Université de Montréal

Le cinéma omnistéréo ou l'art d'avoir des yeux tout le tour de la tête

par

Vincent Chapdelaine-Couture

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

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

Décembre 2011

Université de Montréal

Faculté des arts et des sciences

Cette thèse intitulée

Le cinéma omnistéréo ou l'art d'avoir des yeux tout le tour de la tête

présentée par

Vincent Chapdelaine-Couture

a été évaluée par un jury composé des personnes suivantes :

Jean Meunier
Président-rapporteur
Représentant du doyen de la FAS

Sébastien Roy
Directeur de recherche

Derek Nowrouzezahrai
Membre du jury

Thierry Dutoit
Examinateur externe

Thèse acceptée le 24 avril 2012

RÉSUMÉ

Cette thèse s'intéresse à des aspects du tournage, de la projection et de la perception du cinéma stéréo panoramique, appelé aussi cinéma omnistéréo. Elle s'inscrit en grande partie dans le domaine de la vision par ordinateur, mais elle touche aussi aux domaines de l'infographie et de la perception visuelle humaine.

Le cinéma omnistéréo projette sur des écrans immersifs des vidéos qui fournissent de l'information sur la profondeur de la scène tout autour des spectateurs. Ce type de cinéma comporte des défis liés notamment au tournage de vidéos omnistéréo de scènes dynamiques, à la projection polarisée sur écrans très réfléchissants rendant difficile l'estimation de leur forme par reconstruction active, aux distorsions introduites par l'omnistéréo pouvant fausser la perception des profondeurs de la scène.

Notre thèse a tenté de relever ces défis en apportant trois contributions majeures. Premièrement, nous avons développé la toute première méthode de création de vidéos omnistéréo par assemblage d'images pour des mouvements stochastiques et localisés. Nous avons mis au point une expérience psychophysique qui montre l'efficacité de la méthode pour des scènes sans structure isolée, comme des courants d'eau. Nous proposons aussi une méthode de tournage qui ajoute à ces vidéos des mouvements moins contraints, comme ceux d'acteurs. Deuxièmement, nous avons introduit de nouveaux motifs lumineux qui permettent à une caméra et un projecteur de retrouver la forme d'objets susceptibles de produire des interréflexions. Ces motifs sont assez généraux pour reconstruire non seulement les écrans omnistéréo, mais aussi des objets très complexes qui comportent des discontinuités de profondeur du point de vue de la caméra. Troisièmement, nous avons montré que les distorsions omnistéréo sont négligeables pour un spectateur placé au centre d'un écran cylindrique, puisqu'elles se situent à la périphérie du champ visuel où l'acuité devient moins précise.

Mots clés: cinéma, omnistéréo, immersion, panoramique, stéréo, vision par ordinateur, perception visuelle, reconstruction active, expérience psychophysique.

ABSTRACT

This thesis deals with aspects of shooting, projection and perception of stereo panoramic cinema, also called omnistereo cinema. It falls largely in the field of computer vision, but it also in the areas of computer graphics and human visual perception.

Omnistereo cinema uses immersive screens to project videos that provide depth information of a scene all around the spectators. Many challenges remain in omnistereo cinema, in particular shooting omnistereo videos for dynamic scenes, polarized projection on highly reflective screens making difficult the process to recover their shape by active reconstruction, and perception of depth distortions introduced by omnistereo images.

Our thesis addressed these challenges by making three major contributions. First, we developed the first mosaicing method of omnistereo videos for stochastic and localized motions. We developed a psychophysical experiment that shows the effectiveness of the method for scenes without isolated structure, such as water flows. We also propose a shooting method that adds to these videos foreground motions that are not as constrained, like a moving actor. Second, we introduced new light patterns that allow a camera and a projector to recover the shape of objects likely to produce interreflections. These patterns are general enough to not only recover the shape of omnistereo screens, but also very complex objects that have depth discontinuities from the viewpoint of the camera. Third, we showed that omnistereo distortions are negligible for a viewer located at the center of a cylindrical screen, as they are in the periphery of the visual field where the human visual system becomes less accurate.

Keywords: cinema, omnistereo, immersion, panoramic, stereo, computer vision, visual perception, active reconstruction, psychophysic experiment.

TABLE DES MATIÈRES

LISTE DES FIGURES

REMERCIEMENTS

Je tiens tout d'abord à remercier mon directeur de recherche, Sébastien Roy, de m'avoir laissé la liberté de choisir le sujet de ma thèse, et pour toutes les discussions à différentes étapes de mon travail.

Je tiens également à remercier Michael Langer, professeur à l'université McGill, de m'avoir donné l'opportunité de débuter le doctorat, pour nos discussions et sa collaboration à la rédaction de plusieurs de mes articles.

Je remercie deux professeurs rencontrés lors de conférences qui m'ont conseillé sur l'analyse de la perception visuelle : Marty Banks pour m'avoir suggéré d'évaluer la perception omnistéréo de façon plus qualitative, et Ian Howard pour m'avoir fourni une partie de son livre *Seeing in Depth* non encore publiée.

Cette thèse a nécessité l'élaboration d'un écran panoramique stéréo. Je remercie Jean Piché d'avoir approuvé l'achat du matériel, de même que Jean-Michel Dumas et Martin Marier pour les nombreuses démarches administratives ; l'équipe d'eXtension concepts pour la conception de la structure de l'écran panoramique, en particulier Yves Loignon pour sa grande disponibilité ; Manfred Mattis pour avoir répondu rapidement à mes nombreuses questions en rapport au tissu *SilverFabrics*.

Je remercie tous les membres du laboratoire que j'ai eu le plaisir de côtoyer durant ces années, particulièrement Louis Bouchard pour m'avoir aidé au montage de l'écran et avoir voulu tester les nombreuses versions de mes expériences perceptuelles.

Je remercie finalement ma famille, plus particulièrement mes parents pour leur soutien et leur participation comme acteurs et co-scénaristes aux différents tournages que j'ai réalisés, et mon chat Mylie qui, loin de suivre mes directives lors des tournages, a bien voulu, entre deux roupillons, se laisser filmer courant dans les feuilles d'automne.

INTRODUCTION

La vision par ordinateur cherche à définir des algorithmes qui permettent l'analyse de l'environnement visible à partir d'images. Cette thèse porte sur les images stéréo panoramiques, appelées omnistéréo[1], c'est-à-dire une paire d'images panoramiques, une pour l'oeil gauche et une pour l'oeil droit, qui permet une perception binoculaire de la profondeur d'une scène[2] tout autour de spectateurs. Les images omnistéréo de scènes fixes ont été utilisées à l'origine dans le domaine de la robotique pour permettre à un robot de découvrir son environnement. Notre thèse s'intéresse, entre autres, au tournage *vidéo* omnistéréo de scènes dynamiques, ce qui ouvre la voie au tournage du cinéma omnistéréo.

Cinéma immersif

Dès les débuts du cinéma à la fin du XIX^{e} siècle, des pionniers visent le projet ambitieux d'une immersion totale du spectateur. Ainsi de nombreuses innovations techniques vont jalonner l'histoire du cinéma. Parmi celles-ci, nous retrouvons l'immersion stéréoscopique et l'immersion panoramique.

L'immersion par la stéréoscopie naît dans les années 1930, renaît dans les années 1950, et réapparaît dans les 1980. La stéréoscopie (ou stéréo) tente de reproduire la perception des profondeurs d'une scène (le relief) à partir de deux images, une image pour l'oeil gauche et une pour l'oeil droit (voir fig. 1(a)). La fusion de ces deux images par le cerveau permet de percevoir les profondeurs de la scène. Aujourd'hui,

[1] Le mot *omnistéréo*, introduit par Peleg *et al.* [49], est composé du préfixe des mots *omni*directionnel et *stéréo*scopique.

[2] Dans ce travail, la scène fait référence à un lieu ou un ensemble d'objets réels éclairés par des sources lumineuses afin d'en faire une capture par caméras. La scène devient virtuelle lorsque les images capturées sont projetées sur un écran.

la stéréo semble plus que jamais d'actualité avec les écrans IMAX 3D et l'arrivée de la télévision 3D. Les techniques d'animation numérique en facilitent la production, mais le processus exige des investissements majeurs tant au niveau des salles de projection que du processus de production.

Quant à l'immersion panoramique, elle utilise souvent un écran large, courbe et placé devant les spectateurs, comme le Cinérama[3] ou IMAX DOME[4]. Le tournage et la projection pour IMAX DOME se font à l'aide d'une caméra et d'un projecteur IMAX[5] très haute résolution munis d'une lentille très grand angle. L'écran peut aussi entourer les spectateurs, comme le Cineorama[6] ou le Circle-Vision[7]. Comme nous le verrons plus loin, l'utilisation d'un écran 360° complique grandement l'ajout de la stéréo, puisque l'orientation du regard des spectateurs est inconnue. La prise d'images panoramiques par caméras trouve aussi d'autres applications comme *Google Street View*[8] qui utilise le système Ladybug [52] à 5 caméras ou plus pour couvrir 360° (voir fig. 1(b)).

La combinaison du panoramique et de la stéréo donne le cinéma immersif de type omnistéréo. Il est important de noter que nous faisons ici référence à une immersion stéréo qui couvre 360° autour des spectateurs, contrairement au système

[3] Créé en 1952, le Cinérama, dont le nom est une contraction de cinéma et de panorama, utilise trois caméras synchronisées pour la prise de vue et un écran de projection très large couvrant 146°.

[4] IMAX DOME projette sur un dôme incliné de façon à couvrir à 172° à l'horizontale et 140° à la verticale.

[5] La résolution IMAX est typiquement de 10000×7000 pixels.

[6] Le Cineorama utilisait 10 caméras pour le tournage et le même nombre de projecteurs pour couvrir un écran circulaire 360°. Il a été présenté à l'Exposition universelle de 1900.

[7] Le Circle-Vision de Disney, dont la première utilisation remonte à 1955, utilise neuf caméras pour le tournage et le même nombre de projecteurs pour couvrir un écran de 360°.

[8] Google Street View donne une vue 360° dans les rues de plusieurs villes à travers le monde.

IMAX SOLIDO[9] ou l'attraction King Kong 360° 3D[10]. Le cinéma stéréo traditionnel projette généralement des images stéréo sur une surface quasi planaire, et l'orientation relative des spectateurs par rapport à cette surface est connue puisqu'ils y font face. Ainsi lors du tournage, l'orientation de la caméra stéréo fait face à l'action, comme le spectateur fait face à l'écran. Mais le cinéma omnistéréo peut projeter sur un écran cylindrique ou un dôme, ce qui complique le tournage parce que l'orientation du regard des spectateurs est inconnue. Le tournage et la projection omnistéréo ne permettent pas pour l'instant une perception stéréo pour toutes les orientations à la fois sans créer des distorsions.

Dans le cadre de ce travail, nous avons choisi de construire un écran de forme cylindrique, d'un rayon de 230cm et d'une hauteur de 150cm, fait d'un tissu argenté qui permet une projection omnistéréo polarisée. Cet espace peut contenir au plus une quinzaine de spectateurs, qui doivent porter des lunettes conçues pour que chaque oeil voit l'image qui lui est destinée. Ce type d'écran s'apparente aux environnements CAVE [17, 18], mais évite la présence de coins qui ajoutent des distorsions au niveau de la perception des profondeurs, tel que nous le décrivons au chapitre 6. Les environnements CAVE sont formés de 4, 5 ou 6 écrans de projection plats disposés en cube. Certains environnements CAVE utilisent un système de suivi de la tête du spectateur pour calculer les images stéréo exactes à partir de son point de vue. Les premières projections omnistéréo datent du milieu des années 1990, où des environnements CAVE ont délaissé le suivi du spectateur pour projeter des images stéréo adaptées à l'orientation de chaque mur.

De plus, Peleg *et al.* [49] ont proposé des concepts de lentilles ou miroirs omnistéréo, comme des miroirs en spirale ou des lentilles à milliers de segments,

[9] IMAX SOLIDO ou IMAX 3D Dynamique, que l'on peut voir au parc d'attraction du Futuroscope à Poitiers, combine, entre autres, la technologie du stéréo à celle de IMAX DOME.

[10] L'attraction King Kong 360° 3D, présentée aux Universal Studios à Hollywood, se rapproche du cinéma omnistéréo en utilisant deux écrans stéréo courbes de chaque côté d'un train. Cependant, le spectateur ne peut regarder ni à l'avant ni à l'arrière du train.

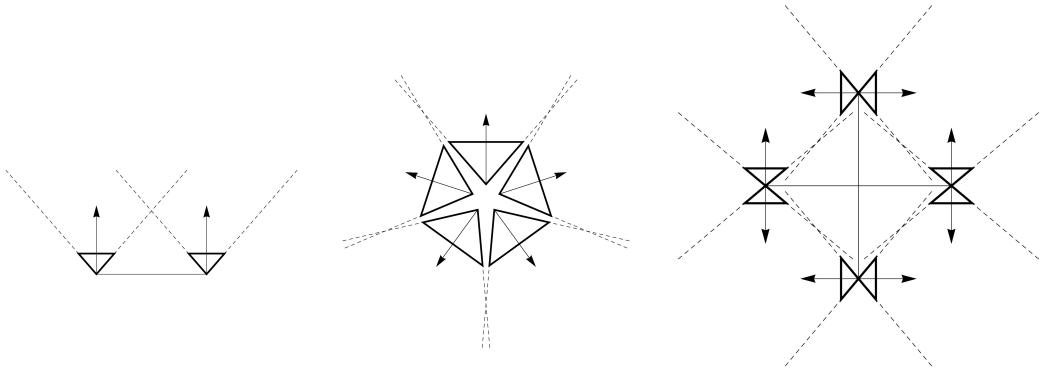


Figure 1. Exemple de disposition de caméras pour un tournage stéréo (a), panoramique (b) ou omnistéréo (c). Les flèches indiquent l'orientation des caméras. Les lignes pointillées montrent leur champ de vision.

mais ceux-ci n'ont toujours pas été réalisés. Gluckman *et al.* [24] ont considéré une caméra panoramique au-dessus d'une autre, mais les images capturées ne peuvent être utilisées directement lors d'une projection omnistéréo parce que les caméras sont séparées verticalement alors que nos yeux le sont horizontalement. C'est pourquoi la production d'images omnistéréo se fait pour l'instant par l'assemblage de multiples prises de vue. Nous avons mentionné précédemment que des systèmes panoramiques à plusieurs caméras existent, mais ces systèmes n'ont pas la contrainte de capturer une image pour chaque oeil, comme pour l'omnistéréo. L'ajout de l'omnistéréo au système Ladybug, par exemple, requerrait de doubler le nombre de caméras (de 5 à 10), et de les disposer de telle sorte qu'aucune ne cache la vue d'une autre (voir fig. 1(c) pour un exemple à huit caméras). De plus, ce système serait très dispendieux.

Dans cette thèse, nous tentons notamment de montrer que l'utilisation de méthodes de vision par ordinateur permet de réaliser un cinéma omnistéréo très haute résolution à des coûts abordables, tant au tournage qu'à la projection.

Contributions et structure de la thèse

Cette thèse par articles se divise en trois parties en rapport respectivement avec le tournage, la projection et la perception du cinéma omnistéréo. Chaque partie comprend un chapitre d'introduction aux notions de base nécessaires à sa compréhension, ainsi qu'un chapitre qui présente une de nos trois contributions majeures sous forme d'un article de journal.

La première partie traite du tournage omnistéréo à travers les chapitres 1, 2 et 3. Au chapitre 1, nous présenterons les problématiques liées au tournage omnistéréo de scènes dynamiques, en particulier celle de la parallaxe de mouvement. Nous résumerons quelques méthodes pertinentes et introduiront des notions de vision par ordinateur utiles à la création de vidéos omnistéréo. Nous présenterons au chapitre 2 la toute première méthode de création de vidéos omnistéréo par assemblage d'images. Cette méthode touche les domaines de la vision par ordinateur, de la perception visuelle et de l'infographie (nous faisons ici référence à l'infographie comme domaine qui vise à produire des images par des moyens informatiques). Nous verrons que ces vidéos peuvent être jouées en boucle sans coupure de mouvement et projetées jusqu'à 360° autour des spectateurs. Cette méthode suppose une scène de mouvements stochastiques et localisés. Nous avons élaboré une expérience psychophysique pour analyser notre méthode au niveau perceptuel. Cette expérience a été approuvée officiellement par le Comité d'éthique de la recherche de la Faculté des arts et des sciences (CÉRFAS). Une vingtaine de volontaires y ont participé. Nous en présenterons les résultats qui permettent de conclure que la méthode fonctionne pour des scènes sans structure isolée, comme des courants d'eau, mais qu'elle crée des duplications visibles pour des objets bien définis en mouvement, comme des branches au vent. Nous discuterons au chapitre 3 d'une méthode de tournage omnistéréo qui permet l'ajout de mouvements plus généraux comme ceux des acteurs. Nous avons

testé notre méthode par le tournage de plusieurs séquences et d'un court métrage omnistéréo d'une durée d'un peu plus de 4 minutes.

La deuxième partie porte sur la projection omnistéréo à travers les chapitres 4 et 5. Au chapitre 4, nous présenterons la multi-projection comme moyen de réaliser une projection très haute résolution à moindre coût. L'alignement automatique des projecteurs nécessite une correspondance caméra-projecteur. Nous expliquerons la problématique de l'illumination indirecte dans des écrans omnistéréo et présenterons trois méthodes de lumière structurée liées à cette problématique. Nous traiterons au chapitre 5 du problème de la reconstruction active de surfaces susceptibles de produire des interréflexions, comme des écrans omnistéréo ou toute surface concave. Ce chapitre s'inscrit dans le domaine de la vision par ordinateur. La reconstruction active retrouve la forme d'objets à l'aide d'une caméra qui observe des motifs lumineux projetés sur ces objets. Nous verrons que les interréflexions affectent grandement l'une des étapes de la reconstruction active, soit la mise en correspondance caméra-projecteur que nous utilisons dans ce travail pour faciliter la mise sur pied d'un système multi-projecteur. L'utilisation de plusieurs projecteurs pour la projection d'une image unique permet de réaliser une projection haute résolution à un moindre coût. Nous introduirons de nouveaux motifs lumineux spécifiquement conçus pour réduire les interréflexions, tout en restant robustes aux défis standard dans les systèmes de reconstruction active tels que les discontinuités de profondeur du point de vue de la caméra. Il s'agit de la première méthode capable de retrouver la forme d'objets susceptibles de produire des interréflexions, même s'il y a présence de discontinuités de profondeur.

La troisième partie traite de la perception de l'omnistéréo à travers les chapitres 6 et 7. Au chapitre 6, nous donnerons les bases de la vision binoculaire humaine et un aperçu des distorsions de profondeur produites par les images omnistéréo. Nous présenterons au chapitre 7 une analyse des distorsions stéréo introduites par l'omnistéréo à partir de deux modèles existants de projection omnistéréo. Nous

montrerons que les distorsions stéréo augmentent progressivement à la périphérie du champ visuel. Nous établirons le lien entre ces distorsions et les limites connues de la vision stéréo humaine. À notre connaissance, il s'agit de la première tentative d'établissement d'un tel lien. Nous verrons que les distorsions stéréo introduites par ces modèles sont négligeables pour un spectateur au centre d'un écran cylindrique.

Cette thèse se terminera par une conclusion et une discussion sur les avenues de recherches futures.

Publications

Nous présentons ici la liste des publications (articles de journaux et conférences) et les coauteurs des articles présentés dans ce travail.

Journaux

V. Couture, M.S. Langer, S. Roy. *Panoramic stereo video textures*. International Journal of Computer Vision (IJCV), Springer, soumis en octobre 2011.

V. Couture, N. Martin, S. Roy. *Unstructured Light Scanning Robust to Indirect Illumination and Depth Discontinuities*. International Journal of Computer Vision (IJCV), Springer, soumis en décembre 2011.

V. Couture, M.S. Langer, S. Roy. *Analysis of Disparity Distortions in Omnistereoscopic Displays*. ACM Transactions on Applied Perception, vol. 7, n^o 4 (2010), présenté au Symposium on Applied Perception in Graphics and Visualization (APGV 2010).

Conférences

V. Couture, M.S. Langer, S. Roy. *Panoramic stereo video textures.* IEEE International Conference on Computer Vision (ICCV), Barcelone (Espagne), novembre 2011.

V. Couture, N. Martin, S. Roy. *Unstructured Light Scanning to Overcome Inter-reflections.* IEEE International Conference on Computer Vision (ICCV), Barcelone (Espagne), Novembre 2011.

V. Couture, M.S. Langer, S. Roy. *Capturing Non-Periodic Omnistereo Motions.* 10th Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras (OMNIVIS), Zaragoza (Espagne), juin 2010.

Coauteurs

Les coauteurs des articles sont :

Sébastien Roy, directeur de cette thèse et directeur du laboratoire Vision3D du Département d'informatique et de recherche opérationnelle de l'Université de Montréal.

Michael S. Langer, professeur agrégé de l'Université McGill (School of Computer Science) et membre du CIM (Center for Intelligent Machines).

Nicolas Martin, étudiant au doctorat et membre du laboratoire Vision3D.

PARTIE I

TOURNAGE OMNISTÉRÉO

Chapitre 1

VISION PAR ORDINATEUR APPLIQUÉE AUX IMAGES OMNISTÉRÉO

Nous avons mentionné en introduction qu'une image omnistéréo vise à permettre une perception des profondeurs tout autour des spectateurs. Nous avons vu aussi qu'un tournage omnistéréo requerrait un grand nombre de caméras. Nous préférons utiliser une seule caméra stéréo, c'est-à-dire deux caméras standard placées côte à côte, et des techniques de vision par ordinateur pour assembler des images prises en des temps différents. Cependant, un tel assemblage doit aligner ces images et assurer des mouvements continus et cohérents en omnistéréo. Pour mieux comprendre cette problématique, nous discutons à la section 1.1 du type de caméra stéréo nécessaire au tournage. Puis, à la section 1.2, nous décrivons deux méthodes existantes de création de vidéos panoramiques monoculaires par mosaïque, et nous présentons les défis supplémentaires qu'ajoute l'omnistéréo. À la section 1.3, nous modélisons l'un de ces défis, le phénomène de la parallaxe de mouvement. Finalement, nous présentons à la section 1.4 des notions de vision par ordinateur utiles à l'alignement (ou recalage) des images.

