artificial Intelligence*, pages 467–47, 1999.

[79] Y. Nakamura, T. Matsuura, K. Satoh, and Y. Ohta. Occlusion detectable stereo - occlusion patterns in camera matrix-. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 371–379, 1996.

[80] Y. Ohta and T. Kanade. Stereo by intra- and inter-scanline using dynamic programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(2):139–154, 1985.

[81] T. Okatani and K. Deguchi. Autocalibration of a projector-camera system. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1845–1855, 2005.

[82] M. Okutomi and Takeo Kanade. A multiple-baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(4):353–363, 1993.

[83] S. J. Osher and R. P. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces.* Springer-Verlag, Berlin, 2002.

[84] H. Pan and J. Magarey. Multiresolution phase-based bidirectional stereo matching with provision for discontinuity and occlusion. *Digital Signal Processing*, 8(12):255–266, 1998.

[85] S. Paris, F. X. Sillion, and L. Quan. A surface reconstruction method using global graph cut optimization. *International Journal of Computer Vision*, 66(2):141–161, 2006.

[86] J. Park and S. Inoue. Hierarchical depth mapping from multiple cameras. In *International Conference on Image Analysis and Processing*, pages 685–692, 1997.

[87] J. Park and S. Inoue. Acquisition of sharp depth map from multiple cameras. *Signal Processing: Image Communication*, 14(1-2):7–19, 1998.

[88] L. H. Quam. Hierarchical warp stereo. In *In Image Understanding Workshop*, pages 149–155, 1984.

[89] S. Roy. Stereo without epipolar lines : A maximum-flow formulation. *International Journal of Computer Vision*, 34(2-3):147–162, 1999.

[90] S. Roy and M-A Drouin. Non-uniform hierarchical pyramid stereo for large images. In *Vision, Modeling and Visualization*, pages 403–410, 2002.

[91] J. Salvi, J. Pages, and J. Batlle. Pattern codification strategies in structured light systems. *Pattern recognition letters*, 37(4):827–849, 2004.

[92] M. Sanfourche, G. Le Besnerais, and F. Champagant. On the choice of the correlation term for multi-baseline stereo-vision. In *British Machine Vision Conference*, 2004.

[93] D. Scharstein and R. Szeliski. Middlebury stereo vision page, www.middlebury.edu/stereo.

[94] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(7):7–42, 2002.

[95] D. Scharstein and R. Szeliski. High-accuracy stereo depth maps using structured light. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 195–203, 2003.

[96] H. Schultz. Terrain reconstruction from oblique views. In *ARPA Image Understanding Workshop*, pages 1001–1008, 1994.

[97] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 519–528, 2006.

[98] S. M. Seitz and C. R. Dyer. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision*, 35(2):151–173, 1999.

[99] C. Strecha, R. Fransens, and L. Van Gool. Combined depth and outlier estimation in multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2394–2401, 2006.

[100] P. F. Sturm and S. J. Maybank. On plane-based camera calibration: A general algorithm, singularities, applications. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 432–437, 1999.

[101] C. Sun. Multi-resolution stereo matching using maximum-surface techniques. In *Digital Image Computing: Techniques and Applications*, pages 195–200, 1999.

[102] J. Sun, Y. Li, S. B. Kang, and H.-Y. Shum. Symmetric stereo matching for occlusion handling. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 399–406, 2005.

[103] J. Sun, N.N. Zheng, and H.Y. Shum. Stereo matching using belief propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7):787–800, July 2003.

[104] R. Szeliski and D. Scharstein. Sampling the disparity space image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(3):419–425, 2004.

[105] R. Szeliski and R. Zabih. An experimental comparison of stereo algorithms. In *Vision Algorithms: Theory and Practice*, pages 1–19. Springer-Verlag, 1999.

[106] R. Szeliski, R. Zabith, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwale, M. Tappen, and C. Rother. A comparative study of energy minimization methods for markov random fields. In *European Conference on Computer Vision*, pages 19–26, 2006.

[107] M. F. Tappen and W. T. Freeman. Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters. In *International Conference on Computer Vision*, pages 900–908, 2003.

[108] J.-P. Tardif and S. Roy. A MRF formulation for coded structured light. In *3-D Digital Imaging and Modeling*, pages 22–29, 2005.

[109] R. E. Tarjan. *Data structures and network algorithms.* Society for Industrial and Applied Mathematics, 1983.

[110] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision.* Prentice Hall, 1998.

[111] O. Veksler. *Efficient Graph-based Energy Minimization Methods in Computer Vision.* PhD thesis, Conell University, 1999.

[112] O. Veksler. Fast variable window for stereo correspondence using integral images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 556–561, 2003.

[113] O. Veksler. Stereo correspondence by dynamic programming on a tree. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 384–390, 2005.

[114] O. Veksler. Reducing search space for stereo correspondence with graph cuts. In *British Machine Vision Conference*, pages 709–718, 2006.

[115] Olga Veksler. Reducing search space for stereo correspondence with graph cut. In *British Machine Vision Conference*, pages 1834–1841, 2006.

[116] G. Vogiatzis, P. Torr, S. M. Seitz, and R. Cipolla. Reconstructing relief surfaces. In *Proc. of the IEEE Conf. on British Computer Vision*, pages 117–126, 2004.

[117] M. J. Wainwright, T. S. Jaakola, and A. S Willsky. Map estimation via agreement on (hyper)trees: Message-passing and linear programming approches. Technical Report UCB/CSD-3-1269, University of California, 2003.

[118] G.-Q. Wei, W. Brauer, and G. Hirzinger. Intensity- and gradient-based stereo matching using hierarchical gaussian basis functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1143–1160, 1998.

[119] Y. Wei and L. Quan. Asymmetrical occlusion handling using graph cut for multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 902–909, 2005.

[120] A. Witkin, D. Terzopoulos, and M. Kass. Signal matching through scale space. *International Journal of Computer Vision*, 1:133–144, 1987.

[121] K.-J. Yoon and I. S. Kweon. Stereo matching with symmetric cost functions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2371–2377, 2006.