1.1 Vers un tournage de vidéos omnistéréo

Cette section considère d'abord deux caméras stéréo dont l'une, munie de deux *fisheyes* (c'est-à-dire deux lentilles très grand angle), met l'accent sur la capture de mouvements panoramiques et l'autre, munie de deux *fentes*, met l'accent sur l'omnistéréo statique. En pratique, une fente peut être considérée comme une colonne de pixels dans une image prise par une caméra munie d'une lentille standard. Puis,

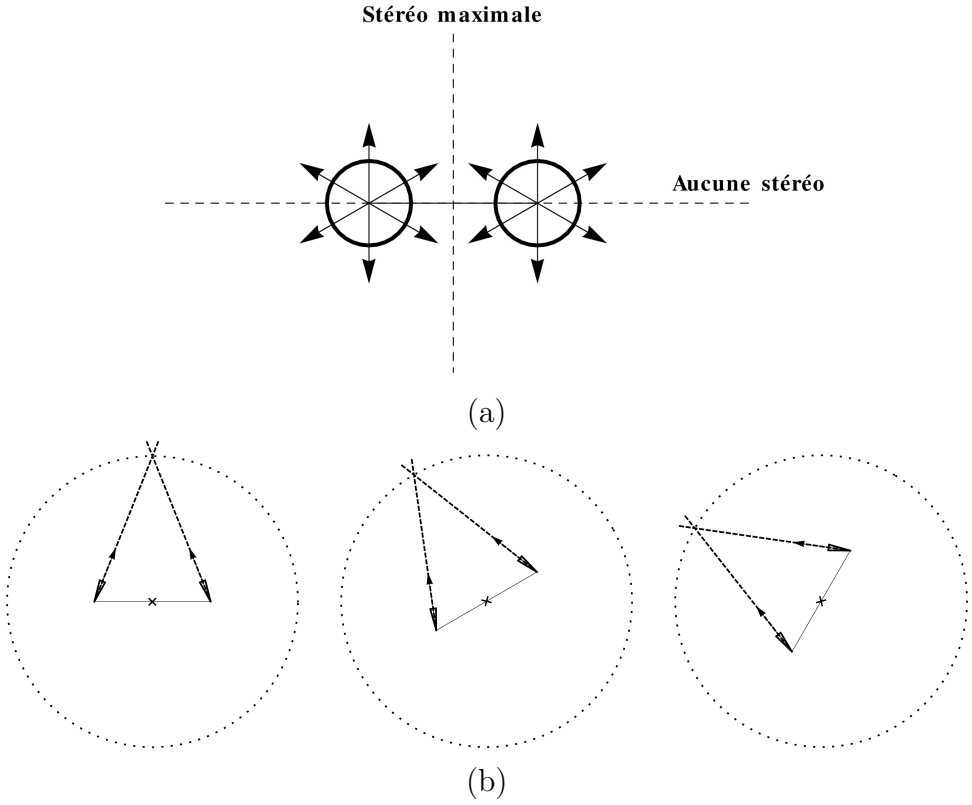


Figure 1.1. (a) Deux caméras très grand angle capturent les mouvements tout autour. Cette configuration permet une perception stéréo maximale des points situés sur la ligne médiane, mais ne permet aucune perception stéréo des points situés sur la ligne qui relie les deux caméras. (b) Deux caméras-fente couvrent graduellement 360° en tournant autour d'un axe (indiqué par un X) pour permettre une perception stéréo tout autour des spectateurs. Seuls les mouvements sur le cercle pointillé sont capturés en même temps par les deux caméras.

nous présentons notre caméra stéréo capable à la fois de capturer des mouvements et de permettre l'omnistéréo.

Soit deux caméras placées côte à côte, munie chacune d'une lentille fisheye qui permet de voir 360° de façon simultanée. Ces caméras capturent, de deux points de vue, le mouvement dans toutes les directions. Cependant, elles ne permettent pas une capture directe de l'omnistéréo. Plus précisément, ces deux caméras ne peuvent capturer en stéréo les points situés sur la ligne qui les relie (voir figure 1.1(a)), parce qu'elles sont placées l'une devant l'autre et non l'une à côté de l'autre, comme nos yeux.

Soit deux caméras ayant chacune un très petit angle de vue horizontal, appelé fente [34, 37, 45, 49]. Le principe des caméras-fente, qui ne capturent qu'une partie étroite de la scène devant elles, est basé sur l'observation que la perception stéréo est maximale lorsque le regard de l'observateur fait face à la scène. La capture 360° se fait graduellement en faisant tourner ces caméras-fente autour d'un seul axe (voir figure 1.1(b)). Bien que cette méthode à fentes fonctionne pour des scènes statiques, elle s'applique mal aux scènes dynamiques. Les deux fentes capturent chaque point visible de la scène à un certain temps, mais il n'y a aucune garantie qu'elles capturent chaque point *en même temps*. À la figure 1.1(b), seuls les mouvements sur le cercle pointillé sont capturés de façon simultanée par les deux caméras.

Au chapitre 2, nous utilisons deux caméras ayant un angle de vue θ entre celui des fisheyes et celui des fentes. Si l'on compare aux fentes, un angle de vue de $\theta = 60°$, par exemple, permet de capturer plus de mouvements de façon simultanée, mais il diminue la perception des profondeurs. En effet, la capture stéréo reste maximale vers le centre du champ visuel, mais elle diminue d'un facteur de $d = \cos(\frac{\theta}{2}) = 0.866$ à la périphérie où la séparation latérale des caméras n'est plus perpendiculaire aux points de la scène. Il est à noter que les fisheyes ($\theta = 180°$ vers l'avant et l'arrière) ont un facteur $d = 0$, ce qui confirme qu'aucune stéréo n'est possible sur les côtés, et

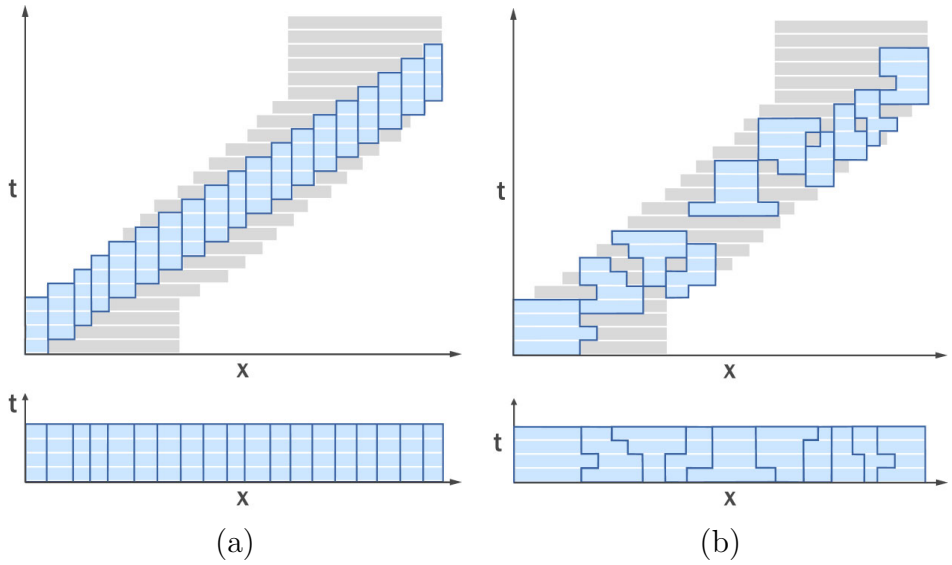


Figure 1.2. Les méthodes existantes créent une vidéo panoramique en assemblant des colonnes (appelées fentes) ou des blocs de pixels. Les figures montrent une coupe x-t d'un volume vidéo XYT tourné par une caméra en rotation vers la droite. Les images originales sont indiquées par des rectangles gris. En (a), une approche simple [49] crée des mosaïques dynamiques à partir d'une colonne dans chaque image. Cette approche crée une vidéo qui cisaille inutilement le mouvement à travers le temps. En (b), une méthode [4] assemble des blocs pour créer une vidéo panoramique cohérente. Tirée de [4].

que les fentes ($\theta \approx 0°$) ont un facteur $d = 1$, ce qui correspond à une stéréo toujours maximale.

En faisant tourner notre caméra stéréo autour d'un seul axe de façon à couvrir 360°, nous capturons des images en des temps différents qui doivent être assemblées en une vidéo cohérente. La section qui suit présente deux méthodes existantes qui s'intéressent au problème d'assemblage des images.

1.2 Assemblage d'images en vidéo panoramique

Dans cette section, nous résumons deux méthodes existantes pour créer des vidéos panoramiques monoculaires à partir d'images capturées par une caméra en rotation

sur un trépied. L'alignement de ces images, les unes par rapport aux autres, forme un volume espace-temps, dont une coupe x-t est montrée en gris à la figure 1.2(a-b). Nous verrons en détail au chapitre 2 la façon dont nous procédons à cet alignement.

Les deux méthodes réorganisent des fentes [57] ou de petits blocs [4] sélectionnés à partir du volume espace-temps pour créer une vidéo panoramique. Plus particulièrement, la méthode de [57] illustrée à la figure 1.2(a) assemble une vidéo panoramique en alignant des colonnes de pixels (ou fentes). Une optimisation peut aussi faire varier la forme des colonnes de façon à réduire les coupures de mouvement. Cette optimisation complexe requiert de faire évoluer, pour chaque image de la vidéo résultante, une coupe 3D du volume espace-temps qui minimise les différences de mouvement. La méthode de [4] sélectionne plutôt des petits blocs de pixels potentiellement disjoints. Une optimisation est nécessaire pour trouver la position temporelle et la forme de ces blocs dans le volume espace-temps. Une optimisation est aussi nécessaire pour minimiser les différences d'intensité entre les blocs tout en maintenant les gradients (les changements d'intensité) à l'intérieur de ceux-ci [50]. Les auteurs ont montré que cette méthode est efficace pour des panoramas dynamiques qui contiennent des vagues sur un lac ou un drapeau au vent.

Mais peu importe si l'on assemble des fentes ou des petits blocs, la création de vidéos omnistéréo exige une cohérence de mouvement en *stéréo*, c'est-à-dire entre les images de gauche et de droite. Cette cohérence est cruciale puisque, comme nous le verrons au chapitre 6, tout désalignement de mouvement entre les deux images peut être perçu comme un changement de profondeur dans la scène. Cependant, une telle cohérence n'est pas garantie si l'on assemble les vidéos panoramiques pour l'oeil gauche et l'oeil droit de façon indépendante. L'assemblage des vidéos gauche/droite avec l'ajout d'une contrainte de cohérence du mouvement stéréo augmenterait la complexité de calcul (déjà grande) des différentes optimisations requises.

Nous proposons au chapitre 2 une méthode qui utilise des images plus larges pour assurer une meilleure cohérence stéréo. Cependant, l'alignement des images

(a) (b) (c)

Figure 1.3. Un exemple de parallaxe pour une caméra en mouvement latéral : de (a) à (c), l'édifice se déplace vers la gauche par rapport à la pancarte.

devient plus difficile en raison du phénomène de la parallaxe de mouvement, que nous expliquons et modélisons à la section qui suit.

1.3 Parallaxe de mouvement

La parallaxe de mouvement se produit lorsqu'une caméra fait un mouvement de translation. Elle fait en sorte que deux objets à des profondeurs différentes se déplacent à des vitesses différentes dans l'image. La figure 1.3 montre ce phénomène pour une caméra en translation presque latérale. Lorsqu'une scène contient plusieurs objets à différentes profondeurs, la parallaxe rend l'alignement des images plus difficile. Dans cette section, nous modélisons la parallaxe pour une disposition spécifique des caméras. Cette modélisation mènera à une observation clé dans le processus d'alignement des images.

Soit deux caméras standard placées côte à côte sur un trépied. Nous supposons que la distance entre le centre des caméras est de 6.5cm, comme la distance entre les yeux [33]. Nous supposons également que ces caméras tournent autour d'un axe unique et que la ligne transversale reliant leur centre de projection (*i.e.* leur centre optique) passe par cet axe (voir fig. 1.1(b)). Chaque caméra suit donc un mouvement circulaire qui contient des composantes translationnelle et rotationnelle. Nous considérons deux points, le premier à une profondeur de 2m et le deuxième à l'infini, qui entrent en

même temps dans le champ de vision de la caméra gauche ou droite. La figure 1.4 montre l'évolution de la parallaxe pour une caméra ayant un angle de vue de 30° (courbe rouge), 60° (courbe verte) et 90° (courbe bleue). Les mesures de la parallaxe sont montrées pour la caméra droite seulement, mais les résultats sont similaires pour la caméra gauche (voir l'annexe A pour le calcul détaillé de la parallaxe). L'entrée des deux points dans le champ de vision correspond à l'angle 0° sur l'abscisse. À mesure que la caméra continue sa rotation, la parallaxe fait en sorte que ces deux points se déplacent à des positions différentes dans l'image. La figure 1.4(a) montre que cette parallaxe peut atteindre de deux à trois pixels lorsque ces points avoisinent le centre de l'image (à environ 15°, 30° et 45° sur l'abscisse). Il est à noter que la parallaxe en pixels dépend de la résolution des caméras, de leur angle de vue, et qu'elle peut s'accentuer si la profondeur du premier point est en deçà de 2m. Cependant, elle redevient nulle lorsque ces deux points sortent du champ de vision (à 30°, 60° et 90° sur l'abscisse), ce qui mène à l'observation clé suivante, tirée de nos travaux publiés dans [15].

Observation clé: *Pour une scène statique, si deux points entrent en même temps dans le champ de vision d'une caméra, ils en sortiront en même temps.*

La figure 1.4(a) montre que cette observation clé est valable pour les trois angles de vue. Elle ne tient plus si les caméras ne sont pas parallèles, par exemple pour une convergence de 10° des caméras (voir fig. 1.4(b)).

Nous utiliserons au chapitre 2 cette observation pour faciliter l'alignement de deux images, l'une où deux points entrent dans le champ visuel des caméras et l'autre où ces deux mêmes points en sortent. Selon l'observation clé, la parallaxe entre ces deux points est nulle dans les deux images.

Le mouvement circulaire des caméras produit aussi de la parallaxe verticale. La figure 1.5 montre que deux points qui entrent dans le champ visuel au coin supérieur gauche de l'image (0° sur l'abscisse) ne se projettent plus à la même position verticale

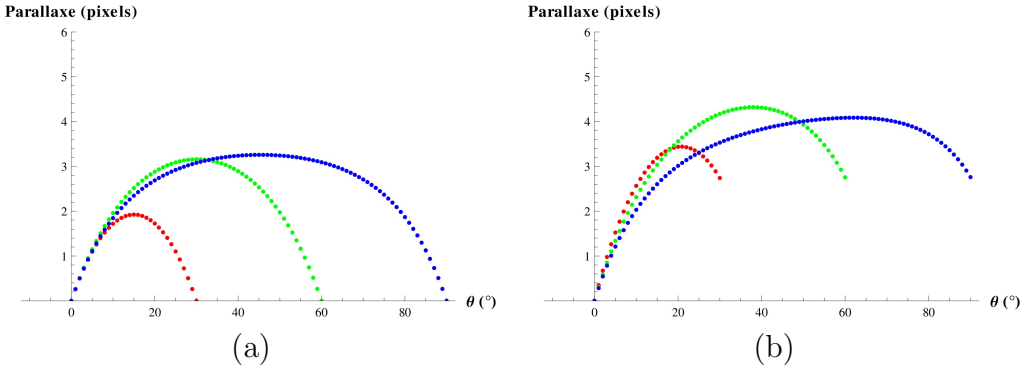


Figure 1.4. Parallaxe horizontale pour la caméra droite HD (1920×1080 pixels) ayant un angle de vue de $30°$ (rouge), $60°$ (vert) et $90°$ (bleu) pour deux points entrant en même temps dans son champ de vision ($0°$ sur l'abscisse). Nous supposons les caméras droite et gauche parallèles (a) ou ayant une convergence de $10°$ en (b).

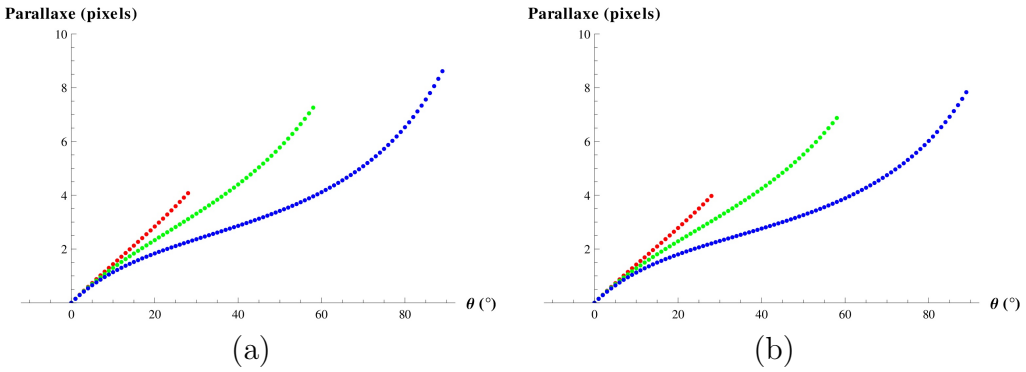


Figure 1.5. Parallaxe verticale pour la caméra droite HD (1920×1080 pixels) ayant un angle de vue de $30°$ (rouge), $60°$ (vert) et $90°$ (bleu) pour deux points entrant en même temps dans son champ de vision ($0°$ sur l'abscisse). Nous supposons les caméras droite et gauche parallèles (a) ou ayant une convergence de $10°$ en (b).

lors de leur sortie au coin supérieur droit (à 30°, 60° et 90° sur l'abscisse), ce qui donne une parallaxe verticale maximale de 4 à 9 pixels. Cette parallaxe verticale est maximale en haut et en bas de l'image, mais elle est nulle au centre vertical du champ visuel puisque les points sont dans le plan défini par la trajectoire circulaire des caméras.

Nous avons mentionné plus haut que l'observation clé est valide seulement si les caméras sont parallèles et si la ligne transversale qui relie leur centre de projection passe par l'axe de rotation. En pratique, nous plaçons les caméras en parallèle manuellement et ajustons leur position vers la gauche ou la droite de façon à ce que le plan médian (*i.e.* le plan qui sépare les deux caméras) passe par l'axe de rotation. Cependant, il n'y a aucun moyen de savoir à quelle distance de l'avant des caméras passe la ligne transversale. Si les caméras sont trop avancées ou reculées, l'observation clé ne tient plus, comme à la figure 1.4(b). Pour s'assurer d'un bon alignement, nous affichons l'image d'une des deux caméras sur un moniteur HD externe pour pouvoir observer le comportement de la parallaxe, et avançons ou reculons les caméras jusqu'à ce que la parallaxe suive l'orientation clé.

1.4 Notions de vision par ordinateur utiles à l'alignement des images

L'alignement des images, décrit en détail à la section 2.7, nécessite un calibrage précis des caméras. Le calibrage estime la position et le modèle de lentille d'une caméra. Nous expliquons ici les notions de vision par ordinateur requises pour procéder à un autocalibrage (en anglais, *auto-calibration* ou *self-calibration*), c'est-à-dire un calibrage à partir des images de la scène seulement, sans y inclure des quadrillés ou autre forme régulière [29]. Nous décrivons d'abord le modèle de caméra et celui de la distorsion radiale. Puis, nous présentons deux méthodes de détection de points saillants dans des images, SIFT et KLT, dont la première permet également d'établir des correspondances entre eux et la deuxième, leur suivi. En dernier lieu, nous

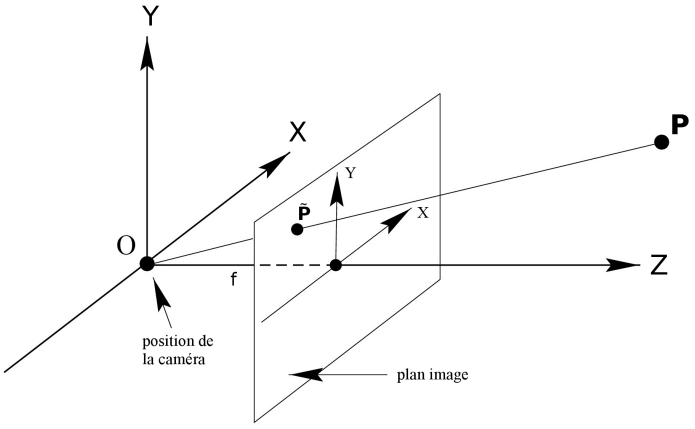


Figure 1.6. Géométrie d'une projection perspective.

détaillons l'estimation de tous les paramètres des modèles par ajustement de faisceaux à l'aide de la méthode RANSAC, laquelle élimine les valeurs aberrantes.

1.4.1 Modèle de caméra

Un modèle de caméra définit le processus de formation des images, c'est-à-dire le passage du monde 3D (la scène) au plan image 2D, et le passage du plan image 2D aux coordonnées pixels de l'image. Le modèle d'une caméra perspective (voir fig. 1.6) basé sur le sténopé[1] projette un point P, aux coordonnées $(X, Y, Z)^T$ dans le monde, à la position $\tilde{p} = (\tilde{x}, \tilde{y})^T$ dans le plan image de la caméra :

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{pmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{R}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{T}}}_{\text{projection perspective}} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{1.1}$$

[1] Un sténopé (*pinhole*) est un appareil photographique dont l'objectif est un minuscule trou.

où les matrices $\boldsymbol{T}$ et $\boldsymbol{R}$ dépendent respectivement de la position et de l'orientation de la caméra. À noter que l'équation 1.1 utilise la représentation projective des points, qui ajoute une dimension à l'espace euclidien (*i.e.* un 1 est ajouté aux coordonnées de P et $\tilde{p}$). Deux points projectifs à un facteur d'échelle près sont considérés équivalents :

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{pmatrix} \equiv \begin{pmatrix} w\tilde{x} \\ w\tilde{y} \\ w \end{pmatrix} \quad \forall w \neq 0.$$

Il convient donc de diviser le résultat de l'équation 1.1 par sa troisième coordonnée, ce qui revient à diviser un point par sa profondeur $\tilde{z} = w$ dans l'espace caméra. L'espace projectif permet, entre autres, la représentation d'un point à l'infini ($w = 0$) et la combinaison d'une translation à une transformation affine (une rotation, par exemple), qui sont alors toutes deux appliquées par la multiplication d'une matrice.

La conversion en coordonnées pixels $p = (x, y)^T$ se fait ensuite par une transformation $\boldsymbol{K}$:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \underbrace{\begin{bmatrix} f & 0 & o_x \\ 0 & f & o_y \\ 0 & 0 & 1 \end{bmatrix}}_{\mathrm{K}} \begin{pmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{pmatrix} \tag{1.2}$$

où la distance focale f dépend de l'angle de vue de la caméra (et inclut de façon implicite la taille des pixels de la matrice CCD ou CMOS), et $(o_x,\ o_y)$ représente l'intersection de l'axe optique avec le plan image. Pour une image de taille $L \times H$, $(o_x,\ o_y)$ est habituellement très proche du centre de l'image $(\frac{L}{2}, \frac{H}{2})$. La matrice $\boldsymbol{K}$ regroupe ce que l'on appelle les paramètres internes de la caméra, tandis que les matrices $\boldsymbol{R}$ et $\boldsymbol{T}$ constituent ses paramètres externes.

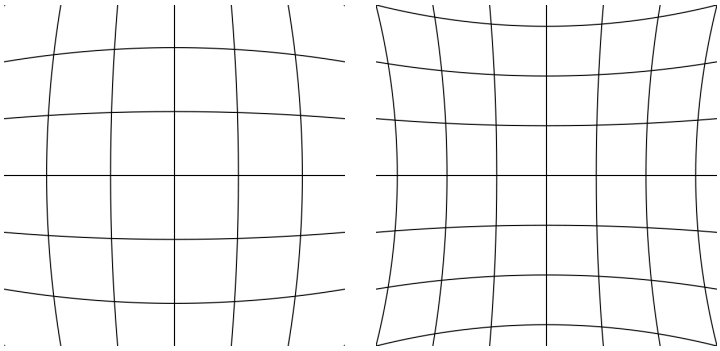


Figure 1.7. Image d'un quadrillé affecté par la distorsion radiale (a) en barillet (b) en coussinet. Dans les deux cas, la distorsion est nulle au centre de l'image.

1.4.2 Distorsion radiale

Le modèle de caméra perspective ne modélise pas les distorsions dues à la lentille de caméra (voir fig. 1.7). La distorsion radiale[2] déplace un point $(\tilde{x}, \tilde{y})^T$ radialement sur le plan image en fonction de son rayon $r = \sqrt{\tilde{x}^2 + \tilde{y}^2}$ par rapport au centre de l'image :

$$\begin{aligned} \tilde{x}' &= \tilde{x}(1 + k_1 r^2 + k_2 r^4) \\ \tilde{y}' &= \tilde{y}(1 + k_1 r^2 + k_2 r^4) \end{aligned}$$

où k_1 et k_2 sont des paramètres à estimer. Le point $(\tilde{x}', \tilde{y}')$ distordu peut ensuite être transformé en coordonnées pixels avec la matrice $\boldsymbol{K}$, tel que décrit précédemment.

[2] Nous ignorons ici la distorsion tangentielle.

1.4.3 SIFT: points saillants invariants à l'échelle

Nous estimons les paramètres de caméra et de distorsion radiale à partir directement des images, plus particulièrement à partir de points correspondants dans deux images ou plus. La méthode SIFT (Scale-Invariant Feature Transform) permet de détecter et d'identifier automatiquement les points saillants d'une image. Les points saillants SIFT se veulent le plus indépendant possible de l'échelle, de l'orientation et de l'exposition. Un point d'intérêt est détecté par la mesure des gradients (les changements d'intensité) de l'image à différentes échelles. Puis, son orientation est déterminée à partir de la direction des gradients dans un voisinage. Un histogramme calculé à partir de cette orientation sert finalement à identifier chaque point saillant par un descripteur (un vecteur de 128 valeurs). Ce vecteur s'avère utile pour la mise en correspondance de deux images, même si l'objet est capturé de deux points de vue différents.

1.4.4 KLT: détection et suivi de points saillants

La méthode KLT (Kanade-Lucas-Tomasi) permet de détecter et de suivre la position de points saillants de manière beaucoup plus rapide que la détection et la mise en correspondance SIFT. Cependant, cette méthode peut être utilisée seulement si le déplacement entre deux images est petit. Dans un premier temps, les coins d'une image sont détectés en examinant localement les gradients de l'image. Puis, leur déplacement est estimé en supposant un mouvement constant dans une petite région de l'image.

1.4.5 RANSAC: estimation robuste d'un modèle

RANSAC (abréviation de RANdom SAmple Consensus) [22] est une méthode itérative d'estimation des paramètres d'un modèle mathématique à partir de données, malgré la présence de valeurs aberrantes (*outliers*). Les valeurs aberrantes sont des

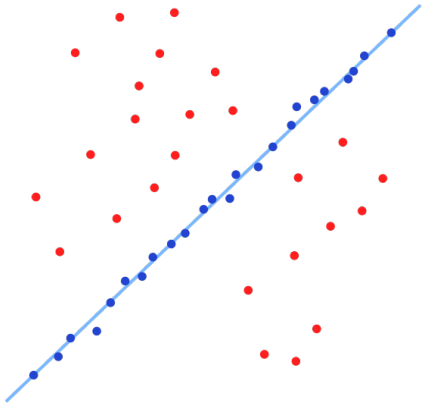


Figure 1.8. Le modèle d'une ligne droite est ajusté avec la méthode RANSAC, malgré plusieurs valeurs aberrantes. Tirée de http://fr.wikipedia.org/wiki/RANSAC.

mesures erronées ou des valeurs extrêmes de bruit. À chaque itération, les paramètres du modèle sont estimés à partir d'un sous-ensemble aléatoire des données. Toutes les autres données sont ensuite testées par rapport à ces paramètres, qui sont considérés corrects si un nombre suffisant de données correspondent au modèle. La figure 1.8 montre le modèle d'une ligne 2D à partir de points.

1.4.6 Ajustement de faisceaux

L'ajustement de faisceaux (*bundle adjustment*) est une optimisation qui tente de retrouver à la fois les paramètres internes et externes de caméras et la forme de la scène. Dans le contexte de l'omnistéréo, il s'agit d'estimer les paramètres internes de deux caméras en configuration stéréo (la distance focale f et les coefficients de distorsion radiale k_1 et k_2) et de leurs paramètres externes (position et orientation) pour chaque image. Nous utilisons l'algorithme de Levenberg-Marquardt [43] pour effectuer l'ajustement de faisceaux. Cet algorithme minimise de façon stable et efficace des fonctions non linéaires à plusieurs variables.

Soit une caméra stéréo qui capture 30 images par seconde durant un tour complet d'environ une minute. Au chapitre 2, nous utilisons la méthode d'ajustement de faisceaux pour optimiser les paramètres de 60 images dans chacune des caméras, soit une image par seconde. Ce nombre assure un chevauchement assez grand entre deux images contiguës, sans nécessiter une trop grande quantité de données à traiter.

Nous avons vu à la section 1.4.1 qu'une caméra perspective projette un point dans le monde vers une image par une matrice de dimensions 3×4, $\boldsymbol{M}_j = \boldsymbol{K}_j \boldsymbol{R}_j \boldsymbol{T}_j$, l'indice j faisant référence à l'une des 120 positions et orientations des caméras. À partir de points saillants SIFT mis en correspondance dans les j images, l'ajustement de faisceaux estime les paramètres de caméra en minimisant l'erreur non-linéaire entre la position des points saillants $p_{ij} = (x_{ij}, y_{ij})^T$ et la reprojection des points $P_i = (X_i, Y_i, Z_i, W_i)^T$ correspondants dans le monde, autrement dit en minimisant :

$$\sum_{i=1}^{N_p} \sum_{j=1}^{N_c} \left[\left(\frac{m_{j1}^T P_i}{m_{j3}^T P_i} - x_{ij} \right)^2 + \left(\frac{m_{j2}^T P_i}{m_{j3}^T P_i} - y_{ij} \right)^2 \right] \tag{1.3}$$

où m_{j1}^T, m_{j2}^T et m_{j3}^T représentent les trois rangées de $\boldsymbol{M}_j$. La double somme de l'équation 1.3 considère les N_p points dans le monde et les $N_c = 120$ images, à condition qu'un point i soit visible dans l'image j. Bien que les facteurs de distorsion radiale k_1 et k_2 sont pris en compte (voir la section 1.4.2), ils ne sont pas indiqués ici pour simplifier la notation.

L'estimation de la position d'un point P_i dans le monde se fait par triangulation à partir d'au moins deux images de caméra. Dans notre cas, la triangulation se fait toujours à partir de deux images gauche/droite capturées en même temps, mais la reprojection se fait dans toutes les images j où le point P_i est visible.

Plus précisément, $P_i = (X_i, Y_i, Z_i, W_i)^T$ a quatre inconnues, et chaque point $p_{ij} = (x_{ij}, y_{ij}, 1)^T$ ajoute deux contraintes dérivées à partir du modèle de projection :

$$p_{ij} = \boldsymbol{M}_j P_i \, . \tag{1.4}$$

En réarrangeant l'équation 1.4, on obtient les deux contraintes :

$$x_{ij} = \frac{m_{j_{11}} X_i + m_{j_{12}} Y_i + m_{j_{13}} Z_i + m_{j_{14}} W_i}{m_{j_{31}} X_i + m_{j_{32}} Y_i + m_{j_{33}} Z_i + m_{j_{34}}}$$
$$y_{ij} = \frac{m_{j_{21}} X_i + m_{j_{22}} Y_i + m_{j_{23}} Z_i + m_{j_{24}} W_i}{m_{j_{31}} X_i + m_{j_{32}} Y_i + m_{j_{33}} Z_i + m_{j_{34}}}$$

où $m_{j_{rc}}$ représente la valeur à la rangée r et la colonne c de la matrice $\boldsymbol{M}_j$. En multipliant par le dénominateur des deux côtés, et en factorisant le point du monde inconnu $P_i = (X_i, Y_i, Z_i, W_i)^T$, on obtient un système linéaire homogène dont la résolution donne P_i.

Cependant, un ajustement par faisceaux requiert une bonne approximation des matrices $\boldsymbol{M}_j$ puisque la minimisation de l'erreur de reprojection est un problème non convexe avec minimums locaux. Pour cette approximation seulement, nous supposons que la translation des caméras est nulle, autrement dit qu'elles sont en rotation pure, et que $\boldsymbol{M}_j = \boldsymbol{K}_j \boldsymbol{R}_j$. Il reste alors à estimer la rotation entre chaque image et la distance focale f (voir la définition de $\boldsymbol{K}$ à la section 1.4.1). Le lien entre les points saillants correspondants de deux images contiguës j et $j+1$ peut alors être modélisé comme une homographie $\boldsymbol{H}_j$:

$$\begin{pmatrix} x_{ij+1} \\ y_{ij+1} \\ 1 \end{pmatrix} = \boldsymbol{H}_j \begin{pmatrix} x_{ij} \\ y_{ij} \\ 1 \end{pmatrix}$$

où $\boldsymbol{H}_j = \boldsymbol{K}_{j+1} \boldsymbol{R}_{j+1} \boldsymbol{R}_j^{-1} \boldsymbol{K}_j^{-1}$ est une matrice 3×3. L'estimation linéaire d'une homographie à partir de deux images nécessite huit points saillants correspondants, que nous détectons à l'aide de la méthode SIFT. Habituellement, le nombre de points correspondants dépasse largement huit et une méthode RANSAC rend l'estimation robuste aux erreurs de correspondance.

L'axe et l'angle de rotation sont donnés respectivement par les vecteurs et les valeurs propres de $\boldsymbol{H}_j$. Une fois la rotation estimée, on peut retrouver f à partir d'un système d'équations linéaires basé sur les homographies, qui peuvent être paramétrisées comme suit :

$$\boldsymbol{H}_j = \begin{bmatrix} r_{j11} & r_{j12} & f r_{j13} \\ r_{j21} & r_{j22} & f r_{j23} \\ \frac{r_{j31}}{f} & \frac{r_{j32}}{f} & r_{j33} \end{bmatrix}.$$

où les coefficients $r_{j_{rc}}$ correspondent à la matrice de rotation entre l'image j et $j+1$.

Finalement, il est à noter que nous utilisons le protocole LANC pour synchroniser en stéréo la capture d'images et le zoom. Ainsi, nous pourrions supposer que les angles de vue et de rotation sont les mêmes pour les deux caméras. Cependant, cette supposition peut être légèrement erronée en raison d'imprécisions au niveau de la fabrication des caméras ou du protocole LANC. Typiquement, la capture par les deux caméras peut être désynchronisée de trois millisecondes ou moins. Ainsi, nous modéliserons, au chapitre 2, un angle de vue différent pour chaque caméra, mais cet angle restera constant pour toute la séquence. Nous modéliserons aussi un léger délai de capture pour une des deux caméras.

Chapitre 2

PANORAMIC STEREO VIDEO TEXTURES (ARTICLE)

Ce chapitre présente l'article suivant :

> V. Couture, M.S. Langer, S. Roy. *Panoramic stereo video textures*. International Journal of Computer Vision (IJCV), Springer, soumis en octobre 2011.

L'article présente notre méthode de création de vidéos omnistéréo par assemblage d'images, qui s'est inspirée de méthodes existantes pour la création de vidéos panoramiques monoculaires. Comme ces méthodes, la nôtre suppose que les mouvements dans la scène sont stochastiques et localisés, mais elle en diffère de deux manières fondamentales. Premièrement, notre méthode utilise des images vidéo complètes au lieu de fentes ou de petits blocs de pixels, ce qui réduit le problème de synchronisation du mouvement en stéréo. Deuxièmement, elle utilise un simple dégradé entre les régions voisines au lieu de minimisations complexes ou d'un lissage des gradients.

Nous présentons également une expérience psychophysique qui étudie la visibilité des dégradés dans différentes scènes. Cette expérience a nécessité le tournage stéréo de plusieurs scènes afin de créer des stimuli avec dégradés, lesquels ont servi au montage d'un test visuel. Ce test visait à mesurer le temps de détection des dégradés. Les résultats de cette expérience permettent de conclure à l'efficacité de la méthode pour des scènes sans objets saillants, comme des courants d'eau. Cependant, des duplications sont visibles lorsque les objets en mouvement sont bien définis, comme des branches. Les résultats des participants qui n'ont pas réussi le test de vision stéréo sont montrés à l'annexe B.

Nous présentons ici l'article dans sa version originale.

Abstract

A panoramic stereo (or omnistereo) pair of images provides depth information from stereo up to 360 degrees around a central observer. Because omnistereo lenses or mirrors do not yet exist, synthesizing omnistereo images requires multiple stereo camera positions and baseline orientations. Recent omnistereo methods stitch together many small field of view images called slits which are captured by one or two cameras following a circular motion. However, these methods produce omnistereo images for static scenes only. The situation is much more challenging for dynamic scenes since stitching would need to occur over both space and time and should synchronize the motion between left and right views as much as possible. This paper presents the first ever method for synthesizing panoramic stereo video textures. The method uses full frames rather than slits and it uses blending across seams rather than more complex stitching. The method produces loopable panoramic stereo videos that can be displayed up to 360 degrees around a viewer. We also present results of a perceptual experiment that evaluates our approach by asking naive observers to locate a blended region for different types of scenes. The results are consistent with our expectations about the strengths and limitations of our method, namely that the blending method is more effective when the motion is texture-like than when the motion consists of isolated moving objects.

2.1 Introduction

Traditional stereo (3D) cinema uses two cameras with heavily overlapping field of views which capture two videos of a scene from slightly different viewpoints When the stereo pair is displayed to a human viewer, one video to each eye, the videos are fused and the disparities provide strong cues to scene depth, thereby enhancing the immersion experience. Well-known issues arise at the boundaries of the fields of view, namely 3D points that are rendered to be in front of the screen must not cut across a

frame boundary, since this leads to incorrect occlusion cues which are known as stereo window violations [44]. Correctly avoiding such window violations is one of the key technical challenges which must be addressed for stereo cinema to be successful.

One way to reduce stereo window violations is to capture and display images with a very wide field of view, for example, a panorama. Many methods have been developed for synthesizing panoramas. For static scenes, these methods are now a standard part of the software toolkit of basic consumer level digital cameras [74]. However, these tools are for monocular images, not stereo images. Making stereo panoramas (also known as *omnistereo* [1]) remains very challenging, even for static scenes. The problem of making panoramas with stereo and motion has, until recently, been so challenging that it has not been addressed at all.

This paper addresses the more challenging problem of capturing stereo video over a much wider field of view, up to 360 degrees, and synthesizing the videos into a stereo panorama. One application of such omnistereo videos is for display screens with a very wide field of view. In the extreme case of a 360 degree cylindrical screen, observers would be be able to turn their gaze in any orientation and there could be more than one observer present, with different observers looking in different directions at the same time. A second and more every day application of a 360 degree stereo video panorama would be to use a standard display such as a stereo computer monitor, and to allow the user to pan over the 360 degree view. An example would be a stereo-video extension of Google Street View. Here we have in mind a case in which the motion is a texture such as waves on a river or lake, trees blowing in the wind, or a flag waving.

To capture stereo video in a wide range of directions, one could extend multi-camera systems. For example, the commercially available Ladybug [52] has

[1] In the literature, the terms *omnistereo* and *stereo panorama* sometimes refer to different imaging situations, the former being a full 360 degrees and the latter being possibly less than 360 degrees. We use the terms interchangeably in this paper. Note that the method that we present is described for the 360 degrees capture situation, but it could also be applied to less than 360 degrees.

five cameras to cover 360 degrees. One could extend such systems to stereo by doubling the number of cameras. Alternatively, one could attempt to combine previous computational approaches for static omnistereo and dynamic panoramas. We argue in Section 2.2, however, that one faces fundamental difficulties in doing so, related to stereo-motion synchronization. This leads us to take a different approach.

The approach we introduce uses a conventional stereo video rig where the two cameras each follow a circular motion and capture a space-time volume of images. We combine the full frames of these videos into left and right panoramic video textures. The method is most effective with localized stochastic motions such as leaves moving in the wind or waves on a lake. Because our method uses full frames, most of the scene is viewed by both cameras simultaneously which guarantees stereo motion synchronization for these points. The method produces omnistereo video textures that are several seconds long and are loopable in time [62].

Our method was presented in [16]. The contribution of the present paper is to present a formal perceptual experiment that examines the conditions in which our method works well or less well. The experiment measures psychophysical thresholds of how fast human observers can locate blended regions for various types of dynamic scenes. We will discuss the strengths and limitations of our method based on the results of this study.

The paper is organized as follows. Section 2.2 gives a brief overview of previous work on panoramic stereo and video panoramas. Section 2.3 gives the main details of our approach and outlines key differences from previous approaches. Section 2.4 presents example results using our method. Section 2.5 presents our perceptual experiment. We conclude in Section 2.6.

2.2 Previous Work

Existing computer vision methods for generating static stereo panoramas typically gather several small horizontal field of view images, namely vertical *slits*, from cameras rotating off-axis and then make a mosaic of these slits [34, 37, 45, 49]. The idea is that each point in the world projects into some slit in the left camera and some slit in the right camera and the disparity between the corresponding slits depends on the depth of the point. The slits typically cover one or two degrees, so 200-400 slits are used to capture 360 degrees. Figure 2.1 illustrates an omnistereo system that uses two cameras. An alternative omnistereo configuration uses a single camera with two slits, one to the left of center and one to the right [48]. This corresponds to two virtual cameras following a circle.

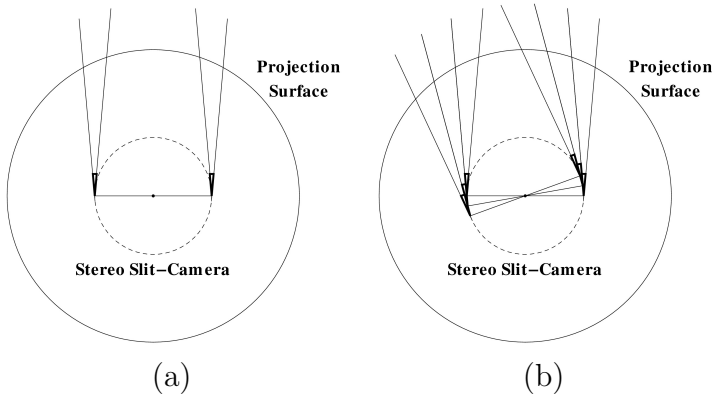


Figure 2.1. An omnistereo method that uses a rotating stereo pair of parallel slit-cameras. (a) a pair of slits (b) slits of each camera are stitched together into a mosaic that covers 360° (only three slits are shown).

The reason for using slits is that, when the stereo rig is rotated, each camera rotates but also translates. Because of the translation there is parallax between the images captured by each camera. If one were to stitch together a small number of images with large horizontal fields of view (i.e. not slits), then the parallax would produce visible seams at the image boundaries. This parallax problem is also a

problem in standard monocular panoramas, but one can solve it in the monocular case by using a tripod mount that allows one to rotate the camera about its nodal point [74]. In the case of stereo panoramas, however, there is no corresponding solution since cameras must translate as well as rotate in order to provide disparities in all directions around the observer[63].

While these slit-based methods work well for static scenes, they do not generalize well to dynamic scenes. Each camera's slit captures each visible scene point at some time, but there is no guarantee that each scene point will be captured by the two slits *at the same time*. Indeed this simultaneity condition is only met if the scene point happens to have a similar disparity as the slits. This condition might be met for some scene points, but it does not hold in general.

A related set of methods have been invented for monocular panoramas with dynamic scenes. Consider a video as a space-time volume. These methods rearrange either slices [57] or small 3D blocks [4] from the volume, such that one tries to avoid visual seams between the slices or blocks. An example is the *dynamosaicing* method [57] which makes a video mosaic by using graph cuts to compute a time evolving surface in a video's space-time volume. The surfaces are then stitched together to yield a video. A second graph cut based approach [4] is *panoramic video textures*. This method renders a video seen by a rotating camera. Rather than selecting and stitching together slices in the space-time volume, it selects and stitches small space-time blocks. In addition to using graph cut matching to make the seams less visible, it also uses gradient smoothing. This method has been shown to be effective for dynamic panoramas that contain waves on a lake, a flag in the wind, etc. Such methods have been used successfully to generate panoramas from videos taken by a (purely) rotating camera.

It is not clear if one could generalize these monocular panoramic video methods to stereo. Regardless of whether one stitches together surfaces in XYT or small blocks, the key problem remains of how to synchronize the left and right views at the

stitch points. If one were to compute monocular panoramic video textures for the left and right eye independently using the above methods, there is no reason why the resulting panoramas would be synchronized, that is, there is no reason why the same scene events (*e.g.* a leaf blowing to the right) would appear at the same time in the left and right views and would have the correct disparity. One might try to extend the panoramic video methods to stereo by enforcing a constraint on stereo motion consistency, but such an extension is not obvious and would significantly increase the (already large) computational complexity of these methods.

The approach that we take is much simpler, and differs from previous approaches in two fundamental ways. First, we use full video frames rather than slits or small blocks. Using full frames reduces the stereo-motion synchronization problem, which arises only near the boundaries of the regions being stitched together. That is, using full frames reduces the percentage of pixels that lie near the boundaries. The second difference is that, rather than using graph cut based stitching or gradient smoothing to reduce the visual seam boundaries between regions, our method simply blends neighbouring regions together.

We have found that blending is sufficient in many cases, namely the blended regions are not visually salient in practice for motions that are stochastic and localized, such as water flows or leaves in the wind. Although the blending is visible in some cases if one scrutinizes the video, it is typically not visible in casual viewing. The perceptual experiment described in Sec. 2.5 further investigates how visible the blended regions are for different scenes.

2.3 Our Method

The panoramic stereo video problem begins with a stereo video pair which has been captured by a stereo rig rotating around a vertical axis. Each camera follows a circular path similar to Figure 2.1. In each of the examples we present in Section

2.4, we capture a full 360 degrees. The videos are about 2 minutes each, *i.e.* a few thousand stereo frames.

In this section we present the details of our method. In Sec. 2.3.1, we summarize the camera calibration method and the mapping from camera pixels to the cylindrical projection pixels. In Sec. 2.3.2, we make some basic observations about the paths of points in the left and right space-time volumes which are generated by the rotating stereo rig, for the case of constant rotational velocity. In Sec. 2.3.3 we show how we partition the space-time volumes. In Sec. 2.3.4 we show how we blend the neighboring volumes parts into left and right panoramic videos. In Sec. 2.3.5 we address with the more general case that the camera rotation speed may vary over time.

2.3.1 Camera calibration and video registration

Given the left and right videos, we first calibrate the stereo camera rig, both for the camera internals (focal length) and externals (position and orientation in each frame). This calibration allows us to map pixels in the two cameras in each frame to pixels on a cylindrical projection surface (one for each camera). This yields left and right XYT volumes, composed of frames that shift over time as the cameras move.

As a first approximation, we estimate camera parameters by ignoring camera translation and sub-sampling the frame sequence in time. We compute SIFT features in these frames and compute homographies between frames using RANSAC for robustness. We then estimate camera parameters (rotation, focal length) and perform bundle adjustment, taking radial distortions into account [29]. Next we improve and complete these estimates by considering all the frames. We track features between frames, allowing for small camera translation, and perform another bundle adjustment that triangulates features in 3D [66]. All of these steps use standard computer vision techniques. Further details are given in the Appendix.

2.3.2 Motion paths and parallax

As the stereo rig rotates, the projection of a fixed point in the scene moves across the image. This image motion is a combination of the motion of the scene point and the camera's rotation and translation. To understand this image motion, first consider two *static* scene points $P_{Z_{min}}$ and P_∞ that enter the field of view at the same time and the same position, namely at one edge of the frame. See Figure 2.2. As the rig rotates, motion parallax occurs and the x-pixel positions of these two points diverge slightly as they cross the frame. The x positions converge and meet again when the points leave the field of view at the opposite edge, namely when the camera center again lies on the line connecting the two points. In practice, the separation that is due to this motion parallax is maximum at the center of the frame and is a few pixels only [15]. This parallax magnitude is comparable to that seen by a person who translates his head slightly, as in normal posture adjustment.

The space-time paths of various points are sketched in Figure 2.3. For the sake of clarity, we are assuming here that the stereo rig rotates at constant angular velocity. This would be the case if the camera motion were driven by a motor, for example. With uniform rotation, the frame boundaries of the calibrated space-time volume define two diagonal planes of constant x-slope, namely the diagonal lines in Figure 2.3. In Sec. 2.3.5, we return to the general case that the camera rotation speed can vary over time.

Next consider the space-time paths that represent projections of scene points. The dashed vertical line traces the constant visual direction of the point at infinity P_∞. The red curve traces the changing visual direction of the point $P_{Z_{min}}$ which is at a finite distance. The curvature is greatly exaggerated in this figure, since baseline of the camera is much smaller than the distance to visible points in the scene [15].