[122] J Zhang, B Curless, and S Seitz. Rapid shape acquisition using color structured light and multi-pass dynamic programming. In *3D Data Processing, Visualization and Transmission*, pages 24–36, 2002.

[123] S. Zhang and P. S Huang. Novel method for structured light system calibration. *Optical Engineering*, 45(8), 2006.

[124] Z. Zhang. A flexible new technique for camera calibration. Technical Report MSR-TR-98-71, Microsoft Research, 1998.

[125] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–3334, 2000.

Annexe A

NON-UNIFORM HIERARCHICAL GEO-CONSISTENCY FOR MULTI-BASELINE STEREO

Cet article [19] a été publié comme l'indique la référence bibliographique.

Marc-Antoine Drouin, Martin Trudeau and Sébastien Roy, Non-Uniform Hierarchical Geo-consistency for Multi-baseline Stereo, *Fourth Canadian Conference on Computer and Robot Vision (CRV'07)*, Montréal, Canada, June 2005.

Abstract

We propose a new and flexible hierarchical multi-baseline stereo algorithm that features a non-uniform spatial decomposition of the disparity map. The visibility computation and refinement of the disparity map are integrated into a single iterative framework that does not add extra constraints to the cost function. This makes it possible to use a standard efficient stereo matcher during each iteration. The level of refinement is increased automatically where it is needed in order to preserve a good localization of boundaries. While two graph-theoretic stereo matchers are used in our experiments, our framework is general enough to be applied to many others. The validity of our framework is demonstrated using real imagery with ground truth.

A.1 Introduction

The goal of binocular stereo is to reconstruct the 3D structure of a scene from two views. Occlusion occurs when part of a scene is visible in one camera but not the

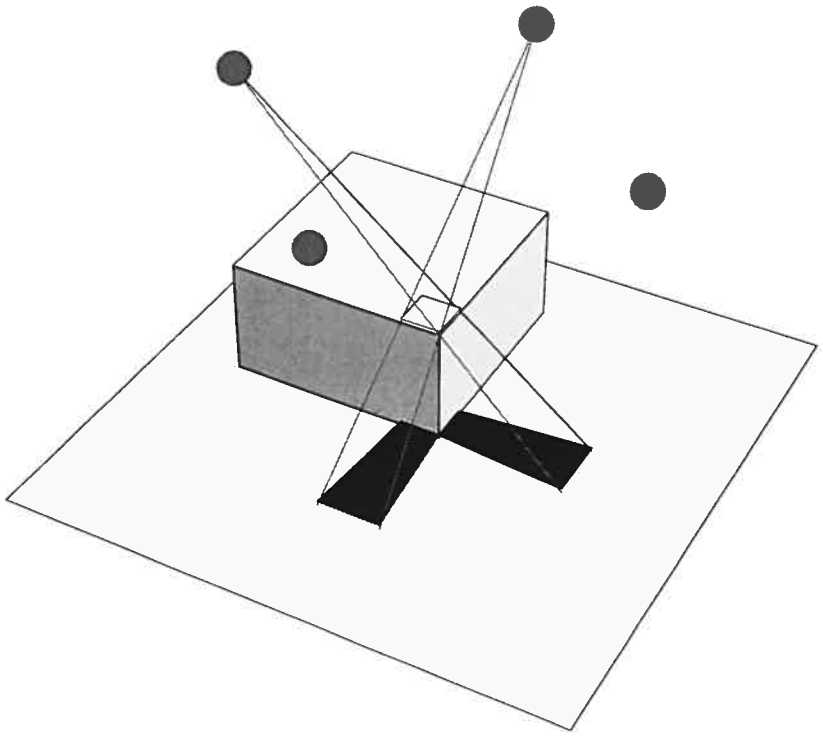

Figure A.1: Example of occlusion. Occluded pixels appear in black, occluders in white.

other (see figure A.1). In this paper, we use a cross-shaped configuration with 4 cameras equidistant to the center one which is the reference. The difficulty of detecting occlusion comes from the fact that it is induced by the 3D structure of the scene, which is unknown until the correspondence is established, as it is the final goal of the algorithm.

In this paper, we propose a new and flexible hierarchical stereo algorithm that features a non-uniform spatial resolution. The visibility computation and refinement of the disparity map are integrated into a single iterative framework that does not add extra constraints to the cost function. The level of refinement is increased automatically where it is needed in order to preserve a good localization of boundaries.

Our algorithm uses the occlusion model proposed by [17]. This model relies on geometric inconsistencies in the disparity map to detect occlusions. As will be shown, our hierarchical algorithm provides major speedups over the non-hierarchical one of [17] while preserving the quality of the final solution. In this paper, we use graph-cut[8] and maximum flow formulation with a linear smoothing term[89]. As in [17], our framework is general enough to be used with other stereo algorithms. A survey paper by Scharstein and Szeliski [94, 93] compares various standard algorithms.

The rest of this paper is divided as follows: in Section A.2, previous work will be presented. Section A.3 describes the visibility modeling framework. Our proposed framework is described in Section A.4. Experimental results are presented in Section A.5.

A.2 Previous works

In Egnal [20], five basic strategies to overcome occlusion for two cameras are presented: left-right checking, bimodality test, goodness Jumps constraint, duality of disparity discontinuity and occlusion, and uniqueness constraint. Some algorithms rely on one or more of these strategies, and are often based on varying a correlation window position or size [52, 31, 112, 53]. Other algorithms use dynamic programming [80, 47, 14] or graph-cut [49, 57]. In [102], visibility and disparity are iteratively minimized using belief propagation.

Many algorithms are specially designed to cope with occlusion in multi-baseline stereo. They can be coarsely divided into three categories [18]. Some approaches are based on visibility heuristics [53, 79, 92, 87, 33, 99, 18]. Others guarantee a solution that is geo-consistent [17, 66, 98, 23, 59]. These approaches preserve the consistency between the recovered visibility and the geometry [17]. Our algorithm belongs to this category. Finally, some algorithms are mixes between heuristic and geo-consistent algorithms [15, 119, 36].