In addition to the horizontal parallax (x) just discussed, there can also be a small amount of vertical parallax (y) which arises from the forward camera translation

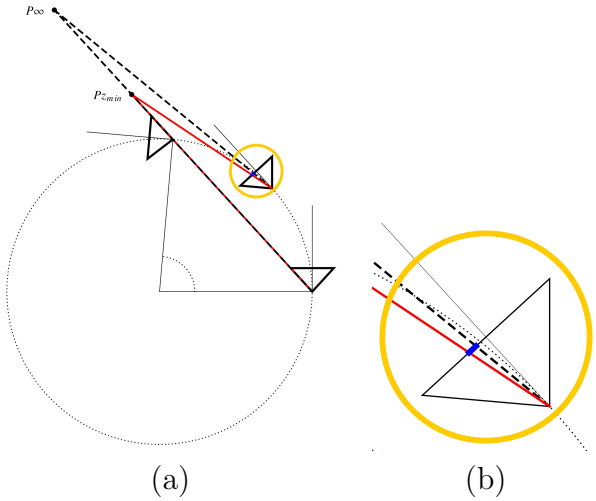


Figure 2.2. The right camera only is shown. Assume the camera rig rotates clockwise. (a) Two points $P_{Z_{min}}$ and P_∞ are shown that enter the camera frame at the right edge at the same time (top). These points also exit the frame at the same time (right). As the camera moves, the positions of the points drift across the frames. The depth difference of the two points leads to motion parallax (blue line). See expansion of the yellow circle in (b). The figure is not to scale, namely the camera baseline is typically much smaller than the distance to scene points and so parallax is typically very small. In this example, the camera field of view is 90°, but the argument about coincidence at the left and right edge holds for any field of view size.

component. For example, a point at infinity that enters the field of view at the top right corner of a frame will leave the frame at the top left corner, but if a point that is a finite distance away were to enter at the top right corner at the same time then it would leave the frame earlier, namely at the vertical edge (before it reaches the top left corner). In general, this vertical parallax is zero for points on the horizontal mid-line and increases to a few pixels toward the upper and lower corners.[2]

A few other observations about motion paths should be made. First, our arguments above follow from geometry illustrated in Figure 2.2 and do *not* depend

[2] The amount of horizontal and vertical parallax depends on camera resolution, field of view and the range of scene depths. For HD cameras having a 60 degree field of view and scene depths ranging for 2m to infinity, the maximum parallax is about 5 pixels wide and 7 pixels high.

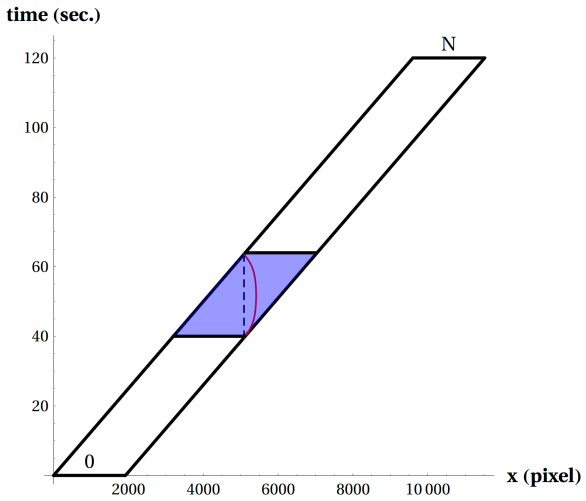


Figure 2.3. A sequence of N frames captured by the left or right camera performing a full turnaround, after calibration and registration. For simplicity, we assume in this figure that the rig rotates at constant speed. The dashed and red lines represent the path followed by two static points, one far and one close, respectively (see Fig. 2.2), that enter and exit the field of view at the same time. The two thick black lines represent the entry and exit frames [15].

on the assumption that the camera rotation speed is uniform. In particular, the paths of the two points in Figure 2.3 meet at the frame boundary, regardless of whether the boundaries are straight or curved. Second, the above argument considers static scene points only. For scene points that are moving, the image paths will depend on the parallax just discussed and on the scene motion. Only the latter creates synchronization problems at frame boundaries, as we discuss in the next sub-section. Third, the motion paths discussed above were for one camera only. How are the motion paths for the two cameras related? Points that are a finite distance away will enter each camera's field of view at slightly different frames. This is the window violation problem of standard stereo cinema. In addition, the shape of the corresponding red curves will be slightly different for the two eyes, which causes

disparities to vary slightly over time. The disparity variations are so small, however, that they go unnoticed.

2.3.3 Partition and alignment of space-time blocks

At the stage, the calibration and registration has been done, so that the pixels in each frame have been remapped to the cylindrical projection surface. Let the two videos have N frames each. In the case that the camera turns 360 degrees, frame N would be registered with frame 0. If we were to display the image sequence in stereo on a cylindrical projection screen, we would see the scene through a window translating slowly over time, namely we would see the scene in stereo as captured by the rotating camera and projected in the correct direction. At any time, we would see only the field of view of the stereo camera, however. The problem that we are solving is to take this stereo video and make a panorama stereo video from it, which is defined over the entire cylinder and at every time.

Our solution to this problem is to partition each of the stereo XYT volumes into blocks (parallelipeds), and to blend the blocks together to form left and right video textures. To explain how this partitioning and stitching works, we continue for now with the simplified case that the camera rotation is uniform.

Suppose that it takes T frames for any point to enter the field of view at the right edge and exit the field of view at the left edge. (This time is constant when the camera rotation speed is constant, and the scene point is static.) We partition the entire image sequence into a set of consecutive blocks, each of T frames. We then re-align the blocks so that they all start at the same frame in the video texture. See Figure 2.4. In this example, the entire video is 120 seconds and is partitioned into five blocks that are 24 seconds each.

Consider a scene point at infinity that enters the field of view somewhere on the right diagonal of the first block. Since this point is within the field of view for T frames, its path extends beyond the first block. When the blocks are aligned so that

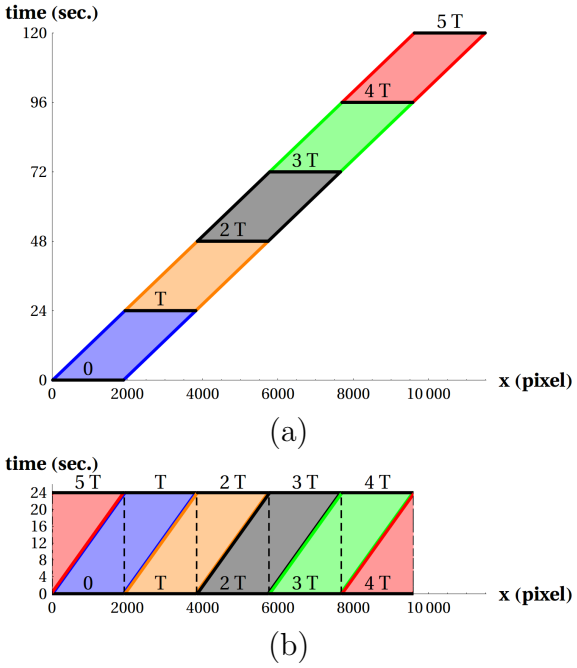


Figure 2.4. For a frame sequence captured by a camera performing a full turnaround in $N = 5T$ seconds at constant speed. (a) The full original space-time volume divided in five non-overlapping blocks. (b) The blocks are aligned to start at the same time.

they all start at the same frame, the vertical path followed by this point wraps around from frame T to frame 0 and again forms a vertical line. See Figure 2.5.

Recalling the arguments of Section 2.3.2, if a static scene point at a finite depth were to enter the space-time volume at the same frame, then it would take a curved path instead of a vertical path. The curved path would also wrap around from frame T to frame 0, and rise again to meet the vertical dashed line at the diagonal boundary. Thus, the paths of static points in the scene would be continuous both at their temporal boundaries (allowing the video to loop with a period of T frames) and also at the seams that define the frame's spatial boundaries, *i.e.* the diagonals.

The continuity at the temporal boundary (looping) does *not* depend on any assumptions about the points being static in the scene, nor does it depend on the

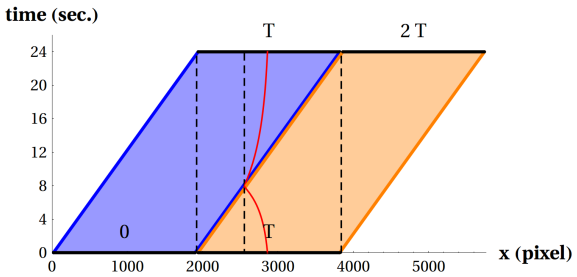


Figure 2.5. Motion paths of two static objects, one far away (central dashed vertical line) and one close-by (red curve). In both cases, the motion paths are continuous and loop. The video goes from frame 0 to T-1 and then loops so that frame T equals frame 0.

camera rotation speed being constant. *This looping property at the temporal boundaries always holds.*[3] The continuity at the frame boundary, though, often does not always hold exactly. Discontinuities can occur when there is scene motion (see Fig. 2.6) and also when there is vertical parallax, or lighting changes over time, or exposure changes due to a camera aperture change e.g. if one is in in shutter priority mode.

How can one avoid such visual seams? Existing monocular methods that render dynamic mosaics [4, 57] attempt to minimize seams both in space and time by using sophisticated image stitching *e.g.* a combination of graph cut matching and gradient smoothing. As we discussed in Section 2.2, however, it is unclear whether such methods could be extended to dynamic stereo since such methods use thin slits or small blocks, and there are fundamental difficulties in stereo motion synchronization in these cases. Our approach to avoiding visual seams is to avoid boundaries as much as possible, by using full frames rather than slits or small blocks. We still need to stitch boundaries together, however, and for this we use blending as we describe next.

[3] The only exception occurs when the frame at 360° loops to the frame at 0°. In this case, if there are moving scene points at these limit frames and/or the lighting changes, then the video will not be loopable at these points. The problem could be lessened by starting the capture in a direction in which the scene is static, or using a blending technique similar to what we discuss next. Similarly, if the panorama is less than 360 degrees and the scene has motion at frames 0 (or N), our method will not produce looping there.

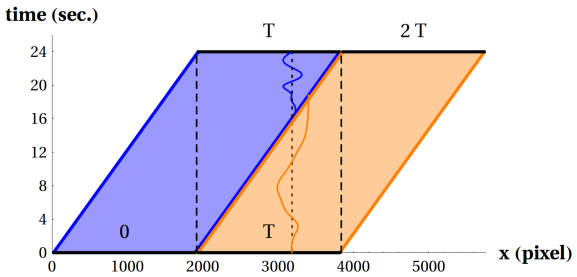


Figure 2.6. For an object moving in time in a small area (a leaf for instance), the motion is continuous at the temporal boundary (horizontal edge), but there will be a motion discontinuity at the spatial boundary (seam), namely the diagonal edge.

2.3.4 Blending adjacent blocks

To blend adjacent blocks, we decrease the duration T of each block and shift the block by the number of pixels covered during that decrease in duration. See Figure 2.7 for an illustration of what happens for static scene points, and see Figure 2.8 for the case of a moving scene point. In these figures, T has been decreased from 24 to 20 seconds. In the overlap region, we blend the frames together using a simple linear ramp function (see Figure 2.9).

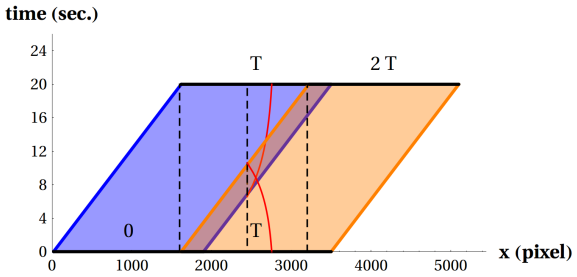


Figure 2.7. Images are blended near the boundaries of two blocks.

The reason we use blending, rather than a more sophisticated technique such as gradient based smoothing [51], is that it seemed to be sufficient in many cases. Although blending does leads to a duplication of the points – or "ghosting" – the

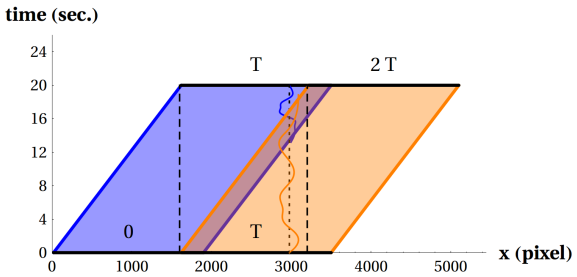


Figure 2.8. For an object moving in time (a leaf, for instance), motion is blended over the overlap between the two blocks.

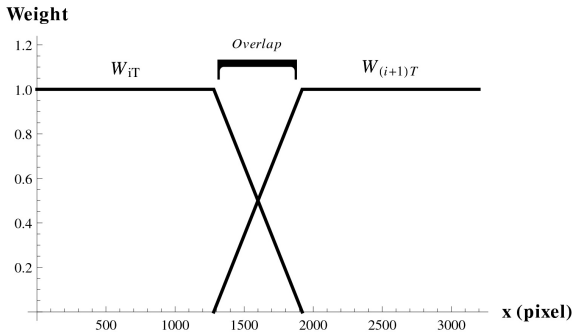


Figure 2.9. Blending function for overlapping frames. Weighting functions $W_{iT}(x)$ and $W_{(i+1)T}(x)$ are given in Sec. 2.3.5.

duplication is typically not perceived, unless one is looking for it. We believe there are several reasons for this. First, the blending is continuous over time, with one copy fading out over time and the other copy fading in, and so there are no salient low level features that grab the eye such as block edges or temporal popping. Second, in the case of motion texture such as leaves or grass blowing, or waves evolving, the "texels" often change over time or undergo occlusions and so are not visually isolated and trackable entities. Third, the blending window translates which results in further variation that may mask the duplication of texels. These reasons will be discussed again in Sec. 2.5, where our perceptual experiment on the visibility of blended regions is presented.

2.3.5 Non-uniform camera rotation and blending

We next turn to the more general case that the camera rig is rotating at a non-uniform speed, and so the boundaries of the space-time volume are curved (see Fig. 2.10). To handle this case, we continue to use a constant duration T for all blocks, but now we vary the blending overlap. The blending width depends on the frame-to-frame overlap at each boundary which corresponds to the distance between two adjacent diagonal curves in Figure 2.10(b).

To ensure there is some overlap between each pair of adjacent frames, T must be chosen carefully. Let $d(i, i')$ be the angular distance (in units of pixels on the cylindrical projection surface) travelled by the camera between frames i and i'. To be conservative, we require that, for all frames j, the distance $d(j, j+T)$ is less than or equal to some chosen fraction α of the width W of the original frame. This ensures a blending overlap of at least $(1-\alpha)W$ pixels between frames. Given α and W, a sufficient condition on T that ensures some overlap is that, for all frames j,

$$\sum_{i=j}^{j+T-1} d(i, i+1) \;\leq\; \alpha W \;.$$

In our experiments, we chose $\alpha = 0.8$ which ensures a minimum overlap of 20%. The result is that the overlap is smaller when the stereo rig rotates faster and the overlap is larger when the rig rotates slower.

2.3.6 Stereo motion synchronization and blending

For scene points that are imaged simultaneously by the left and right cameras, stereo-motion is automatically synchronized. Since we are using the full frames, this includes the vast majority of scene points. Indeed asynchronization can occur only near the frame boundaries, namely where there is a window violation.

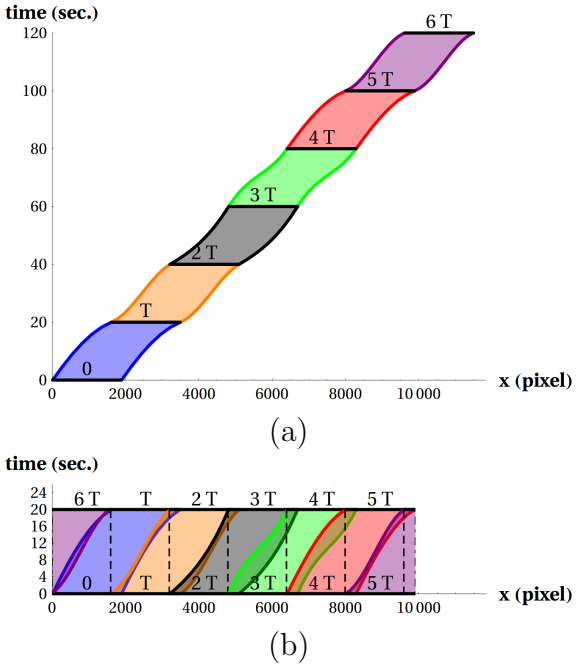


Figure 2.10. Similar to Figure 2.4 except that here the camera rig rotational velocity is not constant. The blocks are no longer aligned. Blending over frame boundaries is used to reduce the visibility of seams.

For those points that are imaged near a frame boundary at some given time, it is important to distinguish synchronization issues from blending issues. Blending can introduce duplicated scene points within the left or right camera's video, and the motion of a point and the ghost with which it is blended will be asynchronous. If such a point (or its ghost) is seen simultaneously in the left and right views, though, then its left and right view will be synchronized. The result is that there may be two blended but distinct stereo copies of the scene points. The disparities of each stereo copy (the point and its ghost) will be correct. In the following section, we present a perceptual experiment that tests how visible these duplicated points are in practice for different scenes.

2.4 Examples

In our capture experiments, we used two Canon HFS11 cameras on a fixed tripod. This allowed camera rotation around an almost vertical axis (y axis). The distance between the centers of both lens was about 6.5 cm, similar to the typical distance between human eyes. To synchronize frame capture as well as zoom, both cameras were controlled through the LANC protocol (a LANC Shepherd was connected to the cameras by RA-V1 remote control adaptors).

To speed up the experiments, we down-sampled the HD original content, from 1920×1080 resolution to 960×540. The final panoramic video is high-resolution at about 6500×540 pixels per eye. A GPU was used for the dense frame calibration and the blending. Each example took about an hour to render, separated about evenly between calibration and blending, on a laptop with an NVidia GeForce 8400M graphics card and an Intel dual core T7500 2.2 Ghz CPU and 2GB of RAM. Both steps could be accelerated by having a separate thread handling disk operations (loading and saving frames). Moreover, both the calibration and the blending steps have low memory requirements. Every output frame of the video texture can be blended in parallel, which allows the method to render very high-resolution 360 degree textures. This contrasts with other approaches [4, 57] that require solving a large minimization over the whole space-time volume.

We present two examples: one containing a river and another containing a field with blowing tall grass. See Figs. 2.11 and 2.12 for a third of a single frame of each video (full videos are available online at [1]). The reason we show a third of a frame only is that the 12:1 aspect ratio (horizontal:vertical) of the entire frame is very large and the vertical dimension would be excessively squeezed.

Figure 2.12 shows a single frame from the left camera's panoramic video texture and compares (a) no blending, versus (b) blending. At first glance, the seams in (b)

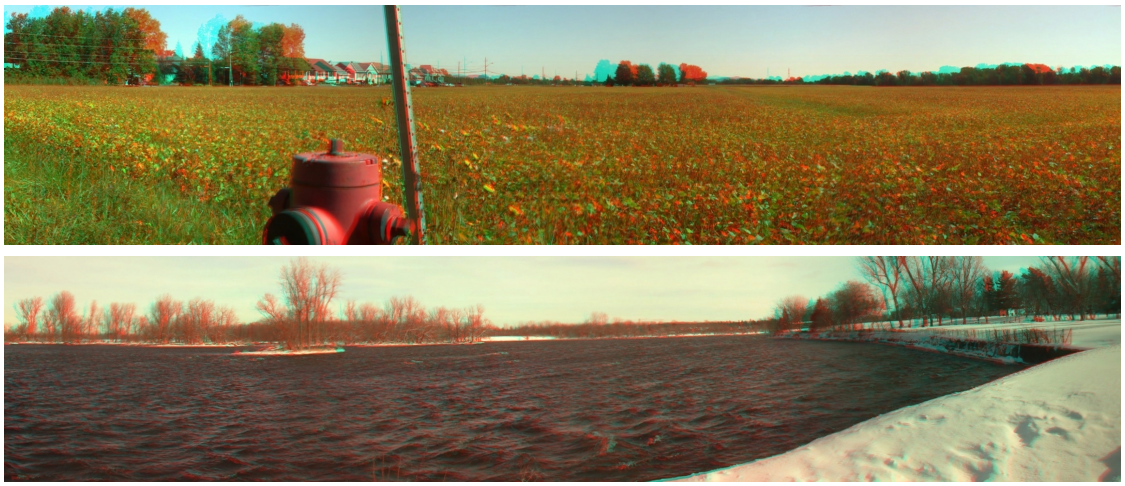
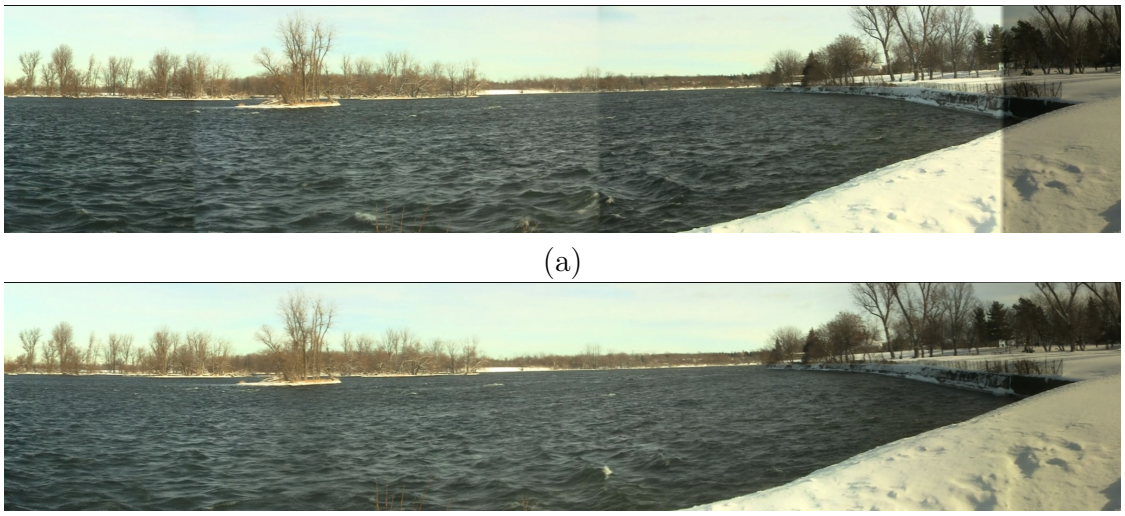


Figure 2.11. Third a frame (120° field of view out of 360°) of two panoramic stereo videos of a field and river, shown in red/cyan anaglyph format.

are slightly visible when seen below (a). However, this is an illusory contour effect. The reader should cover up (a) when examining (b).

To fully appreciate the stereo effects, the videos should be displayed with correct perspective. We have projected them on a cylindrical screen made of a silver fabric that maintains light polarization. The screen is about $1.5m$ high with a $4.5m$ diameter. A multiprojection system [68, 69] was setup with half the projectors polarized horizontally and the other half polarized vertically, and viewed with glasses for polarized projection. To our knowledge, this is the first time that a 360 panoramic stereo video texture has been captured, computed and displayed.

Finally, although our method is motivated by the problem of stereo video panoramas, it also applies to the more specific problems of static omnistereo and to dynamic monocular panoramas. For example, we tested our method on monocular input videos from [4], which were shot by a single camera rotating on a tripod. These sequences do not have parallax since the camera undergoes pure rotation. The result for the *Yachts* sequence is available online at [1]. In this example, camera



(a)

(b)

Figure 2.12. Third a frame (120° field of view out of 360°) of one camera's panoramic video with (a) no overlap between the blocks. (b) a minimum overlap of 20% between blocks. The overlap blends motion discontinuities as well as lighting changes.

rotation stops at a few discrete positions which causes blending overlaps to increase considerably. Nonetheless, our method produces very good results.

2.5 Perception Experiments

We presented the above examples on the cylindrical screen in our lab and observed informally that the stereo experience was excellent. We observed that the blended regions used by our method were typically not noticeable, even though the ghosting could be observed under scrutiny. In particular, once one knows what to look for in each scene, the ghosting becomes more obvious although we found this varied from scene to scene. We also found that naive observers, namely visitors to our lab, did not notice the ghosting until it was pointed out to them.

To better understand the visibility of ghosting and how it might vary from scene to scene, we carried out a perceptual experiment. The experiment measured how fast naive observers could detect moving blended regions for different blending widths and for different scenes. We also compared performance with and without stereo, since it was not obvious whether stereo disparities would make the ghosting more or less salient.

2.5.1 Stimuli

Five new dynamic stereo scenes were captured: `flowers, bush, smoke, lake,` and `river` (see Fig. 2.13). Unlike the scenes used in our method, these new scenes were shot in a fixed camera direction, similar to stereo versions of "dynamic textures" [21] as the term is used in computer vision. The reason we did not rotate the camera and compute panoramas is that it was not necessary for our purposes here, namely to investigate the visibility of the moving blending window and the ghosting that results. We also wanted to introduce an experimental design that could be easily replicated by others.

Figure 2.13. The five scenes used in the experiment, namely from left to right (top) `flowers` **and** `bush` **(bottom)** `smoke`, `lake` **and** `river`.

Videos were displayed at 30 fps with a 16:9 projection aspect ratio on the same cylindrical screen that was used as in Sec. 2.4. distance of about 2.5m from the screen. The projection area covered 1.2m vertically. The field of view was thus about 45 degrees horizontal and 25 degrees vertical.

Observers wore stereo glasses for polarized viewing. In each trial, a video was presented with a moving blending region. See Fig. 2.14. The blending region moved either left or right at a speed of about 3° per second which was similar to that of the panning cameras used in our capture experiments in Sec. 2.4.

The moving blending region in each trial was restricted either to the left or right half of the video. The regions to the left and right parts of the blended regions (blue and orange, respectively, in the figure) were composed of two temporally non-overlapping regions of the XYT volume. In the blending region (gray in the figure), the spatial overlap was thus asynchronous, which produced ghosting of any moving objects.

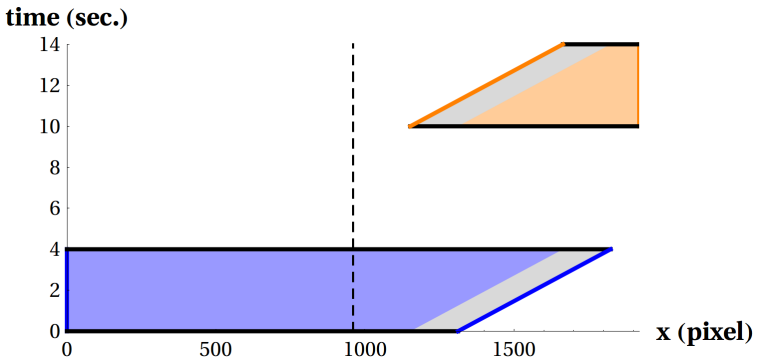


Figure 2.14. To create a stimuli from a video, a blended region (gray) is created by overlapping the left part (blue) with the delayed right part (orange). The delay was set in this figure to 10 seconds for display purposes, but a delay between 25 and 35 seconds was used in practice. Here, the blended region is originally located at $\frac{3}{5}$ of the frame and moving right. Its width is $\frac{1}{12}$ the whole frame.

Within the blending region, the stereo disparities of each moving object and its ghost are correct. As such, the depth cue of a moving object and its ghost do not conflict. Thus when a blending window arrives at and then passes over each moving object, the object is first seen without ghosting, then the ghost appears gradually as the window passes over, and then the original object fades and only the ghost remains as the window moves on.

The scenes were chosen to have similar left and right depth distributions. This was to avoid having subject's attention drawn consistently to one part of a scene, as happens when there is one dominant, close, spatially isolated and moving object, especially in stereo.

Three types of conditions were examined. The first was the underlying video used, namely we asked if ghosting was more visible in some of the videos than others. The second condition was the blending width. The third condition was monocular versus stereo. In the monocular version, the *same* image was displayed for both eyes. (This is just the case of standard 2D cinema.) Two versions of the experiment were performed. Version A tested 2 different blending widths, namely $\frac{1}{3}$ and $\frac{1}{12}$ of the frame width. Version B tested a single blending width, namely $\frac{1}{3}$ of the frame as in our panorama method, and compared stereo and monocular videos. Each experiment had $10 = 5 \times 2$ conditions in total.

2.5.2 Observers and Procedure

For each condition, we wanted to measure the minimum display time required to detect if the blended region was located in the left or right half. We tested 17 men and 3 women for a total of 20 subjects, 16 of which were graduate or undergraduate students at l'Université de Montréal. Ages ranged 18 to 65. Participants had varying degrees of familiarity with stereo cinema. All participants were naive about the motivation of the experiments. All signed an informed consent form.

Stereo perception of each participant was tested using a random dots stereogram for which the disparities of two squares were adjusted so that one was to be perceived closer than the screen, and the other further away. The two squares were respectively located to the left and right of the stereogram and participants were asked to say which square was closer. This simple test was performed several times, each time randomly flipping the location of the squares. Two male participants failed the test and their results were ignored in Fig. 2.16.

Each participant ran either version A or B of the experiment. The task in both experiments was the same, namely identify whether blending artifacts were present in the left or right half of the video. Before the experiment began, typical blending artifacts were shown to the participants using the `flowers` and `lake` sequences.

The procedure for the experiment itself was as follows. Before each trial, a small cross indicated the screen center. Observers were not required to look at the cross, however, nor were they required to keep their head fixed. Although this was less controlled than what is typical in a psychophysics experiment, we felt that such a procedure would be better representative of how people might locate artifacts in a real stereo video such as in our rendered panoramic stereo videos. The stimuli video was then displayed for some varying duration. After the video finished, participants answered if the blended region was to the left or the right of the midpoint, using joystick buttons. We did not give the possibility to answer *I don't know*, because people usually have different confidence levels in their answers.

The 10 conditions appeared in random order and multiple times. For each condition, the initial display time was 2 seconds. The display time was then updated to compute a detection threshold of about 75 % probability of being correct. [4] We used the up/down method proposed by [20], later transformed by [72]. Following

[4] We used this method rather than allowing for infinite display time (and computing reaction times) since in a pilot study we found that some observers took too long to respond and this led to high variabilities.

the 1up/2down rule, the display time increased after every incorrect response, and decreased after two consecutive correct responses. This strategy made the display time increase if a participant guessed each response. The increase and decrease time factor was set to $\sqrt{2}$. For faster convergence towards the threshold, we used the 1up/1down rule until the 1st reversal as suggested in [72].

Our termination criteria considered *reversals* of direction [38], i.e. when two good consecutive responses are given after a wrong answer, or vice-versa. A condition was stopped after 10 such reversals. To avoid losing time over a sequence too hard for a participant, we also stopped if a participant missed at a display time of 4 seconds. See Fig. 2.15 for a few data examples.

2.5.3 Results

Over both experiments, participants did on average 255 trials in total (the minimum and maximum number of trials were respectively 169 and 348). For each condition, the final threshold display time was computed as the average time at the last 8 reversals. If a participant missed at a display time of 4 seconds, then this time was considered as the threshold instead of the average. Mean thresholds across observers as well as the standard error of the mean are shown in Fig. 2.16.

The main finding is that, for both experiments, the type of scene had a large effect. Thresholds were lower in the vegetation scenes (`flowers` and `bush`) than in the other scenes (`smoke`, `river` and `lake`), that is, the blending is easier to detect in the vegetation scenes.

The other conditions had smaller effects, if any. For experiment A, large blending widths seemed to be slightly easier to detect in the vegetation scenes, but more difficult to detect for the smoke and water scenes. For experiment B, monocular and stereo conditions gave similar thresholds. This suggests that detecting the blending is basically a monocular task. To support this claim, we tested the two people that

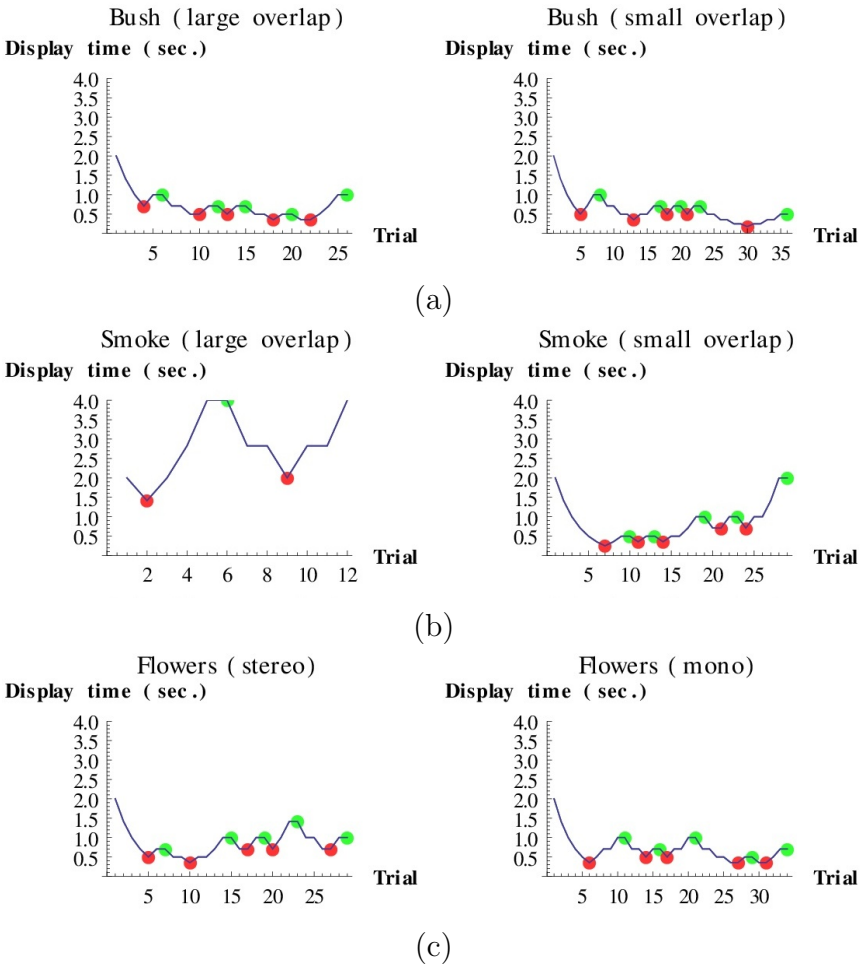
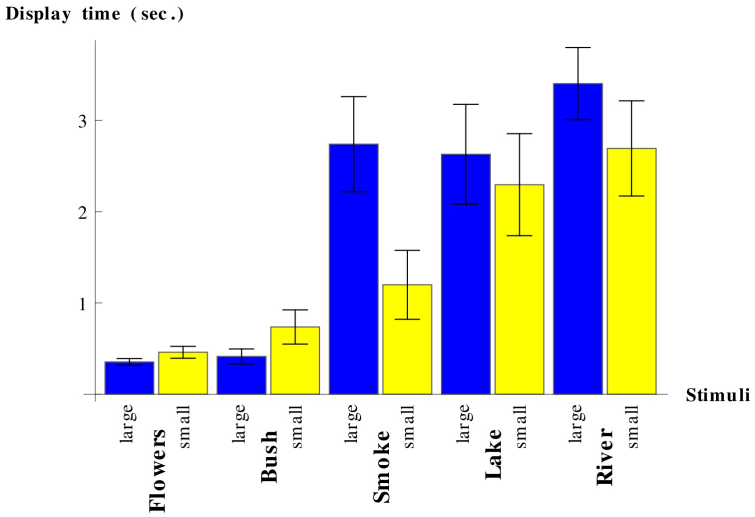
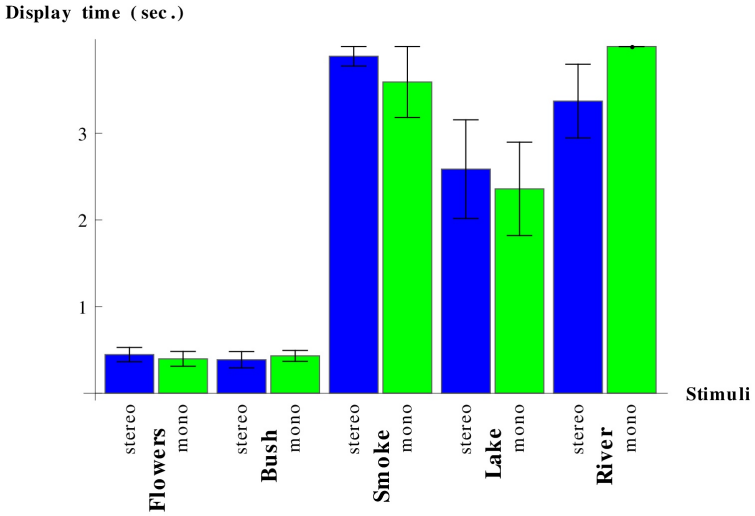


Figure 2.15. Example of data gathered for experiment A (a,b) and experiment B (c). The red and green circles correspond to reversals. The trials stopped at the 10^{th} reversal or if a participant missed at a display time of 4 seconds.



(a)



(b)

Figure 2.16. Detection time thresholds for (a) experiment A (b) experiment B. Note that the blue bars in the two experiments correspond to the same stimuli, namely a large blending width seen in stereo.

failed the stereo test and they gave similar thresholds for experiment A as the other participants (data not shown).

Finally we note that, in the debriefing sessions, participants mentioned that they used different strategies to improve performance for very small display times. For example, some looked only on one side (left of right) and tried to detect if there was blending or not. If no blending was seen, then the chosen answer was the other side. Others reported learning specific scene details where blending was more apparent.

2.5.4 Discussion

The differences in thresholds for the vegetation scenes versus the smoke and water scenes suggest that our panoramic stereo video method is better suited for some types of scenes than others. The method seems to be very well suited for the smoke and water scenes, since the blending is quite difficult to detect even after prolonged viewing (several seconds) and much practice. We believe the reason why blending detection is so difficult for these scenes is that there are no well defined features that can be tracked from frame to frame. Smoke is partly transparent anyhow, and the blending seems noticeable for smoke scenes mostly because the blending window is a vertically oriented ramp which is unnatural. For the water scenes, features on the water surface appear and disappear rapidly over time, varying as waves move, and blending seems noticeable mostly because it produces slightly lower wave contrast. We would argue that blending works quite well for such video textures and that previous monocular methods based on sophisticated stitching using graph cuts are unnecessarily complicated.

The vegetation scenes are arguably more problematic for our method. In these scenes, the ghosting causes moving features such as leaves to appear and disappear through time, as if they had time-varying transparency. Although leaves can appear and disappear over time because of occlusions and shadows, time varying transparency is unnatural and appears so when one notices it. This might explain

why larger blending widths which create more ghosting (simply because of the larger width) were slightly easier to detect than the smaller blending widths in experiment A.

2.6 Conclusion

This paper has introduced a method for computing panoramic stereo video textures using a pair of off-the-shelf consumer video cameras. There are several key ideas to the method. First, it uses the full frame video which gives automatic stereo motion synchronization for the vast majority of the visible points. This synchronization issue would be problematic if one were to use slits or small blocks as in previous stereo or motion panorama methods. Second, rather than using graph cuts and/or smoothing to stitch together seams from different parts of each camera's XYT volume, we use simple blending. While blending can create ghosting when points are moving, we found that this ghosting goes unnoticed under casual viewing. The reason is that the blending occurs over a large window and this window moves across the image over time.

This paper also presented a perceptual validation of the rendering method. We found detection of blending is basically a monocular task. We also found that blending is easier to detect for some scenes than others. In particular, when well defined moving objects such as branches or leaves are blended, visible ghosting occurs. Although this ghosting goes unnoticed under casual viewing, we found that it can be detected for such scenes when the task is to do so. Future work could try to improve our method for the cases in which simple blending is not sufficient, for example, by addressing the challenging problem of how to remove the ghosting within each frame [67] while also preserving the transition from an object to its ghost over time.

2.7 Appendix: Panoramic Stereo Camera Calibration

This section deals with camera calibration from frames captured by rotating two cameras on a tripod for a full turnaround of 360 degrees. Using these captured images, the method calibrates both internal and external parameters of the cameras.

2.7.1 Sparse frame calibration

Let the image sequence be $I_{\theta_1}, I_{\theta_2}, ..., I_{\theta_N}$, where $\theta_1 = 0$ and $\theta_N = 2\pi$, and where each image is of size $W \times H$ pixels. We use $N = 60$, and $\theta_{i+1} - \theta_i \approx \frac{2\pi}{N}$ radians. Left and right internal camera parameters $\mathbf{K}_{l,r}$ are assumed to be constant over all images. We further define the axis of rotation of the stereo rig to be the y-axis and thus, the perspective projection model of the rotating stereo rig is in general given by:

$$p_{i_{l,r}} = \mathbf{K}_{l,r}\ \mathbf{R}_{l,r}\ \mathbf{T}_{l,r}\ \mathbf{R}_y(\theta_i)P$$

where $p_{i_{l,r}}$ is a pair of homogeneous corresponding pixels in left and right images i, $P = (X, Y, Z)$ is a point in the world, $\mathbf{R}_y(\theta_i)$ is a rotation matrix around the y-axis that brings the rig back to its orientation at $i = 0$, and $\mathbf{R}_{l,r}\ \mathbf{T}_{l,r}$ bring the left and right cameras at $i = 0$ to have axes and origin aligned with the rig. Also, there is typically up to a 3 ms time difference between left and right camera capture, even if using the LANC protocol for synchronisation. We model this difference by considering $\mathbf{R}_y(\theta_i + d_r \frac{\delta\theta_i}{\delta t})$ instead of $\mathbf{R}_y(\theta_i)$, where d_r is non-zero for the right camera only.

We can remove two degrees of freedom by observing that the scene and the rig at $i = 0$ are defined up to an unknown scale factor and a y-rotation. Fig. 2.17(a) shows how we model the stereo rig. We set the baseline joining the two cameras to be of unit length, assume that the z-axis passes by its center, and simplify the projection

model to:

$$p_{i_{l,r}} = \mathbf{K}_{l,r}\ \mathbf{R}_{l,r}\ \mathbf{T}_b\ \mathbf{R}_{rig}\ \mathbf{T}_z\ \mathbf{R}_y(\theta_i + d_r\frac{\delta\theta_i}{\delta t})P \tag{2.1}$$

where $\mathbf{T}_z$ undoes the camera rig translation along the z-axis, $\mathbf{R}_{rig}$ rotates the rig back to canonical orientation, $\mathbf{T}_b$ is a ± 0.5 translation along the x-axis that removes the baseline separation of the two cameras, and $\mathbf{R}_{l,r}$ is as before.

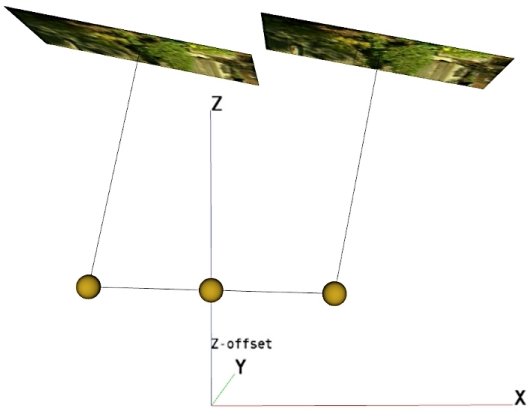
The number of parameters to be solved for is $N+20$: the focal length f, the image center o_x, o_y and the radial distorsion parameters k_1, k_2 of both cameras, the time difference d_r, the six components of rotations $\mathbf{R}_{l,r}$ as well as the three components of rotation $\mathbf{R}_{rig}$, the z-offset, and the $N-1$ rotation angles $\theta_{i,i+1}$ between consecutive images.

As a first approximation, camera parameters were estimated by ignoring camera translation, *i.e.* setting $\mathbf{T}_b$, $\mathbf{T}_z$ and $\mathbf{R}_{rig}$ to be the identity matrices, and using standard techniques, namely robustly matching SIFT features between frames using a homography model and RANSAC, followed by an initial estimate of the camera parameters and then bundle adjustment, taking radial distortions into account [29]. The estimates are then improved by allowing for non-zero camera translation, namely performing another bundle adjustment that triangulates features in 3D[66]. These are then reprojected in the images and we consider pixel distance to the corresponding features as errors to minimize. Reprojection errors are usually below 0.5 pixels once minimization of the parameters is achieved.

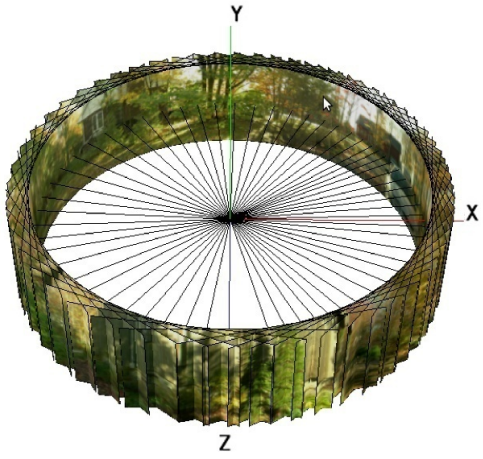
2.7.2 Dense frame calibration

Internal parameters of the cameras estimated in the previous section are used to warp all frames on the cylindrical omnistereo image space. By unwrapping the cylinder, camera rotation produces a horizontal shift that we want to estimated in this section.

The relative orientation of a frame with respect to the previous frame is estimated by tracking features using the KLT method [64]. Note that SIFT features could



(a)



(b)

Figure 2.17. The autocalibration process estimates the rotation angle between consecutive frames and, for each camera of a stereoscopic pair, the focal length of the camera and radial distortion. (a) The external parameters of the stereo setup are modeled by a straight baseline of length 1 with a z-offset from the origin. Three 3D rotations are allowed, indicated by spheres. (b) For the rendering stage, motion parallax artifacts are ignored, as if the projection center of both cameras was centered on the rotation axis.

be computed instead and matched (as in the calibration process described in Sec. 2.7.1), but that we are here using the KLT method which is faster and which is more appropriate for a dense frame sequence. More weight is given to features near the center of the images where parallax is minimum. This can be seen by looking at the motion field for forward camera motion, where parallax is 0 at the image center and increases towards the image borders [42]. Accumulated drift is adjusted so that all the frames cover exactly the calibrated mosaic static background.

One concern is whether this calibration method makes sense in the context of dynamic video, namely when objects are moving. We assume that, although there will be motion in many parts of the scene, there will also be many static parts as well visible and that a single motion vector (v_x, v_y) can model the dominant motion that is due to the camera rotation. The v_y component is used only to model camera vibrations or jitter. A robust fit is done using RANSAC.

2.7.3 Adjusting Disparity

It is usually preferable that the zero image disparities roughly occur at the distance of the projection screen. For standard stereo imaging, homographies can be used to adjust the toe-ing in/out of the cameras. For omnistereo, the right omnistereo background is shifted to the right until objects located at this distance have zero disparity.

Chapitre 3

TOURNAGE OMNISTÉRÉO DE MOUVEMENTS NON RÉPÉTITIFS

Nous avons présenté au chapitre 2 une nouvelle méthode de création de vidéos omnistéréo par assemblage d'images capturées avec deux caméras en rotation autour d'un axe. Ces vidéos peuvent jouer de manière répétitive (en boucle) sans coupure de mouvement. Cette méthode suppose des mouvements stochastiques et localisés, par exemple les feuilles d'un arbre au vent ou les vagues d'une rivière. Le présent chapitre propose une méthodologie qui ouvre la voie à la production de films omnistéréo, en permettant l'ajout de mouvements non répétitifs et non localisés, comme ceux des acteurs. Nous l'avons testée par le tournage d'un court métrage de 4 minutes. Ce chapitre fait suite à nos travaux publiés dans [15] sur l'ajout de ce type de mouvements à une image omnistéréo statique. Nous décrivons la méthode de tournage à la section 3.1, l'alignement (ou recalage) des images à la section 3.2, et leur assemblage à la section 3.3.

3.1 Tournage de couches successives de mouvements

La méthode de tournage, applicable dans un environnement relativement contrôlé, comprend deux étapes : le tournage d'une image ou d'une vidéo omnistéréo qui servira de fond statique ou dynamique et la capture stéréo de plusieurs séquences de premier plan composées de mouvements non répétitifs. La vision par ordinateur permet la superposition des séquences sur le fond omnistéréo, sans avoir à utiliser des installations complexes avec écrans bleus (*chroma keying*). Nous supposons que le fond et les séquences sont tous capturés sur un trépied placé au même endroit.