Many hierarchical approaches have been introduced to speed up computation of stereo matching [75, 84, 88, 120, 75, 101, 3, 84, 118, 64, 96, 44, 70, 86]: multiple levels of image reduction are used to reduce the search space. Unfortunately, some matching errors made in an early stage can never be repaired in the following steps. These errors appear mostly near object boundaries. Many approaches have been proposed to cope with errors induced by pyramids [44, 21, 29, 29, 90, 70, 71, 44]. Some methods extend the search interval where large disparity variations are present. Some of them use feature extraction to improve border localization, whilst others used a non-uniform decomposition of the disparity map. Some approaches use multi-resolution techniques to speed up the convergence of an energy function minimization [1, 24].

A.3 Visibility framework

In this section, we review the visibility framework that will be used and that comes from [17]. When doing stereo matching, there is a set $\mathcal{P}$ of reference pixels, for which we want to compute disparity, and a set $\mathcal{D}$ of disparity labels. A $\mathcal{D}$-configuration $f : \mathcal{P} \mapsto \mathcal{D}$ associates a disparity label to every pixel. In order to simplify the discussion, we will always consider the disparity as a positive value independently of the supporting camera used. We also assume that the supporting images are at an equal distance from the reference. To model occlusion, we must compute the volumetric visibility $V_i(\mathbf{p}, d, f)$ of a reference pixel $\mathbf{p}$ located at disparity d from the point of view of a camera i, given a disparity configuration f defined for all other pixels. It is set to 1 if the point is visible and 0 otherwise. The visibility information is collected into a vector, the *visibility mask*

$$V(\mathbf{p}, d, f) = (V_1(\mathbf{p}, d, f), \ldots, V_N(\mathbf{p}, d, f))$$

where N is the number of cameras outside the reference. We call $\mathcal{M}$ the set of all possible visibility masks; an $\mathcal{M}$-configuration $g : \mathcal{P} \mapsto \mathcal{M}$ associates a mask to every pixel. Let us define the special configuration g^0 with $g^0(\mathbf{p}) = (1, \ldots, 1)$ for all $\mathbf{p} \in \mathcal{P}$;

this corresponds to the case where all cameras are visible by all points. The problem is the minimization in f and g of

$$E(f,g) = \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} e(\mathbf{p}, f(\mathbf{p}), g(\mathbf{p}))}_{\substack{\textit{pixel} \\ \textit{likelihood}}} + \underbrace{\sum_{\mathbf{p}\in\mathcal{P}} \sum_{\mathbf{r}\in\mathcal{N}_\mathbf{p}} s(\mathbf{p}, \mathbf{r}, f(\mathbf{p}), f(\mathbf{r}))}_{\textit{pixel smoothing}} \tag{A.1}$$

with the constraint

$$g(\mathbf{p}) \leq V(\mathbf{p}, f(\mathbf{p}), f) \tag{A.2}$$

for each component of these vectors and all $\mathbf{p} \in \mathcal{P}$. The constraint of Eq. A.2 is named *geo-consistency* and the inequality allows the mask to contain a subset of the visible cameras[17]. The removal of some extra cameras has been observed to have little impact on the quality of the solution and is used in many multi-baseline stereo algorithms[79, 17, 15, 18, 53, 79, 92, 87, 33]. The pixel likelihood term is defined as

$$e(\mathbf{p}, d, \mathbf{m}) = \frac{\mathbf{m} \cdot C(\mathbf{p}, d)}{|\mathbf{m}|} \quad \text{for } \mathbf{p} \in \mathcal{P},\ d \in \mathcal{D},\ \mathbf{m} \in \mathcal{M}$$

where $C(\mathbf{p}, d) = (C_1(\mathbf{p}, d), \ldots, C_N(\mathbf{p}, d))$ is the vector of matching costs of the pixel $\mathbf{p}$ at disparity d for each camera. We use $|\mathbf{m}|$ to represent the l_1-norm which is just the number of cameras used for pixel $\mathbf{p}$ at disparity d.

In [17], it was proposed to reduce the dependency between f and g by making it *temporal*: we let f^0 be the $\mathcal{D}$-configuration minimizing $E(f^0, g^0)$ in f and for $t > 0$, let iteratively f^t be the function minimizing $E(f^t, g^t)$ with g^t defined as

$$g^t(\mathbf{p}) = H(\mathbf{p}, f^t(\mathbf{p}), t-1) \tag{A.3}$$

and

$$H_i(\mathbf{p}, d, t) = \prod_{0\leq k\leq t} V_i'(\mathbf{p}, d, f^k) \tag{A.4}$$

where H is a *visibility history mask* and V' is the pseudo-visibility described below. Because of the way g^t is defined, cameras that are removed at one iteration cannot

be kept at the next. This greedy approach guarantee convergence (or stabilization) in a polynomial number of steps. The case where $|g^t(\mathbf{p}, d)| = 0$ is discussed in [17].

In [17] a significant bias was measured in the localization of depth discontinuities. Front objects are enlarged and this discourages the direct use of visibility to update the *visibility history mask*. Instead, a pseudo-visibility

$$V'(\mathbf{p}, d, f) = (V'_1(\mathbf{p}, d, f), \ldots, V'_N(\mathbf{p}, d, f))$$

was introduced, which compensates for the bias by labeling both occluders and occludees as invisible. Discussion about the computation of the pseudo-visibility is postponed until section A.4.1. In this framework, the disparity map is represented as a continuous mesh, this guarantees the preservation of the ordering constraint between the reference and any supporting camera.

In the next section, we will present the integration of our non-uniform hierarchical decomposition of the disparity map with the iterative visibility framework presented in this section.