Figure 3.1. Chaque image de premier plan doit avoir des marges gauche/droite (indiquées en rouge) sans mouvements non répétitifs.

Lors du tournage d'une séquence de premier plan, la caméra stéréo suit une action qui doit rester entre des marges gauche/droite. Ces marges sont essentielles dans le processus d'assemblage décrit à la section 3.3. Elles doivent être nettes et au moins aussi larges que le déplacement entre deux images consécutives (par exemple, de 30 à 60 pixels). En pratique, un temps exposition d'au plus $\frac{1}{125}$ ou $\frac{1}{250}$ évite le flou causé par une rotation rapide de la caméra. La figure 3.1 montre un exemple d'image avec les marges gauche/droite indiquées en rouge.

3.2 Alignement des images de premier plan

L'alignement (aussi appelé recalage) des images stéréo d'une séquence de premier plan estime la position de chacune de ces images par rapport au fond omnistéréo. Nous estimons la position de la première image en testant toutes les positions possibles par rapport au fond et en sélectionnant la position qui donne une erreur minimum (l'erreur peut mesurer, par exemple, la somme des différences d'intensité au carré). Nous estimons la position des images subséquentes par le suivi de leurs points saillants sur le fond avec la méthode KLT[64]. L'utilisation de tous les points saillants peut fausser cette estimation dans les cas, par exemple, de branches au vent ou d'un objet du fond qui se déplace dans la séquence de premier plan. Nous rendons l'estimation

robuste par la méthode RANSAC [22] qui suppose que 75 % des points saillants sont bons et suivis à une distance de 0.25 pixels.

La position de chaque image de premier plan est affectée par la parallaxe de mouvement. Pour diminuer l'effet de cette dernière, nous ajustons la position de chaque image de premier plan à partir du déplacement de ses points saillants par rapport à ceux de l'image de fond la plus proche (nous faisons référence ici à une image originale qui a servi à créer le fond). De plus, nous accordons plus d'importance aux points saillants près du centre des images puisqu'ils sont beaucoup moins affectés par la parallaxe. En effet, pour deux caméras parallèles en rotation autour d'un axe (voir la section 1.1), chaque caméra se déplace en suivant la tangente de la trajectoire circulaire des caméras. De façon instantanée, l'une des caméras se déplace vers l'avant, l'autre vers l'arrière. Ce déplacement de caméra T_z et la vitesse de rotation Ω_y autour de l'axe génèrent un flot de mouvement 2D $(v_x, v_y)^T$ dans l'image [42] :

$$\begin{pmatrix} v_x \\ v_y \end{pmatrix} = \frac{T_z}{Z(\tilde{x}, \tilde{y})} \begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} + \Omega_y \begin{pmatrix} -1 - \tilde{x}^2 \\ -\tilde{x}\tilde{y} \end{pmatrix} \tag{3.1}$$

où $\tilde{x}, \tilde{y}$ représentent un point sur le plan image dans la caméra (voir la section 1.4.1), et $Z(\tilde{x}, \tilde{y})$ représente la profondeur visible à ce point. La figure 3.2(a,b) illustre le flot généré par le déplacement T_z (le terme de gauche de l'équation 3.1), et celui généré par la rotation Ω_y en (c). Il est à noter que le flot correspondant au déplacement T_z dépend des profondeurs de la scène et peut donc créer un effet de parallaxe. Cependant, on observe à la figure 3.2 que, entre deux images proches capturées par une même caméra, le flot au centre des images est presque constant, et il est généré uniquement par la rotation de la caméra.

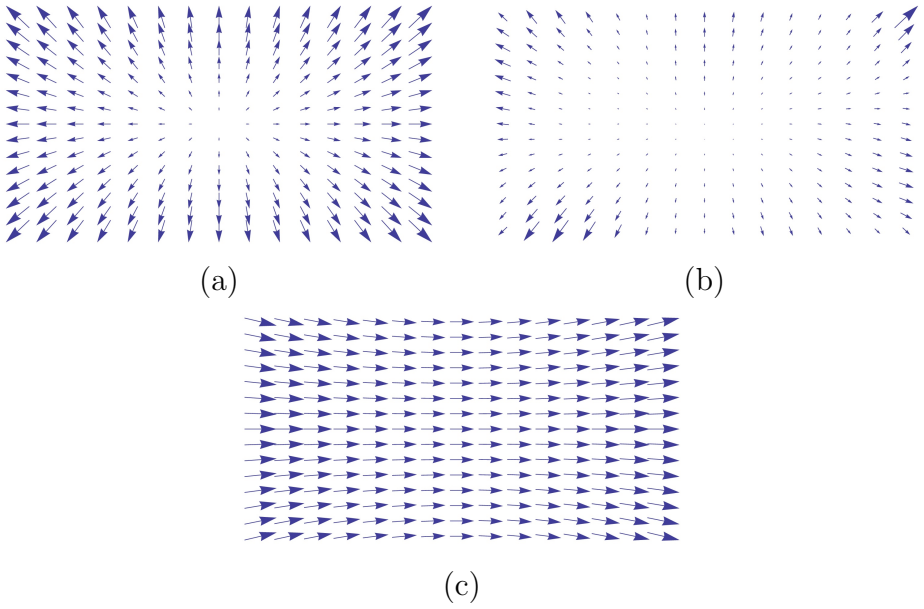


Figure 3.2. En (a) et (b), une caméra subit une translation vers l'avant, créant un flot optique minime au centre de l'image, mais qui s'accroît vers les bords. Le flot causé par une translation dépend des profondeurs de la scène, et s'accroît pour des objets rapprochés. En (a), la scène ne contient qu'une seule profondeur. En (b), deux objets plus rapprochés, en bas à gauche et en haut à droite, augmentent la magnitude des vecteurs ce qui crée de la parallaxe. En (c), une caméra subit une rotation vers la gauche, créant un flot presque constant vers la droite au centre de l'image. Ce flot ne dépend pas des profondeurs de la scène et ne crée donc aucune parallaxe.

3.3 Assemblage des images de premier plan

Nous décrivons tout d'abord notre méthode d'assemblage des images de premier plan sur un fond statique. Elle consiste à superposer sur le fond chaque image de premier plan de façon chronologique, ce qui permet de mettre à jour le fond selon les différentes interactions avec la scène. Cette méthode permet donc de conserver toutes les modifications apportées au fond qui ne sont plus dans le champ de vision des caméras. Par exemple, la figure 3.3 montre un acteur quittant une cour en voiture. Bien que cette dernière faisait partie du fond, la superposition fait en sorte que la voiture n'apparaît plus dans la cour. Il est à noter que les marges des images de premier plan peuvent rester visibles, d'où l'importance de leur netteté (voir la figure 3.4).

Pour éviter tout désalignement dû à la parallaxe entre le fond et les marges d'une image de premier plan, on peut utiliser une caméra stéréo telle que décrite à la section 1.3, et se baser sur l'observation clé de la même section. Par exemple, soit une image omnistéréo de fond construite à partir de fentes prises à la marge gauche d'images capturées en tournant la caméra stéréo. La marge gauche d'une image de premier plan s'aligne alors avec le fond en raison de l'absence de parallaxe entre les deux. Selon l'observation clé, la marge droite sera aussi alignée avec le fond puisque la parallaxe y est similaire.

Quant à l'application de cette méthode d'assemblage sur un fond dynamique, des recherches futures seraient nécessaires pour deux raisons. Premièrement, notre méthode de création de vidéos omnistéréo assemble des images entières au lieu de marges seulement. On peut donc s'attendre à de légères différences de parallaxe entre les marges des images de premier plan et le fond, différences qu'il faudrait minimiser. Deuxièmement, la superposition d'images entières sur le fond écraserait son dynamisme. Une approche plus appropriée serait de détecter de façon robuste

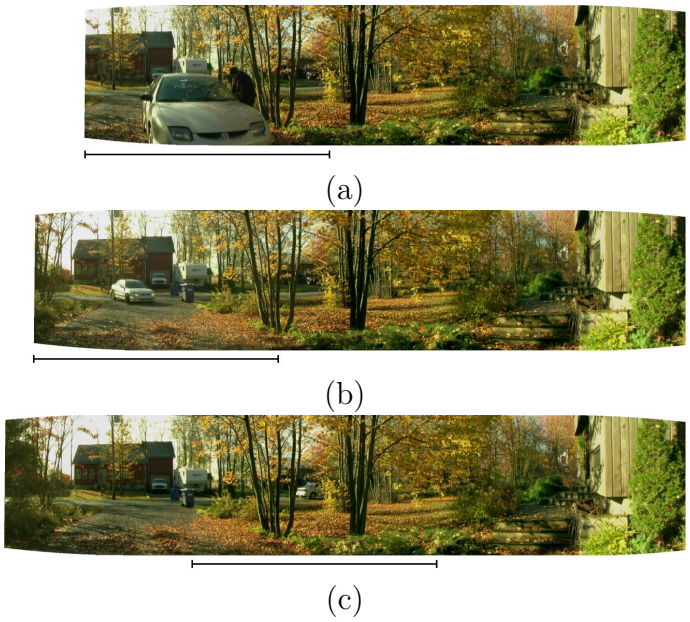


Figure 3.3. De (a) à (c), l'image de premier plan écrase le fond pour le mettre à jour. La voiture qui était dans le fond en (a) n'y apparaît plus en (c). L'emplacement de l'image de premier plan courante est indiquée par une ligne en dessous de chaque image.

les modifications apportées au fond, et de les superposer en laissant la vidéo de fond se dérouler hors des superpositions.

Figure 3.4. Processus de superposition successive des images de premier plan. Ce processus conserve les changements apportés au fond, tel que la voiture qui n'est plus garée dans l'allée. Seulement les images de la caméra gauche sont montrées.

PARTIE II

PROJECTION OMNISTÉRÉO

Chapitre 4

PROJECTION OMNISTÉRÉO À L'AIDE DE LUMIÈRE STRUCTURÉE

La projection omnistéréo sur un écran cylindrique pourrait se faire avec un (ou deux) projecteur 4K[1] très haute résolution muni d'une lentille très grand angle, mais ce système serait très dispendieux. L'utilisation de plusieurs projecteurs d'une résolution HD[2] ou moins permet de réaliser la projection d'une image unique très haute résolution à un moindre coût. Mais l'alignement manuel des projecteurs s'avère une tâche difficile. Nous procédons plutôt à l'alignement automatique à l'aide d'une caméra et de méthodes de vision par ordinateur. Ces méthodes déforment l'image projetée de façon à produire une image multi-projecteur cohérente.

4.1 Système multi-projecteur

Une projection multi-projecteur nécessite un nombre suffisant de projecteurs pour couvrir tout l'écran. Une projection omnistéréo polarisée nécessite de doubler le nombre de projecteurs afin d'obtenir une image pour chaque oeil. La polarisation linéaire de la lumière fixe à l'aide de filtres l'orientation de son oscillation à une différence de 90° entre les yeux. Par exemple, la lumière peut être polarisée horizontalement pour l'oeil gauche, et verticalement pour l'oeil droit. L'écran de projection doit être très réfléchissant pour éviter toute diffusion de la lumière et ainsi maintenir sa polarisation. De plus, les spectateurs doivent porter des lunettes munies de filtres qui, idéalement, permettent à chaque oeil de ne voir que l'image qui lui est destinée. Une projection polarisée sur notre écran omnistéréo double le

[1] Le standard 4K correspond à une résolution de 4096×2160 pixels.

[2] Le standard HD correspond à une résolution de 1920×1080 pixels.

nombre de projecteurs de 7 à 14. L'augmentation de la distance entre les projecteurs et l'écran peut diminuer le nombre de projecteurs nécessaires pour couvrir l'écran, mais ce n'est pas toujours possible, par exemple lorsque les salles des laboratoires ou d'exposition ont des contraintes d'espace. Nous utilisons une méthode d'alignement des projecteurs, développée par notre laboratoire[69], qui ne modélise ni leur position, ni leur lentille. Cette méthode nous permet donc de diminuer le nombre de projecteurs à huit par l'utilisation de miroirs convexes sans toutefois complexifier leur alignement.

Les miroirs maintiennent la polarisation de la lumière. Nous utilisons des miroirs première surface composés d'un revêtement métallique mince par-dessus une surface acrylique ou en vitre. La méthode utilise une caméra pour établir une correspondance entre les pixels de chaque projecteur et les pixels du contenu à projeter, c'est-à-dire qu'elle établit une correspondance afin de calculer l'image de chaque projecteur de façon à produire un résultat cohérent. Elle contraste avec celle de Raskar *et al.* décrite dans [56] qui doit estimer, à l'aide de deux caméras, la forme tridimensionnelle de l'écran ainsi que la position et le modèle de lentille de chaque projecteur.

La figure 4.1 montre les trois étapes de la mise en correspondance, qui suppose que la caméra peut voir l'écran en entier. Pour un écran cylindrique, la caméra est situé au centre et est munie d'un objectif très grand angle (fisheye). La première étape nécessite une mise en correspondance caméra-projecteurs à l'aide de codes lumineux (lumière structurée), qui peut être difficile à établir sur des surfaces susceptibles de produire des interréflexions, comme les écrans omnistéréo ou toute surface concave. Par exemple, dans le cas d'un écran cylindrique très réfléchissant, la lumière d'un projecteur peut rebondir d'un côté à l'autre. Nous présentons à la section 4.2 différentes méthodes de lumière structurée.

La figure 4.2 montre des photos de notre installation avec huit projecteurs. Bien que cette installation soit fonctionnelle, l'intensité lumineuse sur l'écran est beaucoup plus faible, d'une part parce que chaque projecteur couvre une surface deux fois plus grande, et d'autre part parce que l'ouverture des projecteurs doit être fermée au

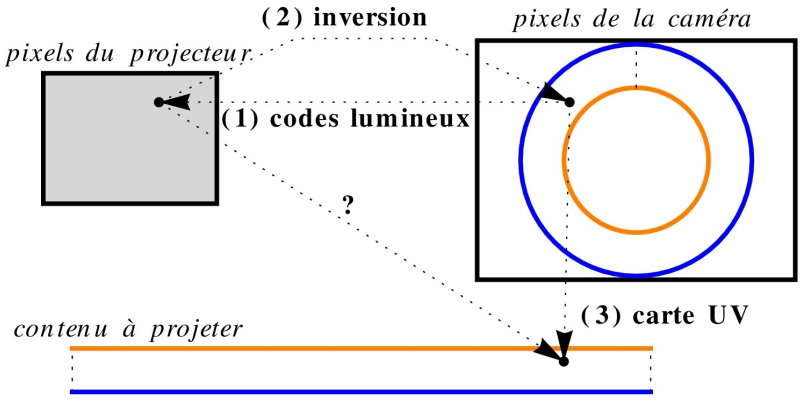


Figure 4.1. L'alignement des projecteurs trouve en trois étapes le lien entre les projecteurs et le contenu à projeter à l'aide d'une caméra. (1) Les correspondances caméra-projecteurs sont établies à l'aide de codes lumineux (de la lumière structurée ou non structurée) projetés par les projecteurs. (2) Les correspondances inverses projecteurs-caméra sont calculées. (3) Les correspondances projecteurs-contenu sont retrouvées par la définition manuelle d'une correspondance (une carte UV) entre l'image de la caméra et le contenu.

(a) (b)

Figure 4.2. L'utilisation de miroirs permet d'élargir la zone couverte par chaque projecteur et ainsi de réduire le nombre de projecteurs. (a) Chaque projecteur projette vers le haut sur un miroir convexe. (b) L'écran peut être couvert deux fois avec huit projecteurs, chacun ayant son miroir (la photo montre quatre des huit miroirs).

maximum pour limiter le flou causé par la courbure des miroirs. En effet, la courbure du miroir fait dévier les rayons lumineux d'un pixel de projecteur de sorte qu'ils ne focalisent plus tous au même endroit. Ce phénomène s'accentue avec l'ouverture du projecteur, qui augmente le nombre de rayons.

4.2 Lumière structurée et illumination indirecte

En vision par ordinateur, la reconstruction d'une scène réfère au procédé d'estimation de sa forme. Ce procédé s'avère utile dans plusieurs domaines, entre autres au cinéma où il permet de générer des modèles par ordinateur à partir d'objets existants ou de sculptures. Nous l'utilisons dans notre système multi-projecteur afin de retrouver la forme de l'écran.

La reconstruction peut être passive ou active. La reconstruction passive tente de retrouver la forme d'une scène à l'aide de caméras seulement, mais elle peut s'avérer une tâche difficile si les surfaces sont de couleur uniforme (un mur blanc par exemple), puisqu'il y a trop de similitude entre les points. Quant à la reconstruction active, elle facilite l'analyse de surfaces uniformes par l'ajout d'information à la scène sous forme de lumière ou laser. Une caméra observe la déformation de motifs structurés projetés sur une scène pour en calculer la forme.

Le qualificatif *structuré* signifie qu'un ou des motifs encodent la position des pixels d'un projecteur. L'étape de la correspondance, qui vise à associer chaque pixel $I_c[x, y]$ d'une caméra à un pixel $I_p[\hat{x}, \hat{y}]$ d'un projecteur, peut alors être directement établie : un pixel de la caméra observe un signal lumineux qui donne la position du pixel correspondant dans le projecteur. Le chapitre 5 présente différentes catégories de motifs de lumière structurée proposées dans la littérature. Nous montrons ici les bases de trois méthodes en vue de faciliter la compréhension de la problématique reliée aux écrans omnistéréo : les codes de Gray (une variante des codes binaires), la méthode par déphasage et la méthode de Gupta *et al.*

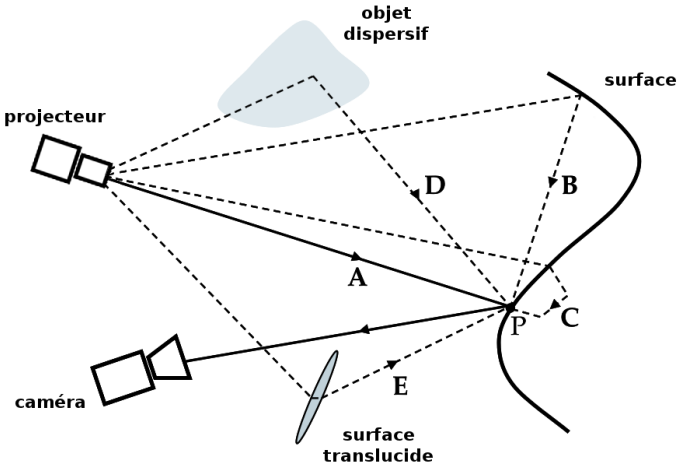


Figure 4.3. La radiance à un point P de la scène dépend de l'illumination directe d'un projecteur (A) et de l'illumination indirecte qui inclut l'interréflexion (B), la dispersion sous-surface (C), la dispersion volumétrique (D) et la translucidité (E). Tirée de [46] (traduction libre).

Un des problèmes majeurs des systèmes de lumière structurée vient du fait que la caméra mesure des intensités lumineuses $I_c[x, y]$ qui sont rarement égales aux intensités correspondantes projetées $I_p[\hat{x}, \hat{y}]$. Nous supposons ici que les intensités des images I_p et I_c varient dans un intervalle de [0,1]. À titre indicatif, nous introduisons un modèle de base qui montre la complexité du processus de capture de la lumière projetée. Dans un premier temps, le projecteur introduit une non-linéarité γ_p lors de la conversion du courant électrique en lumière. Puis, la lumière projetée interagit avec la surface selon une fonction Φ_s, inconnue *a priori*, qui modélise sa réflectance. La réflectance peut varier localement selon le type de surface et, pour des surfaces non diffuses, la quantité de lumière réfléchie vers la caméra dépend de sa position. Finalement, la caméra capture la lumière réfléchie vers elle, mais introduit aussi une non-linéarité γ_c et ajoute un bruit η. Tout ce processus peut être modélisé comme suit :

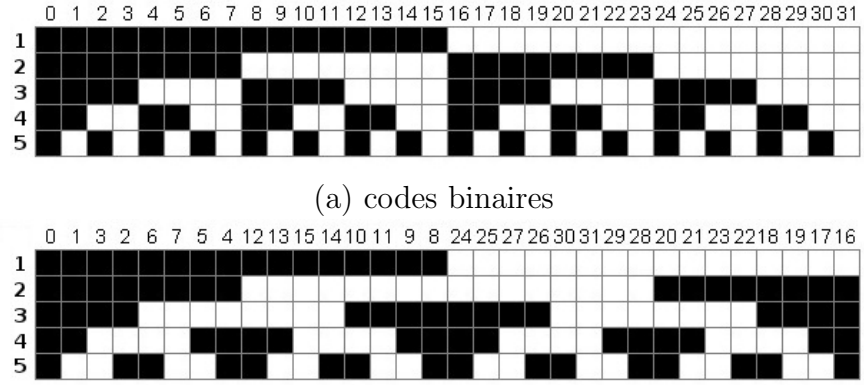
$$I_c[x, y] = \Phi_s(I_g[x, y], I_p[\hat{x}, \hat{y}]^{\gamma_p})^{\gamma_c} + \eta \tag{4.1}$$

où I_g représente la contribution totale de toutes les illuminations indirectes vers le point de la scène vu par le pixel de caméra x, y. L'illumination directe vient directement des intensités I_p du projecteur ; l'illumination indirecte I_g vient aussi du projecteur mais après une ou plusieurs interactions avec la scène (voir la figure 4.3). La modélisation ci-dessus peut être plus complexe si elle tient compte notamment du contraste et de la luminosité du projecteur, et du temps d'exposition de la caméra. De plus, si chaque canal d'une image couleur contient un motif, la modélisation doit tenir compte de la projection, de la réflectance et de la capture pour chaque canal de couleur [14]. En somme, les résultats produits par les systèmes de lumière structurée dépendent en grande partie de la scène, de l'équipement photométrique (caméra et projecteur) et de l'illumination indirecte.

Les codes de Gray (une variante des codes binaires)

Plusieurs méthodes utilisent seulement des motifs composés de pixels blancs ou noirs afin d'éviter l'estimation précise des intensités I_p à partir des mesures I_c. Ces méthodes évitent donc l'estimation des paramètres de l'équation 4.1. La figure 4.4(a) en montre un premier exemple, où cinq motifs projetés successivement identifient 32 pixels d'un projecteur selon leur position, autrement dit selon leur code binaire de 0 à 31 dans l'ordre (*i.e.* 00000, 00001, 00010, ...). Dans un motif, un pixel blanc représente le bit 1, et un pixel noir, le bit 0. Chaque motif ajoute un bit par pixel du projecteur. Par conséquent, un nombre n de motifs peut représenter 2^n pixels.

Pour retrouver un bit dans un pixel de caméra, il suffit d'évaluer si sa valeur correspond à blanc ou noir. Ceci peut être fait par l'utilisation d'un seuil sur l'intensité observée, ou par la projection des motifs inverses, ce qui doublerait le nombre de motifs à la figure 4.4. Dans l'image d'un motif, un pixel de caméra est considéré



(a) codes binaires

(b) codes de Gray

Figure 4.4. (a) Cinq motifs identifient par un code binaire 32 pixels d'un projecteur. Une rangée représente un motif, et une colonne représente un code binaire. Chaque motif ajoute un bit à tous les codes. (b) Les codes de Gray changent la position des codes binaires pour limiter à un bit la différence entre les codes voisins.

blanc si la soustraction entre sa valeur et celle dans l'image du motif inverse donne une intensité positive. Par exemple, l'inverse du motif qui correspond à la première rangée de la figure 4.4(a) devient blanc dans sa moitié gauche et noir dans sa moitié droite. La différence entre le motif original et son inverse devrait être négative dans la moitié gauche et positive dans la moitié droite.