A.4 Our framework

We propose the use of a non-uniform iterative decomposition of the disparity map. The energy minimization does not associate a disparity to every pixel, but it associates a single disparity to all the pixels included in a block. Each pixel has a visibility mask and it is thus possible to have two pixels in the same block having different masks. At each iteration t, the disparity map is represented as a graph $\mathcal{G}^t = (\mathcal{V}^t, \mathcal{E}^t)$ where $\mathcal{V}^t$ is the set of nodes representing the blocks of pixels and $\mathcal{E}^t$ is the set of edges. An edge between two nodes indicates that smoothing is applied between them (see Fig. A.2). The $\mathcal{D}$-configuration $f^t : \mathcal{V}^t \mapsto \mathcal{D}$ associates a disparity label to every block in $\mathcal{V}^t$. The problem is the minimization in f^t of

$$E_{\mathcal{G}^t}(f^t, g^t) = \sum_{\mathbf{p} \in \mathcal{V}^t} e_{\mathcal{G}^t}(\mathbf{p}, f^t(\mathbf{p}), g^t(\mathbf{p})) + \begin{matrix} \textit{inter-block} \\ \textit{smoothing} \end{matrix} \tag{A.5}$$

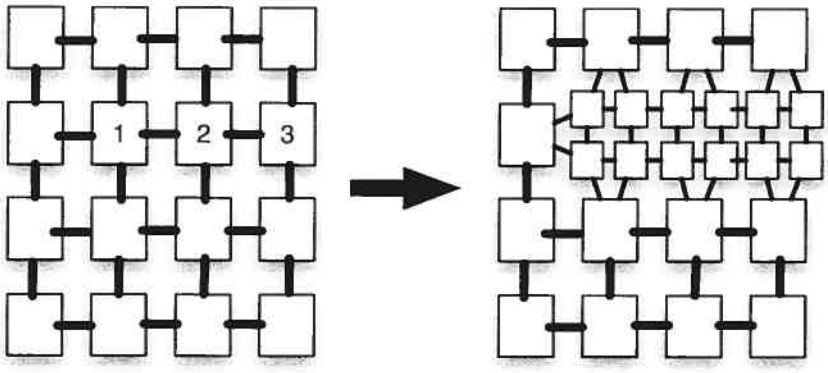

Figure A.2: The division of blocks. Blocks 1 and 3 are labeled as occluder and occludee respectively. Assuming horizontal epipolar lines, the node 2 must also be split.

where g^t is the $\mathcal{M}$-configuration as defined in Eq. A.3. The block likelihood term $e_{\mathcal{G}^t}$ is simply the sum of the likelihoods of pixels included in the block. The inter-block smoothing is simply defined as the sum of the pixel smoothing costs.

In this paper, we used as the initial graph $\mathcal{G}^0$ a regular grid with a user-defined block size. The initial graph could also be obtained from a segmentation of the reference image [67]. Discussion about the block size of the initial graph is postponed until section A.5.3. The initial masks $g^0(\mathbf{p})$ have all cameras visible for all $\mathbf{p}$. Once f^t is found, g^{t+1} and $\mathcal{G}^{t+1}$ can be computed using rendering techniques that will be described in section A.4.1. The pseudo-code of our algorithm is shown in Figure A.4. Since the blocks that are split at one iteration cannot be merged at a later one, the convergence (or stabilization) is guaranteed. This is achieved in a polynomial number of steps. Indeed, $|\mathcal{V}^t|$ is monotonically increasing with t and is at most $|\mathcal{P}|$ and $H(\mathbf{p}, d, t)$ is monotonically decreasing in t for all $\mathbf{p}$ and d. Moreover, if $\mathcal{G}^t = \mathcal{G}^{t+1}$ and $H(\mathbf{p}, d, t-1) = H(\mathbf{p}, d, t)$ for all $\mathbf{p}$ and d, then $f^t = f^{t+1}$ since both are solutions to the same minimization problem, and the process has stabilized. The algorithm converges (or stabilizes) to a geo-consistent solution, but can go through intermediate

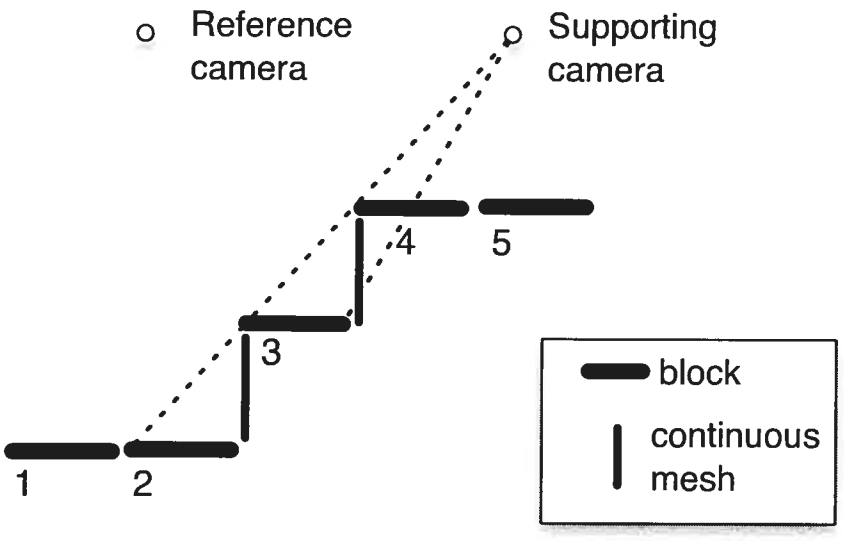

Figure A.3: Nodes 2 and 3 are occludees and node 4 is an occluder.

ones that are not.

A.4.1 Pseudo-visibility and block division

A block is called occluder when one of its pixels occludes a pixel from another blocks (called occludee). When the blocks contain many pixels, the discontinuity between an occluder and an occludee is probably poorly located. Both blocks must be split in order to improve the localization of the depth discontinuity at the next iteration. The labeling of blocks as occludee and occluder is done concurrently with the pseudo-visibility computation. The pseudo-visibility masks V_i' are computed by using rendering techniques. Two renderings of the current disparity map f are done from the point of view of each supporting camera i: one with an regular Z-buffer and one with a reverse Z-buffer test. Two disparity maps S_i^f and L_i^f are thus obtained and they contain the minimal and maximal pixelwise disparity observed by the supporting

NON-UNIFORM HIERARCHICAL GEO()

1 Build $\mathcal{G}^0$
2 $g^0(\mathbf{p}) \leftarrow (1, ..., 1) \quad \forall \mathbf{p} \in \mathcal{P}$
3 $t \leftarrow 0$
4 $f_{\mathcal{G}^t} \leftarrow \arg\min_f E_{\mathcal{G}^t}(f, g^t)$
5 Render $f_{\mathcal{G}^t}$ in all supporting cameras
6 Compute g^{t+1} and $\mathcal{G}^{t+1}$ using render buffers of previous line
7 **if** $g^t = g^{t+1}$and $\mathcal{G}^t = \mathcal{G}^{t+1}$
8 **then return** $f_{\mathcal{G}^t}$
9 $t \leftarrow t + 1$
10 **goto** 4