Il est à noter que deux codes voisins très différents (par exemple les codes 15 et 16 dans la figure 4.4(a) n'ont aucun bit en commun) sont peu robustes si les pixels de la caméra ne sont pas parfaitement alignés avec les pixels du projecteur. Dans ce cas, un pixel de la caméra observe une portion de deux codes, et les transitions d'intensité apparaissent en gris au lieu de blanc ou noir. Pour que la récupération du code soit correcte, il faut que la même position soit choisie dans les cinq motifs, par exemple toujours le pixel 15 ou toujours le pixel 16. Sinon, le code résultant peut être complètement erroné. Même si ce pixel de caméra couvre davantage un des deux codes, l'illumination indirecte, entre autres, rend cette récupération très

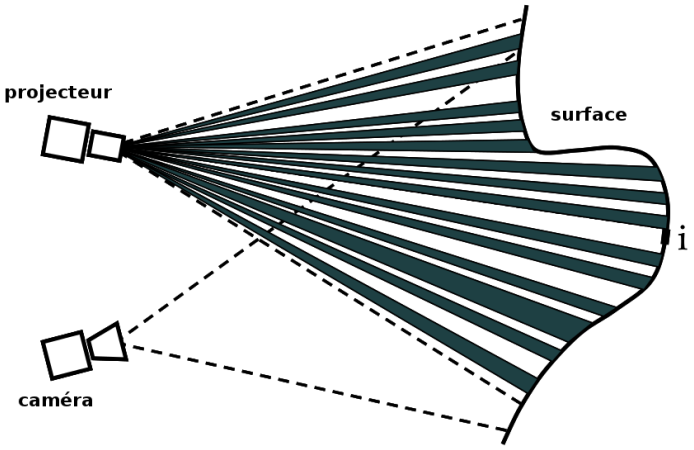


Figure 4.5. Lorsque le projecteur projette un motif haute fréquence, une unité de surface i reçoit une quantité égale de noir et de blanc peu importe la phase du signal. L'illumination indirecte tend alors vers une constante. Tirée de [46] (traduction libre).

incertaine puisqu'elle fait varier l'intensité mesurée à chaque motif. Les codes de Gray [35] montrés à la figure 4.4(b) sont une solution à ce problème. Ils changent la position des codes binaires pour limiter à un bit la différence entre les codes voisins, et donc stabilise la récupération des codes puisque les transitions d'intensité sont au minimum.

Cependant, les codes binaires et de Gray utilisent des motifs allant de basse à haute fréquences. Les motifs basse fréquence, caractérisés par de larges bandes, forment des sources de lumière susceptibles de produire de l'illumination indirecte pouvant causer des erreurs lors de la récupération des codes dans la caméra. Par exemple, l'illumination indirecte peut faire changer le signe de la différence entre un motif et son inverse (voir le chapitre 5 pour plus de détails). De plus, les interréflexions s'accentuent lorsque la surface est très réfléchissante, comme un écran omnistéréo qui permet une projection par polarisation.

La méthode du déphasage

Nayar *et al.* ont observé dans [46] qu'un motif haute fréquence rend l'illumination indirecte (y compris les interréflexions) constante, en ce sens que chaque unité de surface reçoit une quantité plus ou moins égale de noir et de blanc peu importe l'alignement (la phase) du signal, et que l'illumination indirecte produit un gris constant (voir figure 4.5). Nous présentons ici la méthode de [80] basée sur ce principe.

Soit trois signaux sinusoïdaux horizontaux :

$$
\begin{aligned}
I_{1_p}[\hat{x}] &= I_{moy} + I_{magn}\cos(\phi(\hat{x}) - \frac{2\pi}{3}) \\
I_{2_p}[\hat{x}] &= I_{moy} + I_{magn}\cos(\phi(\hat{x})) \\
I_{3_p}[\hat{x}] &= I_{moy} + I_{magn}\cos(\phi(\hat{x}) + \frac{2\pi}{3})
\end{aligned}
$$

où la phase $\phi(\hat{x})$ dépend de la fréquence du signal, et où I_{moy} et I_{magn} sont respectivement l'intensité moyenne du signal et la magnitude de la modulation. Chaque signal est déphasé l'un par rapport à l'autre d'un tiers de période, c'est-à-dire un intervalle de $\frac{2\pi}{3}$ pour une période de 2π. La figure 4.6 montre ces trois signaux haute fréquence I_{1_p}, I_{2_p} et I_{3_p}. En ne considérant pour l'instant que les illuminations directe et indirecte, l'équation 4.1 devient pour ces trois signaux :

$$I_{1_c}[x, y] = I_g + I_{1_p}[\hat{x}]$$

$$I_{2_c}[x, y] = I_g + I_{2_p}[\hat{x}]$$

$$I_{3_c}[x, y] = I_g + I_{3_p}[\hat{x}]$$

Pour un pixel donné de caméra, l'illumination indirecte I_g est constante pour les trois signaux. Elle s'annule donc lors de la soustraction de deux intensités mesurées, $I_{1_c}[x, y] - I_{2_c}[x, y]$ par exemple. Ainsi, la position du pixel dans le projecteur, c'est-à-dire la phase ϕ, peut être directement estimée à partir des intensités mesurées

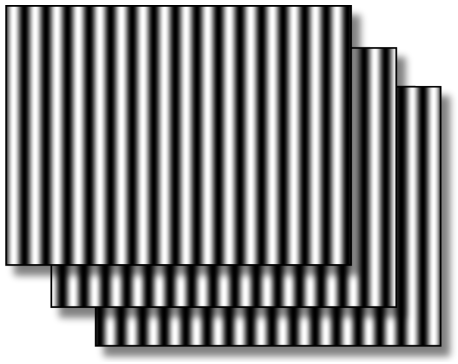


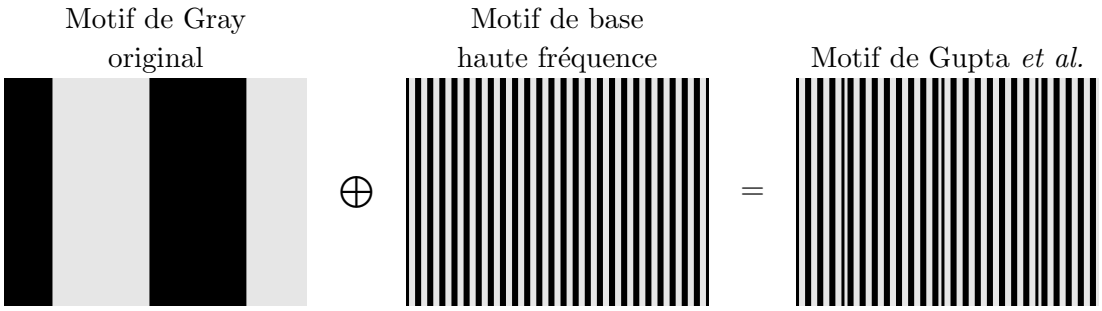
Figure 4.6. Un exemple de méthode par déphasage qui utilise trois signaux sinusoïdaux.

par la caméra sans être affectée par l'illumination indirecte [80] :

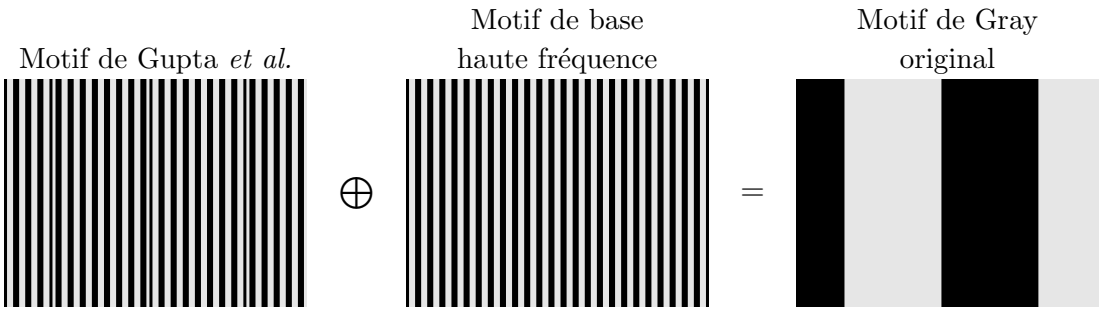
$$\phi(\hat{x}) = \arctan(\sqrt{3}\frac{I_{1_c}[x,y] - I_{3_c}[x,y]}{2I_{2_c}[x,y] - I_{1_c}[x,y] - I_{3_c}[x,y]})$$

Cependant, les paramètres γ_c, γ_p, η et Φ_s ne peuvent être ignorés en pratique. Autrement dit, nous ne pouvons utiliser ce type de méthode sans une modélisation photométrique très précise de la caméra, du projecteur, et aussi de la réflectance de ou des surfaces (voir l'équation 4.1). Ceci est particulièrement important pour les écrans omnistéréo qui ont une surface spéculaire. Ce type de surface réfléchit la lumière comme un miroir, ce qui fait en sorte que la luminosité mesurée d'un point de la scène change en fonction de la position de la caméra.

De plus, ces motifs sont ambigus en raison de leur caractère périodique, la phase n'indiquant la position que dans une des périodes du signal. Le calcul de la phase est donc habituellement suivi par une étape appelée déroulement de la phase (*phase unwrapping*) qui tente de retrouver la période où se situe chaque pixel de caméra. Cette étape suppose que toutes les périodes du signal sont présentes, dans l'ordre, sur la surface. Mais elle devient problématique s'il y a discontinuité de surface qui entraîne



(a) Processus d'encodage utilisé par Gupta *et al.*



(b) Processus de décodage utilisé par Gupta *et al.*

Figure 4.7. La méthode de Gupta *et al.* encode et décode des motifs de Gray à l'aide de l'opération *ou exclusif* (⊕) avec un motif de base haute fréquence.

un saut d'une période ou plus entre deux pixels voisins de caméra, par exemple lorsqu'il y a une ouverture dans l'écran cylindrique pour l'entrée des spectateurs.

La méthode de Gupta et al.

Gupta *et al.* [28] ont récemment présenté des motifs qui ont trois caractéristiques : ils sont haute fréquence, non ambigus, et composés uniquement de blanc et de noir éliminant la nécessité d'estimer les paramètres de l'équation 4.1. Ces auteurs utilisent principalement l'opération *ou exclusif* [3] pour encoder les motifs de Gray en motifs

[3] Le *ou exclusif* prend deux bits en entrée et renvoie la valeur 1 s'ils sont différents, et 0 s'ils sont identiques.

haute fréquence à l'aide d'un motif de base haute fréquence choisi. Cet encodage rend haute fréquence tous les motifs et, par conséquent, l'illumination indirecte devient constante pour un pixel de caméra. Une fois projetés, les motifs de Gray originaux peuvent être récupérés à l'aide du même motif de base (qui doit lui aussi avoir été projeté). La figure 4.7 illustre ce procédé d'encodage et de décodage. Pour déterminer si chaque pixel de caméra est blanc ou noir, on peut utiliser la soustraction d'un motif et son inverse, tel que décrit précédemment. De plus, cette soustraction annule l'effet de l'illumination indirecte, puisque celle-ci est constante.

Les motifs résultants de l'opération *ou exclusif* ne maintiennent pas pour tous les pixels la propriété de base des codes de Gray, à savoir qu'un code et ses voisins ne diffèrent que d'un bit. Comme nous l'avons mentionné plus haut, la récupération des codes devient alors moins robuste. Par exemple, Gupta *et al.* ont généré deux ensembles de codes en choisissant comme motif de base les deux plus hautes fréquences des motifs de Gray. Les codes résultants, appelés *XOR-2* et *XOR-4* (le chiffre fait référence à la largeur maximale des bandes), ne maintiennent pas la propriété de Gray respectivement pour la moitié et le quart des pixels, c'est-à-dire là où il y a transition noir/blanc ou blanc/noir dans le motif de base. Cependant, ces derniers ne sont pas situés aux mêmes pixels dans les deux ensembles, et l'utilisation d'un système de vote permet de réduire leur instabilité. Ce vote prend en compte les codes de Gray, les codes *XOR-2* et *XOR-4*, ainsi que les codes appelés *min-SW*. Ces derniers maintiennent la propriété des codes de Gray tout en limitant les largeurs de bande entre 8 et 32 pixels inclusivement, ce qui les rend du même coup plus robustes à l'illumination indirecte que les codes de Gray, mais moins robustes que les codes *XOR-2* et *XOR-4*. Une correspondance est considérée bonne si elle a obtenu le vote d'au moins deux ensembles de codes.

En somme, cette section a présenté trois méthodes de lumière structurée. La figure 4.8 les compare au niveau des fréquences utilisées. Nous présenterons dans le chapitre qui suit une nouvelle méthode qui utilise des motifs non structurés indépendants de la

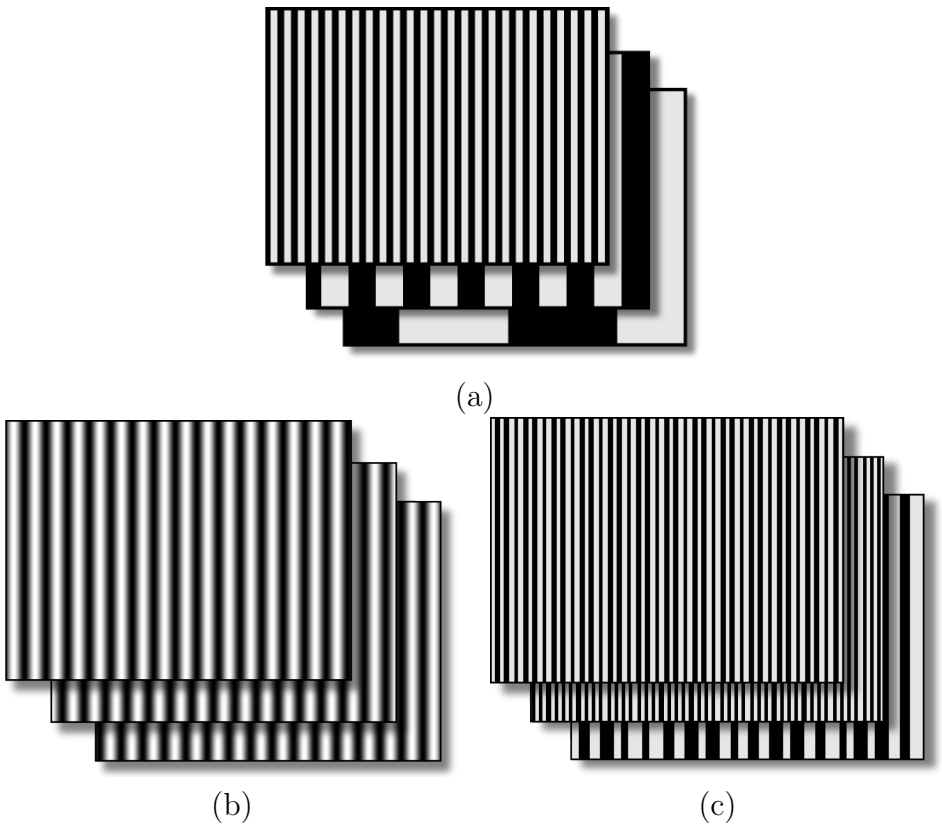


Figure 4.8. Alors que les codes de Gray (a) utilisent des motifs basse et haute fréquence pour identifier uniquement les pixels d'un projecteur, la méthode par déphasage (b) et la méthode de Gupta *et al.* (c) utilisent seulement des motifs haute fréquence afin de réduire l'illumination indirecte.

position des pixels du projecteur. Bien qu'un grand nombre de motifs soit nécessaire pour s'assurer que chaque pixel est représenté par un code unique, cette méthode a l'avantage de limiter la taille des régions blanches et noires, et ainsi de minimiser l'effet de l'illumination indirecte dans la scène. Chaque motif est généré en appliquant l'inverse de la transformée de Fourier discrète à un intervalle de fréquences aléatoires dans un espace à deux dimensions (2D). Alors que la transformée de Fourier discrète permet de retrouver la représentation spectrale (les fréquences de base) d'un signal analogique à partir d'échantillons, la transformée inverse permet de retrouver un signal analogique à partir de sa représentation spectrale.

Nous verrons que notre méthode, qui ne nécessite aucun calibrage photométrique, est également robuste aux défis typiques qui peuvent survenir dans les systèmes de reconstruction par lumière structurée, comme les discontinuités de profondeur dans la scène qui font en sorte qu'un pixel de caméra observe une portion de deux codes *non* voisins. Nous comparerons les résultats de notre méthode avec ceux des trois méthodes que nous venons de présenter, et nous montrerons qu'elle est plus robuste à l'illumination indirecte et aux discontinuités de profondeur. Nous montrerons aussi des résultats sous forme de reconstruction 3D de la scène. Une telle reconstruction, qui retrouve par triangulation la position des points de la scène à partir de correspondances caméra-projecteur, nécessite un calibrage des paramètres internes et externes de la caméra et du projecteur. Pour ce faire, nous utilisons des notions de vision par ordinateur décrites à la section 1.4 : le modèle de caméra perspective modélise la caméra et le projecteur, et l'estimation de leurs paramètres se fait par un ajustement de faisceaux à partir des correspondances données par la lumière structurée.

Chapitre 5

UNSTRUCTURED LIGHT SCANNING ROBUST TO INDIRECT ILLUMINATION AND DEPTH DISCONTINUITIES (ARTICLE)

Ce chapitre présente l'article suivant :

V. Couture, N. Martin, S. Roy. *Unstructured Light Scanning Robust to Indirect Illumination and Depth Discontinuities*. International Journal of Computer Vision(IJCV), Springer, soumis en décembre 2011.

Cet article s'intéresse au problème d'illumination indirecte qui affecte les procédés de reconstruction active par lumière structurée. La solution généralement proposée est l'utilisation de motifs structurés haute fréquence pour limiter la taille de leurs régions blanches ou noires [46]. Cependant, certains motifs sont peu robustes aux discontinuités de profondeur de la scène en raison de leur périodicité [76], et d'autres sont peu robustes aux erreurs de capture par caméra en raison de leur perte de similarité locale [28]. Nous proposons plutôt l'utilisation de motifs non structurés. Ce type de motifs n'est pas nouveau [19, 39, 73], mais nous sommes les premiers à les utiliser pour réduire les interréflexions. De plus, la méthode est indépendante des propriétés photométriques du projecteur et de la caméra, tout en étant robuste aux discontinuités de profondeur. Plus précisément, la méthode génère un signal aléatoire (bruit blanc) dans le domaine des fréquences. Un filtrage passe-bande permet ensuite d'éliminer les basses et les très hautes fréquences. Comme ces motifs non structurés n'encodent pas directement la position des pixels d'un projecteur, il faut un algorithme de recherche de grande dimension pour *trouver* la position du code observé. Nous comparons nos résultats avec ceux de quelques méthodes existantes.

Nous présentons ici l'article dans sa version originale.

Abstract

Reconstruction from structured light can be greatly affected by indirect illumination such as interreflections between surfaces in the scene and sub-surface scattering. This paper introduces band-pass white noise patterns designed specifically to reduce the effects of indirect illumination, and still be robust to standard challenges in scanning systems such as scene depth discontinuities, defocus and low camera-projector pixel ratio. While this approach uses *unstructured* light patterns that increase the number of required projected images, it is to our knowledge the first method that is able to recover scene disparities in the presence of both indirect illumination and scene discontinuities. Furthermore, the method does not require calibration (geometric nor photometric) or post-processing such as phase unwrapping or interpolation from sparse correspondences. We show results for a few challenging scenes and compare them to correspondences obtained with the well-known Gray code and Phase-shift methods, and with the recently introduced method by Gupta *et al.*, designed specifically to handle indirect illumination.

5.1 Introduction

Scene reconstruction from structured light is the process of projecting a known pattern onto a scene, and using a camera to observe the deformation of the pattern to calculate surface information. The classification of having structure comes from the fact that a unique code (a finite set of patterns) is associated to each projector pixel, based on its position in the pattern. Camera-projector pixel correspondence (see Fig. 5.1) can then directly be established and triangulated to estimate scene depths. Results produced by structured light scanning systems greatly depend on the scene and the patterns used. In particular, it was shown in [46] that low frequency patterns create interreflections in scene concavities that cannot be removed. Another issue comes

from scene depth discontinuities, where smoothness of the observed pattern can no longer be assumed.

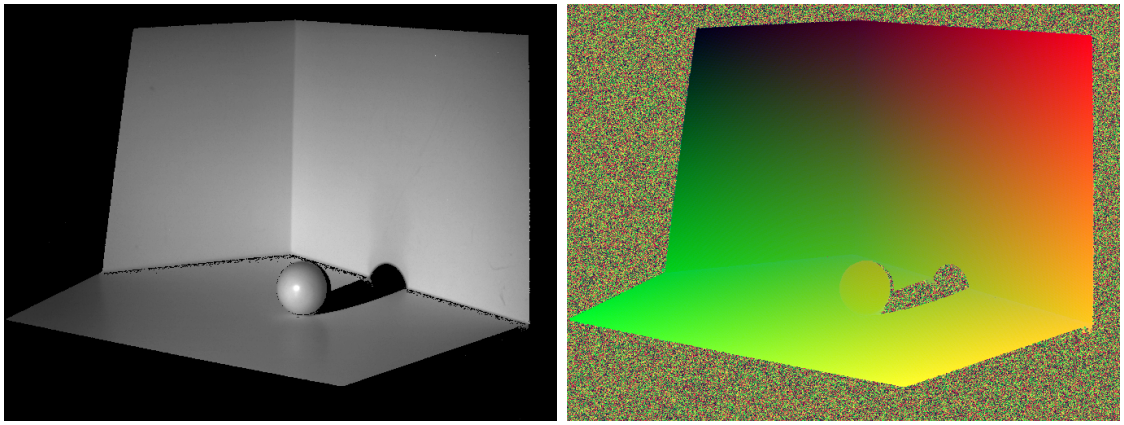


Figure 5.1. Example of a scene (left) and its correspondence map (right). Red and green are used for x and y coordinates respectively.

In this paper, we propose the use of band-pass white noise patterns that are specifically designed to reduce the effects of indirect illumination[1] while still being able to handle depth discontinuities. These patterns follow the basic idea of *unstructured* light patterns [19, 39, 73] that are not related to pixel position in the projector. Their only restriction is that the accumulation of such patterns uniquely identifies every projector pixels. Therefore, the correspondence of a camera pixel is no longer computed directly from the observed pattern sequence, and has to be found using an iterative high-dimensional matching algorithm. The matching method we present here is not limited to epipolar lines to avoid the need to geometrically calibrate any of the device in order to recover correspondence.

[1] In the literature, indirect illumination is sometimes called *global* illumination.

The spatial frequency of these patterns can be adjusted, making them robust to defocus (due to small depth of field, for instance) or low camera-projector pixel ratio[2]. Also, the method is designed to be independent of photometric properties (such as gamma correction) of both the projector and the camera.

The method was first presented in [16] specifically to address the problem of interreflections. Here, we include new results to show that the method also works for other types of indirect illumination such as translucency and sub-surface scattering. We also compare our results with those of other methods, namely the Gray code and Phase-shift methods, and a recently introduced method by Gupta *et al.* [28] to handle indirect illumination.

The layout of this paper is as follows. We begin in Sec. 5.2 by briefly reviewing prior works related to structured light patterns. We then expose in Sec. 5.3 common problems that may arise in structured light setups, namely indirect illumination, scene depth discontinuities and a low camera-projector pixel ratio. In Sec. 5.4, we introduce unstructured band-pass white noise patterns and discuss their properties. Using these patterns, matching between projector and camera pixels requires a high-dimensional match algorithm, namely locally sensitive hashing, which we describe in Sec. 5.5. In Sec. 5.6, the method of Gupta *et al.* [28] that also handles indirect illumination is reviewed. Finally, we compute in Sec. 5.7 camera-projector correspondence maps and reconstructions using our unstructured light patterns and compare results produced by other methods for different challenging scenes. We conclude in Sec. 5.8.

5.2 Previous work

Several sets of structured light patterns were previously proposed to perform active 3D surface reconstruction. Structured light reconstruction are often classified based on the type of encoding used in the patterns: temporal, spatial or direct [60]. Here,

[2] The camera-projector pixel ratio is defined as one camera pixel over the number of projector pixels it can see.

we also emphasize the amount of supplemental information needed by the method to work effectively. For instance, prior photometric or geometric calibration is often required.