Figure A.4: Algorithm overview

camera i. In [17], the pseudo-visibility function $V_i'(\mathbf{p}, d, f)$ is computed as

$$V_i'(\mathbf{p}, d, f) = \delta\left(S_i^f(T_i(\mathbf{p}, d)) - L_i^f(T_i(\mathbf{p}, d))\right)$$

where δ is 1 at 0 and 0 elsewhere and $T_i(\mathbf{p}, d)$ is the projection pixel $\mathbf{p}$ at disparity d in the supporting camera i. This approach might not detect all the occluded pixels. As an example, using this rendering technique, blocks 2 and 4 of figure A.3 are correctly labeled as occludee and occluder. Nevertheless, this technique does not detect the occluded block 3. We propose an approach that identifies all the occluded blocks. The color channels are used to encode the index of each node in $\mathcal{G}^t$. Since a *visibility mask* is assigned to every pixel, the index of a pixel in the block is also encoded using one of the color channel. When $S_i^t(\mathbf{r})$ and $L_i^t(\mathbf{r})$ are different, we can used the color buffer to identify the occluder and occluded blocks having the smallest disparities. Since a continuous mesh representation is used, all blocks (and pixels) between those two along the epipolar line must be both occluder and occludee; the blocks must be split

and the visibility masks associated to their pixels must be updated (see Fig. A.2). Note that when using our camera configuration, this rendering process can be sped up by replacing it by a line drawing using depth buffers.

A.4.2 Stereo matcher

At each iteration of our algorithm, a disparity map is computed. Many algorithms are capable of computing non-uniform disparity maps, for instance belief propagation [103], reweighted message passing [55], dynamic programming on tree[113, 67], graph-cut [8] and maximum flow formulation with a linear smoothing term [89].

When belief propagation is used, the local evidences can be initialized using the values of the previous iteration. In this case, our algorithm is an occlusion modeling extension of the hierarchical belief propagation presented in [24] where the uniform decomposition is replaced by our non-uniform one. Note that bipartite scheduling is no longer possible, but the distance transform can still be used.

Note that search space reduction techniques that reduce the number of disparity labels that must be examined could also be used [114, 90].

A.5 Experimental results

In all our experiments, the matching cost function was the same for all algorithms, that of [59] which is based on [4]. We used color images but only gray scale ones are shown here. As for the pixel smoothing term, we used the experimentally defined smoothing function that also comes from [59]:

$$s(\mathbf{p}, \mathbf{r}, f(\mathbf{p}), f(\mathbf{r})) = \lambda\, h(\mathbf{p}, \mathbf{r})\, (f(\mathbf{p}), f(\mathbf{r}))$$

where h is defined as

$$h(\mathbf{p}, \mathbf{r}) = \begin{cases} 3 & \textit{if } |I_{ref}(\mathbf{p}) - I_{ref}(\mathbf{r})| < 5 \\ 1 & \textit{otherwise} \end{cases}$$

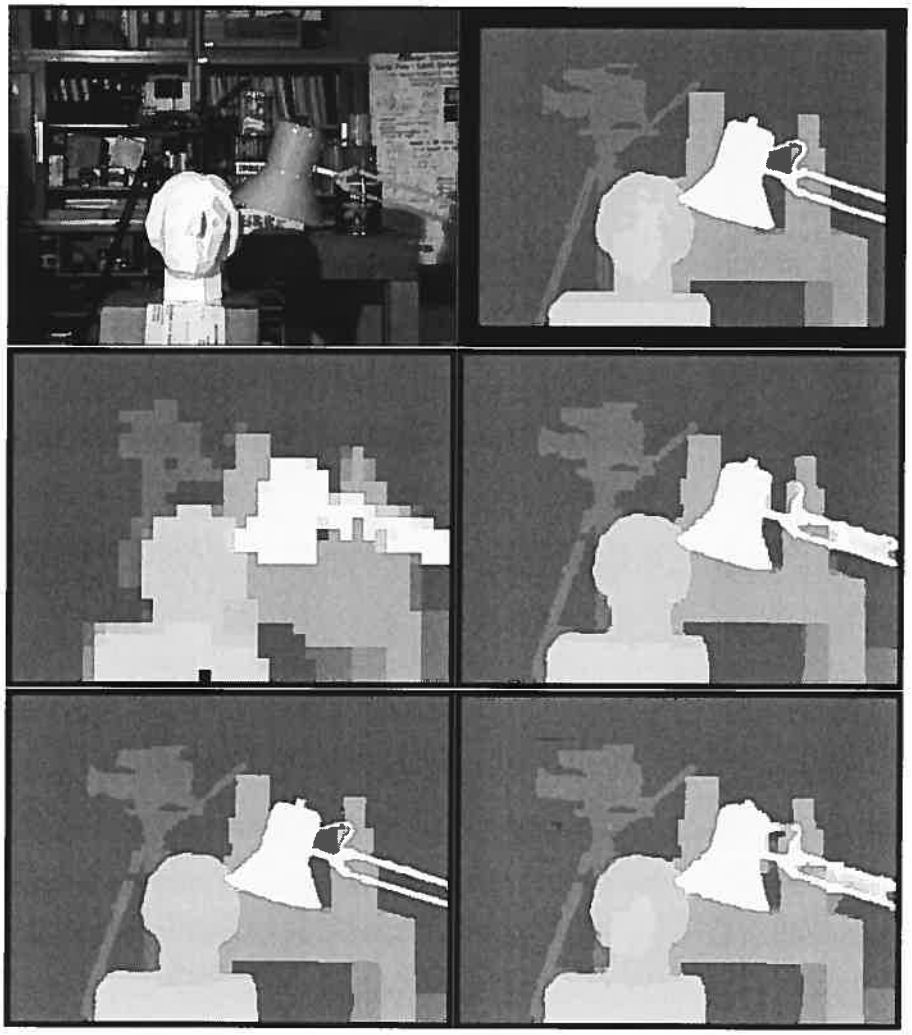

Figure A.5: Reference images for the Head and Lamp (top left) and ground truth (top right). Disparity map obtained from NU-GEO-MF after first (middle left) and last iterations (middle right). The result of applying border-cut to NU-GEO-MF (bottom left) and the result of GEO-MF (bottom right) are shown as well.

with $l(d, d') = |d - d'|$ for the maximum flow with a linear smoothing term (MF) [89] and $l(d, d') = \delta(d - d')$ for graph-cut (BNV)[8]. A pixel disparity is considered erroneous if it differs by more than one disparity step from the ground truth. This error measurement is used by two comparative studies for 2-camera stereo [105, 94]. Note that a better smoothing term and cost function are now available [121]. We did not used them to simplify comparisons.

As in [17], we keep a single *visibility history mask* for each pixel $\mathbf{p}$ regardless of the disparity d. The Eq. A.4 becomes

$$H_i(\mathbf{p}, t) = \prod_{0 \leq k \leq t} V_i'(\mathbf{p}, f^k(\mathbf{p}), f^k).$$

This saves memory but the convergence is no longer guaranteed. We simply stop iterating when $\mathcal{G}^t = \mathcal{G}^{t+1}$ and $H(\mathbf{p}, t) = H(\mathbf{p}, t-1)$ for all $p \in \mathcal{P}$.

Recently, a new type of post-processing algorithms named border-cut (BC) was proposed[18]. Rather than associating a disparity label to every pixel, it associates a position to every disparity discontinuity. This method obtains sharp and well-located disparity discontinuities starting from the output of a wide range of stereo matchers. We provide results with and without this post-processing step. In our experiments, we used the dataset from the Multiview Image Database from the University of Tsukuba.

A.5.1 Head and Lamp scene

This dataset is composed of a 5×5 image grid. Each image has a resolution of 384×288 (see Fig. A.5). The search interval was between 0 and 15 pixels and we used 16 disparity steps. Some disparity maps are shown in Fig. A.5 and error percentages are given in Table A.1. Since we use a visibility framework that preserved the ordering constraint, we also computed the error after removing the pixels breaking the ordering constraint, in particular part of the arm of the lamp. We use the same mask as in [17], the one determined by re-projecting the ground truth in each supporting camera. We provide results for our non-uniform framework using the stereo matcher

Algorithm	Error (whole image)	Error (mask)
KZ1	2.3	-
KZ1'	1.3	-
ASYM-KZ1	1.3	-
REL-DP	1.9	-
NAKA-BNV	1.7	-
HYBRID-IDP	1.7	-
GEO-BNV pt	2.2	1.5
GEO-BNV	2.5	1.6
GEO-MF	2.9	2.0
BC (average)	1.1	-
NU-GEO-MF	2.4	1.5
NU-GEO-BNV	2.5	1.7

Table A.1: Percentages of error of the different algorithms for Head and Lamp scene, using 5 images.

BNV and MF. We label them NU-GEO-BNV and NU-GEO-MF. The non-hierarchical versions are labeled GEO-BNV and GEO-MF and come from [17]. The entries ASYM-KZ1 and KZ1' come directly from [119]. KZ1 and REL-DP come from [59] and [36] respectively. HYBRID-IDP and NAKA-BNV come from [15]. The error rate associated with border-cut (BC) comes from [18] and is the average of the error rates obtained using different initializations. GEO-BNV and GEO-BNV pt come from [15]. We also compared with GEO-MF and our implementation of GEO-MF, that achieved a lower error rate than presented in [17]. The error rate of NU-GEO-MF is 2.4% while GEO-MF obtained a higher error rate of 2.9%. The number of iterations is respectively 12 and 6 for NU-GEO-MF and GEO-MF. The total number of elementary operations is 1.5 million for NU-GEO-MF and 7.1 million for GEO-MF.* The running time for NU-GEO-MF and GEO-MF are 55 and 161 seconds respectively using an AMD Athlon(tm) 64 Processor 3500+. Figure A.6 shows the number of elementary

* We count the number of Discharge operations in the preflow-push-relabel algorithm.

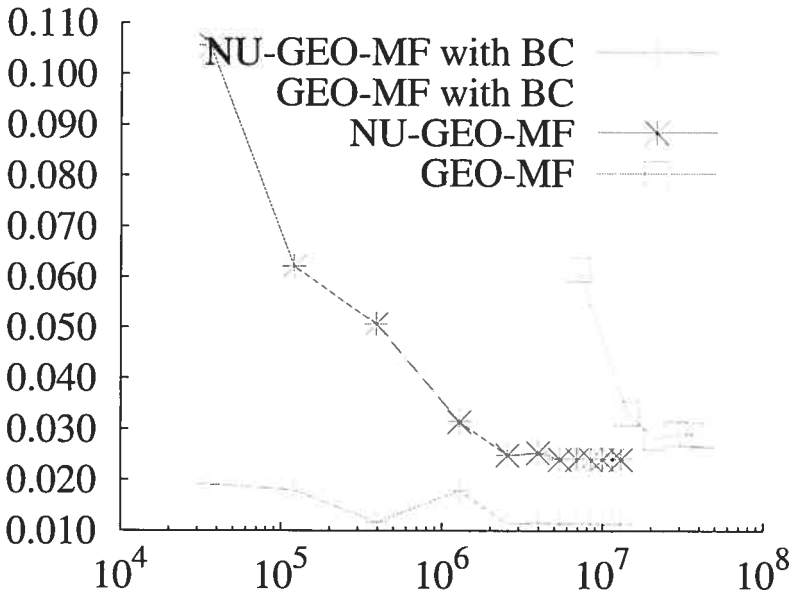

Figure A.6: Variation of the error rate as a function of the number of elementary operations.

operations giving the error rate for NU-GEO-MF and GEO-MF. After 7 iterations, NU-GEO-MF achieved a lower error rate than GEO-MF at a fraction of the cost of computing a single iteration of GEO-MF. The border-cut algorithm was run using the disparity maps obtained after each iteration of GEO-MF and NU-GEO-MF. Using both initialization, the error rates after a few iterations are similar to those obtained in [18]. The parameters for the border-cut are those used in [18].

The initial block size used in our experiments is 10×10 and the initial disparity map is shown in figure A.5. For NU-GEO-MF and NU-GEO-BNV the non-uniform decomposition allows a reduction of approximatively 80% of the problem space. Table A.2 shows the stability to change of the smoothing parameter of NU-GEO-MF, giving the error percentage for 8 values of this parameter.

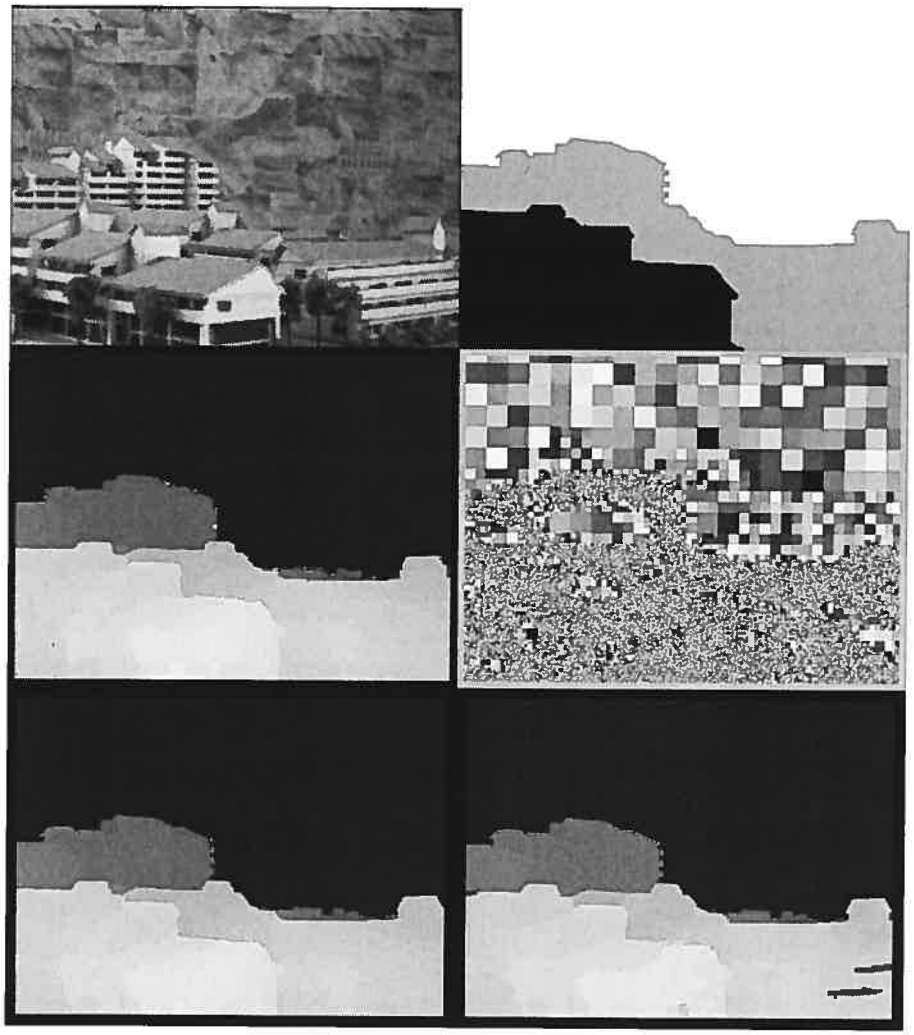

Figure A.7: Reference image for the City scene (top left). The *partial* discontinuity ground truth (top right). Disparity map obtained from NU-GEO-MF (middle left) and improved by border-cut (bottom left). Final graph for NU-GEO-MF (middle right). Disparity map obtained by border-cut starting from the result of Hybrid-IDP (bottom right).

Algorithm	Smoothing parameter							
	1	2	4	8	16	32	64	128
NU-GEO-MF	2.61	2.51	2.51	2.51	2.71	3.04	3.27	5.90

Table A.2: Resistance to change of the smoothing parameter for the Head and Lamp scene. The parameter increases by a factor of 128, while the error rate varies by less than 3.29%.

A.5.2 City scene

This dataset contains 81 640 × 480 images in a 9 × 9 grid (Fig. A.7). We only used 5 images in a cross configuration. Each disparity map was computed using 44 disparity steps and the search interval was between 0 and 43 pixels. The initial block size was 30 × 30. A *partial* discontinuity ground truth that comes from [18] is shown in Fig. A.7. A discontinuity location is considered erroneous if it differs by more than one pixel from the ground truth. The results are presented in Table A.3 and some disparity maps are shown in Fig. A.7. We also show the result obtained using Hybrid-IDP [15] with the border-cut post-processing. The latter algorithm is fast, however it introduces artifacts that are not eliminated by border-cut. The disparity maps obtained by NU-GEO-MF before and after border-cut are shown in Fig. A.7. The percentage of pixels with a difference in disparity greater than one for GEO-MF and NU-GEO-MF is only 0.5%. After applying border-cut on each disparity map, this difference drops to .2%. These different pixels are highlighted in Fig. A.7 by saturating the red channel. Although the disparity maps obtained are almost identical, the reduction in graph sizes is significant. The final non-uniform decomposition of the disparity map is shown in Fig. A.7. The reduction of the search space is approximatively 75%.

Algorithm	Before B-C (best γ)	After B-C (fixed γ)
NU-GEO-MF	18.8	11.8
GEO-MF	23.2	10.4
GEO-BNV	15.4	11.4
NU-GEO-BNV	15.6	12.6
Hybrid-IDP	28.2	14.7

Table A.3: Percentage of error in the discontinuity location, according to the *partial* ground truth, of the different algorithms for City scene, before and after border-cut.

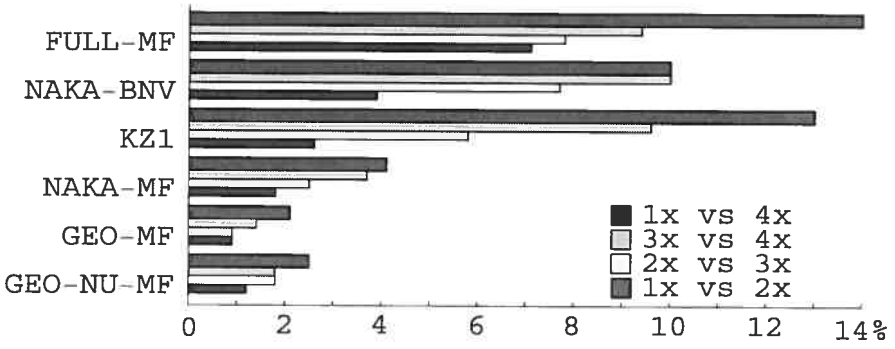

Figure A.8: Resistance to baseline change for 6 algorithms for the Santa scene; each bar represents a percentage of incompatible pixels between depth maps obtained for two different baselines.

A.5.3 Santa scene

This dataset contains 81 images in a 9×9 grid (see figure A.10). We only used 5 images in a cross-shaped configuration. As the baseline increases, the amount of occlusion in the scene increases as well. To measure the level of resistance to change of the baseline, we used the incompatibility metric introduced in [17]. Figure A.8 contains bar charts of the percentages of pixels incompatible between the depth maps obtained for two baselines. Images were reduced by a factor of 2 to achieve a resolution of 320×240 and 23 disparity steps were used. In addition to GEO-MF and NU-GEO-MF, results from 4 algorithms coming from [17] were included. Our hierarchical approach is slightly less resistant to changes of baseline. The reduction in search space for baselines 1x to 4x is 65%, 62%, 29% and 30% respectively. The small reduction for the largest baselines is explained by the geometric inconsistencies introduced by by non-lambertian surfaces. Indeed, these surfaces move significantly when the baseline is large. Figure A.10 shows disparity maps obtained using initial blocks of 130×130 pixels on the full size image using 45 disparity steps. The incompatibility between GEO-MF and NU-GEO-MF for different initial block sizes is shown in figure A.8 for both the city and Santa scenes. A pixel is considered incompatible when its disparities are different ($= 0$) or differ by more than one (> 1).

A.5.4 Conclusion

We proposed a new and flexible hierarchical stereo algorithm that features a non-uniform spatial resolution. The visibility computation and refinement of the disparity map are integrated into a single iterative framework. The disparity map is represented as a graph and the nodes are split using the visibility information. The levels of refinement is increased automatically where they are needed most to preserve accurate localization of object boundaries. Our framework does not add extra constraints to the cost function and is very indifferent to the choice of the initial block size. As has

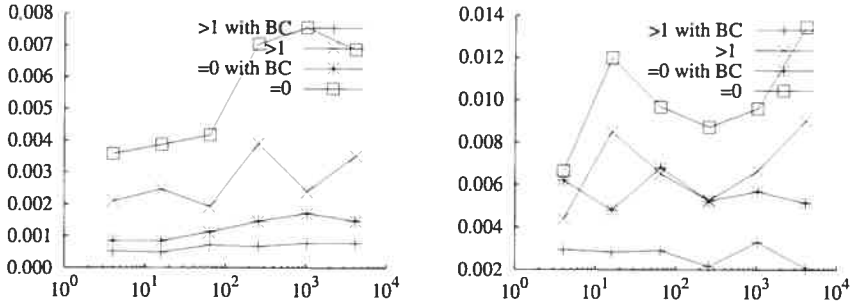

Figure A.9: Ratio of incompatible pixels between GEO-MF and NU-GEO-MF as a function of the initial block size for the city (left) and the Santa scenes (right). The block size varies from 4 to 4096 pixels.

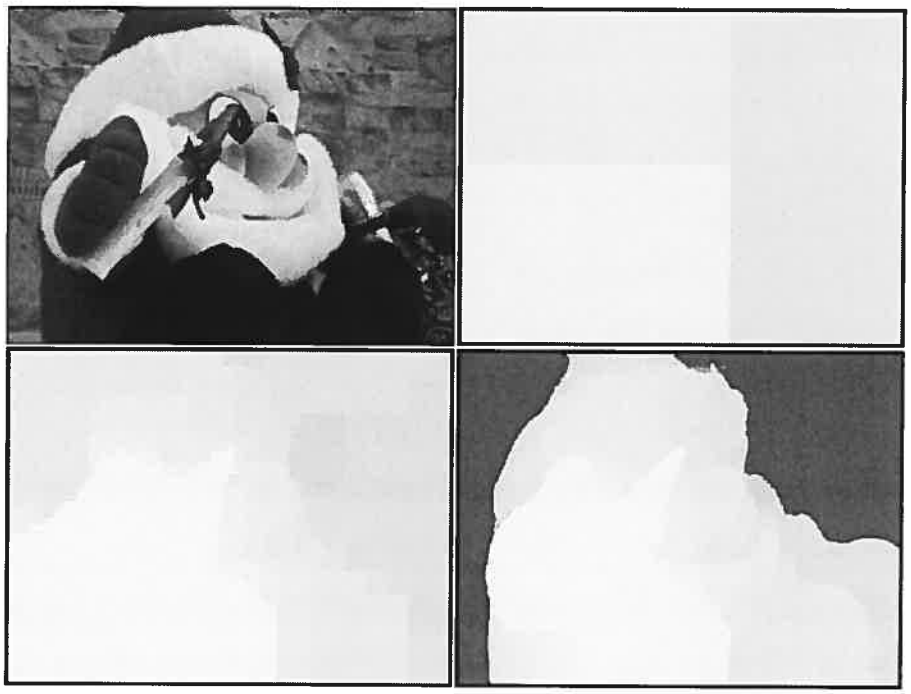

Figure A.10: Reference image for the Santa scene and disparity maps after different iteration of NU-GEO-MF.

been shown, our algorithm allows major speedups and preserves the quality of the final solution. While two graph-theoretic stereo matchers are used in our experiments, our framework is general enough to be applied to many others.

As for future work, the extension of our approach to full volumetric reconstruction, where occlusion becomes the dominant problem, should be investigated.