Temporal methods multiplex codes into pattern sequence [27, 36, 53, 61]. For instance, a pixel position is encoded in [53] by its binary code, representated by a concatenation of binary coded patterns. One variation introduces Gray code patterns [36] that are designed to minimize the effect of bit errors by ensuring that neighboring pixels have a code difference of only one bit. Temporal methods require a high number of patterns and the scene must remain static during the pattern acquisition process. In practice, these methods can give very good results and do not require any kind of calibration. Due to focus issues or low pixel ratio, the lowest significant bits often cannot be recovered. Solutions have been proposed, like in [27] where high frequency patterns are replaced by a shifted version of a pattern to recover the last significant bits. This method (and all variants of binary encoding patterns) also suffers from the significant indirect lighting induced by the lower frequency patterns, as we will see in the next section.

In contrast, spatial methods use the neighborhood of a pixel to recover its code [12, 59, 71] in order to decrease the number of required patterns. For example, the patterns can be stripes [12], grids [55] or a more complicated encoding such as the popular De Bruijn patterns [71]. Except for grids, it is worth mentioning that these patterns are one-dimensional, and thus require a geometric calibration relating the camera and the projector. Some methods even allow "one-shot" calibration [59] (i.e. only one pattern is used), but they require a very good photometric calibration. The main drawback of these methods is that they assume spatial continuity of the scene, which does not hold at depth discontinuities. Furthermore, those methods produce sparse results, as the correspondence can be recovered only at stripe transitions of the pattern. In [78], high quality reconstructions of static scenes are computed using a multi-pass dynamic programming edge matching algorithm. The pattern is shifted

over time to compensate for the sparseness of De Bruijn patterns. The number of patterns required is still a lot less than in the case of temporal methods. However, the method requires both photometric and geometric calibration.

Direct coding methods use the intensity measured by the camera to directly estimate the corresponding projector pixel. Similarly to temporal methods, no spatial neighborhood is required to obtain correspondence. Direct methods need only a few patterns, typically three patterns. Because patterns can be embedded in a single color image, one image is theoretically sufficient to recover depth. The work of [76] introduced the so-called "three phase-shift" method which relies on the projection of three dephased sinusoidal patterns. This method was modified in [79] to project only two sinusoidal patterns and a neutral image used as a texture. These methods often require the estimation of the gamma coefficient (for both the projector and the camera) and, because they are one-dimensional, a geometric calibration as well. Furthermore, matching using these patterns is ambiguous due to their periodic nature. In practice, phase unwrapping is used to overcome this issue, but high frequency patterns remain ambiguous for scenes with large depth discontinuities.

We present in Sec. 5.4 a novel temporal method that use *unstructured* light patterns that are not dependent on projector pixel position. Similar work has been presented in [39] where scanning is performed using a sequence of photographs or a sequence of random noise patterns for flexibility purposes. Contrary to [39] however, we designed the unstructured patterns specifically to minimize the effects of indirect illumination. Another method was recently introduced in [28] to address the problem of indirect illumination using a combination of high frequency patterns, band-pass patterns and standard Gray codes. We will compare this method with our approach in Sec. 5.6. Our method will also address typical challenges that may arise in structured light setups. We review these in the following section.

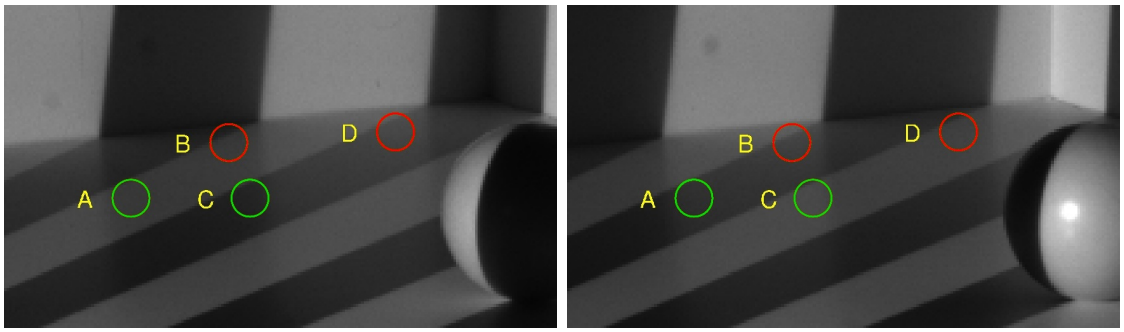


Figure 5.2. A stripe pattern (left) and its inverse (right) are displayed. Measured intensities at points (A, B, C, D) are $(56, 56, 35, 71)$ and $(46, 66, 72, 65)$ in the left and right images respectively. Points B and D are incorrectly classified because of interreflection.

5.3 Problems of structured light systems

This section reviews the problems that may arise in typical structured light setups, such as indirect lighting, varying camera-projector pixel ratios, and scene depth discontinuities. It also discusses strengths and weaknesses of the methods reviewed in Sec. 5.2.

5.3.1 Indirect illumination

When a scene is lit, the radiance measured by the camera has two components, namely direct illumination due to direct lighting from the projector and indirect illumination caused by light reflected from or scattered by other points in the scene for instance[46]. It is generally assumed that when projecting a Gray code pattern followed by its inverse, a camera pixel is lighter when observing a white stripe [60]. This is not always the case however, especially in the presence of indirect illumination, as illustrated in Figure 5.2 by points B and D. This situation severely deteriorates the quality of the recovered codes.

Nayar *et al.* presented in [46] a method to separate direct and indirect components of illumination. They showed that indirect illumination becomes a constant gray intensity when the pattern frequency is high enough, i.e. that geometry, reflectance map and direct illumination are smooth with respect to the frequency of the illumination pattern. Separation is done by subtracting the image of a single high frequency binary pattern and its complement, or by subtracting the minimum from the maximum intensities measured over a few patterns.

Structured light methods that use only high frequency patterns could potentially remove the effects of indirect lighting to improve performance. Phase-shift methods are good examples, but increasing the frequency also increases signal periodicity, which makes the subsequent phase unwrapping step hard if not impossible to accomplish. Therefore, lower frequency patterns tend to be used in practice [60].

Removing the effects of indirect illumination is not possible for low significant bits patterns of the Gray code method. For low frequency patterns, indirect lighting must be estimated using a light transport matrix [26, 41, 47] which relates every pixel of the projector to every pixel of the camera. However, this matrix is huge and very time consuming to measure and process. For illustration purposes, we computed this matrix, which was then transposed and remapped from projector to camera using our matching results. Figure 5.3 shows how different regions in the scene contribute to the intensity measured at selected camera pixels by creating indirect lighting. As in [46], we argue that if the pattern spatial frequency is high enough, then these contributing areas always include an equal mixture of black and white, thereby making indirect lighting near constant.

5.3.2 Depth discontinuities

Spatial methods such as De Bruijn patterns require a neighborhood around a pixel to estimate its code. This allows a reduction in the number of patterns, but creates problems near depth discontinuities where the camera observes a mixture of at least

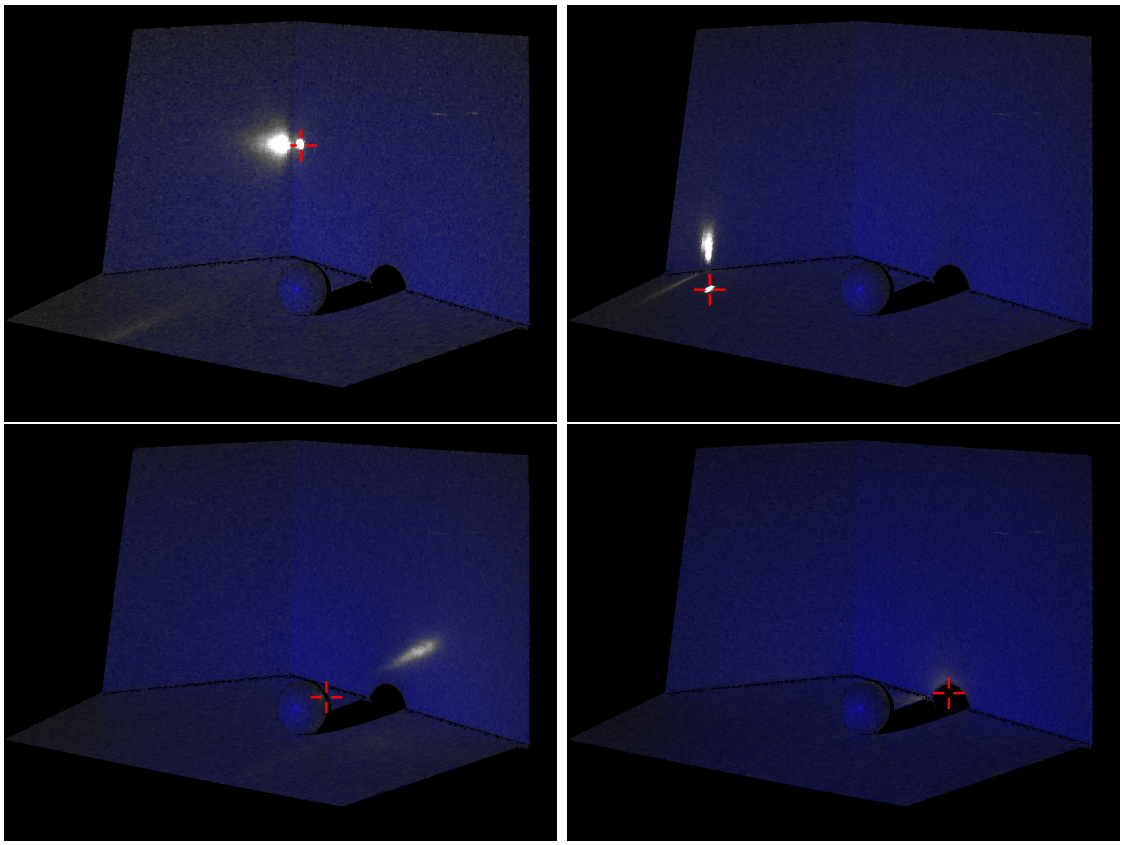


Figure 5.3. Illumination contribution for selected pixels, indicated by red crosshairs. The blue color is added artificially to provide a scene reference. Top has direct lighting with interreflections. Bottom left features indirect lighting. Bottom right is a pure shadow.

two projector pattern regions. This makes decoding unstable. For this reason, spatial methods require a post-processing step to remove wrong matches near discontinuities, usually a dynamic-programming minimization to add smoothness constraints on the correspondence map [78].

For temporal and direct methods, which do not require any spatial neighborhood, correspondence errors can occur when two codes at different depths are both seen by the same camera pixel. This blends two unrelated codes and affects direct methods such as Phase-shift which rely on the measured intensity to estimate correspondence.

5.3.3 Pixel Ratio

Because of the relative geometry and resolution of the camera and projector, it is often the case that a single camera pixel captures a linear combination of two or more adjacent projector pixels. This situation often occurs in multi-projector setups, where the total resolution of the projectors is far greater than the camera resolution. This is known as having a low camera-projector pixel ratio.

The Gray code method degrades gracefully with pixel ratio, as low significant bits become too blurred to be recovered and are simply discarded. Other methods, such as De Bruijn or Phase-shift, are robust to this as long as their pattern frequencies are low enough.

5.4 Unstructured light patterns

This section presents our *unstructured light* method, featuring band-pass white noise patterns that are designed to be robust to indirect illumination by avoiding large black or white pattern regions.

In this paper, we consider surfaces that are mostly diffuse. If we can make one full period of our pattern smaller than the diffusion, then the effect of this diffusion is near constant for any pattern with the same frequency [46].

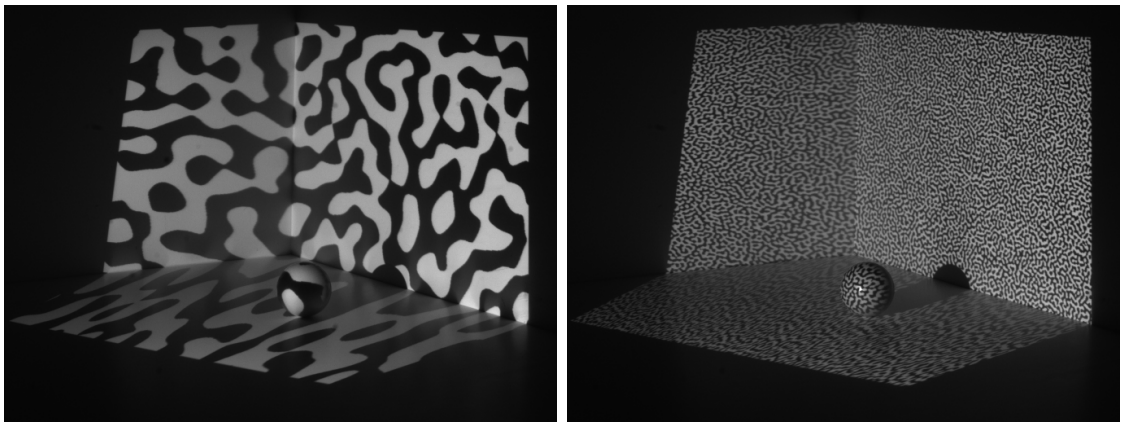


Figure 5.4. Synthetic patterns are generated in the Fourier domain by randomizing phase within an octave. Here, two patterns are shown projected on a scene. Spatial frequencies used are (left) 8 to 16 cycles per frame and (right) 64 to 128 cycles per frame.

We limit the amplitude spectrum to a single octave, ranging from frequency f to $2f$, where a frequency refers to the number of cycles per frame. For each spatial frequency, the amplitude is set to 1 and the phase is randomized, subject to the conjugacy constraint [13], namely that $\hat{I}(f_x, f_y) = \overline{\hat{I}(-f_x, -f_y)}$.

The second step is to take the inverse 2D Fourier transform of $\hat{I}(f_x, f_y)$, yielding a periodic pattern image $I(x, y)$. To avoid periodicity, we generate a pattern larger than the desired width (say 110% larger) and then cut the extra borders. The pattern intensities are then rescaled to have values ranging in [0:255]. Each pattern is finally binarized with a threshold at intensity 127 to make pixels either black (≤ 127) or white (> 127).

Hence, the patterns are parametrized by frequency f and limited to a single octave of variation to control the amount of spatial correlation (see Fig. 5.4). More spatial correlation increases code similarity locally, but also increases the number of required patterns to guaranty code uniqueness. We next discuss these two aspects.

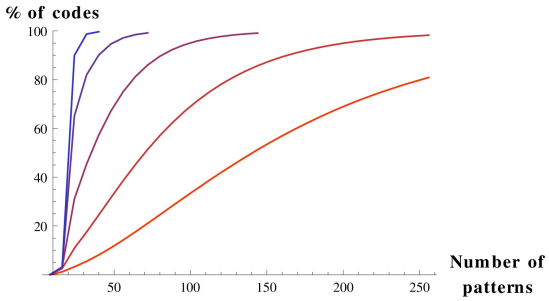


Figure 5.5. For HD images (1920 × 1080 pixel resolution), the percentage of pixels having unique codes while increasing the number of patterns. The curves correspond to f ranging from 8 to 128, with steep curves corresponding to patterns of higher frequencies. Curves stop being drawn if they reached 99%.

5.4.1 Reducing code ambiguity

In this section, we analyze the relationship between frequency f and the number of patterns required to identify projector pixels uniquely with a code sequence of black and white values. Note that the pattern sequence is uncorrelated temporally to ensure that all bits in a code are independent.

In Fig. 5.5, we measure the number of patterns required to disambiguate at least 99% of all pixels as frequency f is varied. We consider HD projectors having 1920 × 1080 pixels. One can see that low frequency noise requires more patterns. Moreover, low frequency patterns often cause interreflections when large white pattern regions are projected in surface concavities and/or highly reflective materials.

Finally, we observe that this 1% of code duplicates usually correspond to small groups of neighboring pixels that are yet to be disambiguated. High frequency patterns, however, tend to quickly produce unique codes locally but have duplicates elsewhere.

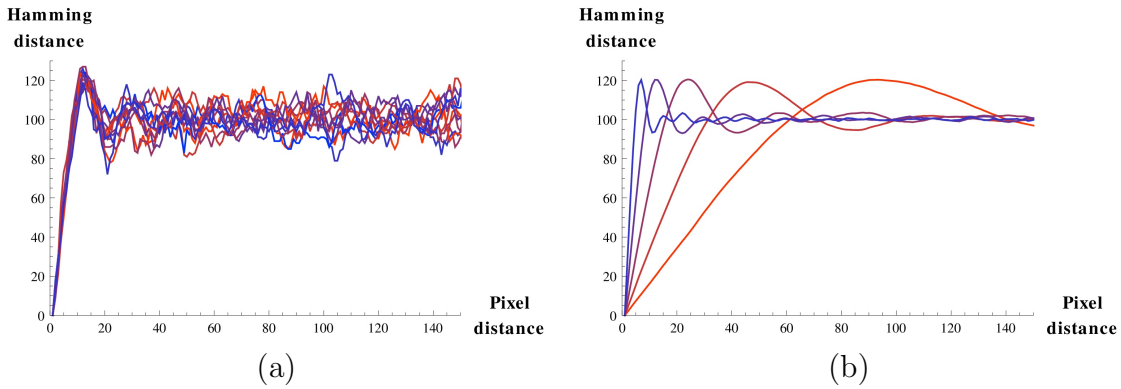


Figure 5.6. Hamming distance between a randomly selected pixel and its neighbors with increasing distance, for a code length of 200 patterns. Distances are shown (a) for a few selected pixels ($f = 64$) (b) as the average over many selected pixels for different frequencies f ranging from 8 to 128, with steeper curves corresponding to patterns of higher frequencies. Each curve follows a sharp increase before decreasing to a constant that is half the number of patterns. Patterns of higher frequencies are not as correlated spatially (steeper increase).

5.4.2 Keeping neighbors similar

One important property of our patterns is the similarity between neighboring codes. Fig. 5.6 presents the hamming code difference with respect to the distance between two neighboring pixels. Regardless of the frequency used, the hamming difference increases gradually with distance until it reaches a negatively correlated maximum before decreasing to a constant level. The standard deviation around this plateau is that of a Binomial distribution and is equal to about 7.07 bits, that is $\frac{\sqrt{N}}{2}$ for $N = 200$ bits.

This correlation between neighboring codes makes it easier for mismatch to happen between neighbors. However, it provides great robustness to pixel ratio variations, since the averaging of a group of neighboring codes is still highly correlated to each original blended codes. Also, this provides robustness to various local imaging problems like out of focus areas because of small depth of field.

Moreover, the lack of correlation between far pixels helps provide very high robustness to scene discontinuities. When a camera pixel observes a scene discontinuity, its intensity is a blend of two uncorrelated codes. Thus, about 50% of the bits are the same in both codes and will be accurately recovered. The remaining bits belong to either code, thereby ensuring that the matching code is composed of at least 75% of all bits of these two codes. This makes them and their neighbors much more likely to match than any other distant code. In contrast, if the recovered bits of two blended Gray code patterns are not all from the same code, then the resulting code may be completely unrelated to the two blended codes.

5.5 Establishing pixel correspondence

This section deals with efficiently establishing the correspondence between camera and projector pixels. We designed our matching method so that it does not require any form of prior calibration. This makes the matching more difficult but much more flexible. For example, the camera could be a non single view point fisheye and the projector illumination could be bouncing off a convex surface. These cases are common in multi-projection setups and are not easily calibrated.

A number of random unstructured light patterns are generated with a preselected band-pass frequency interval. Those patterns are projected one at a time while a camera observes the scene. N patterns are projected, captured by the camera, and then matched.

First, the gray images captured by the camera are converted into binary images for matching. The conversion is simply obtained by measuring if a pixel is above or below the average of previous patterns over time. Let $\Phi_{xy}(i)$ be a monotonic function modeling photometric distortion[3], the average image $\bar{I}_c$ in the camera, computed from all the distorted intensities in the camera, remains a good delimiter because it is well

[3] Photometric distortion includes gamma factors, scene albedo and aperture [14].

within Φ_{xy}(black) and Φ_{xy}(white) when, for a camera pixel, the amount of black and white values is reasonably balanced. Furthermore, the average works well because band-pass noise patterns should not produce big changes in indirect lighting.

Thus, as codes from *unstructured* light patterns no longer have any correlation to projector pixel position, pixel correspondences have to be found by matching two sets of high dimensional vectors to one another. Using N patterns, we obtain a N-dimensional binary vector for each pixel of both the camera and the projector image. For HD images, each set has around $1920 \times 1080 \approx 2$ million N-dimensional vectors. For the remainder of the section, we assume that camera pixels are matched to projector pixels, although matching can be performed the other way around (or even both ways simultaneously), which can be useful, for instance, in multi-projector systems [69] to remove the need to inverse the correspondence maps.

Efficient matching is achieved using a high-dimensional search method based on hashing of binary vectors as described in [5, 6, 23]. All vectors are hashed by selecting b-bits (hopefully noise free) out of the N code bits. We use a key size b that should cover at least the number S of pixels in the projector such that expected number of codes hashed by a single key is around 1. In practice, we use $b = \lceil \log S \rceil$. While the codes should ideally match exactly (i.e. have the same key), there is some level of noise in practice. Thus, the method proceeds in k iterations, and selects a different set of bits for each iteration.

For a given pixel, the probability P that it is matched correctly after k iterations, in other words, that its hashing key has no bit error, can be modeled as

$$P = 1 - (1 - (1 - \rho)^b)^k \tag{5.1}$$

where ρ is the probability that one bit is erroneous. The number of iterations required to get a match within confidence P can be computed as

$$k = \frac{\log(1-P)}{\log(1-(1-\rho)^b)} \quad . \tag{5.2}$$

Several factors can increase the ρ value such as very low contrast and aliasing which becomes worse for higher frequency patterns and lower camera-projector pixel ratios. Thus, ρ can vary locally in the camera image, as scene albedo may change contrast for parts of the scene only. The pixel ratio may also change, in the presence of slanted surfaces for instance. Estimating ρ would yield an indicator of how many iterations are required, given the desired probability of a correct match P. However, Sec. 5.5.1 will introduce heuristics that improve convergence and thus, make the number of iterations predicted by ρ very pessimistic. Other termination criteria are discussed in Sec. 5.5.2.

Fig. 5.8(a) shows how adding code errors affects the convergence. We generated $N = 200$ patterns and applied a noise according to various ρ values. For instance, the best match should have an average optimal error of 20 bits for $\rho = 0.1$. One can see that convergence is still achieved for $\rho \leq 0.1$, but that it becomes much slower for higher ρ values. Since the number of iterations grows exponentially with ρ, a value larger than about 0.3 will result in no convergence.

Matching heuristics (see Sec. 5.5.1) can improve convergence considerably (see Fig. 5.8(b)). However, optimal matches do not guaranty quality matches. For instance, when $\rho = 0.3$ is used, good matches give errors that are not well separated from random codes ($\rho = 0.5$), distributed at about half the number of bits $\frac{N}{2}$.

During an iteration, the hash table can be unbalanced, i.e. more that one code hashes in a single bin. The search for the closest code in each bin can increase significantly the matching time. In practice, the codes hashing to the same bin could be stored in a data structure accelerating the search. Instead, we select the first hashed code. Even if this strategy does not choose the best code, the time gained can be used to perform another matching iteration. Typically, the execution time for

one iteration on a laptop with an Intel dual core 2.2 Ghz CPU with 2GB of RAM is around one second when matching an HD camera to an HD projector, and the iteration time is doubled when applying the heuristics.

5.5.1 Matching heuristics

Usually, reconstruction methods take advantage of *a priori* knowledge about the scene in order to improve the results. One common assumption is that neighboring pixels have similar correspondences, thereby suggesting some form of local smoothing. Unfortunately, smoothing can introduce errors at discontinuities or wherever the assumption does not hold. In our case, we propose two simple heuristics that take advantage of scene smoothness to get a dramatic speedup in convergence. Their great advantage is that they improve the convergence time without any degradation of the final result.

The heuristics are illustrated in Fig. 5.7. *Forward matching* tests if a camera pixel can find a better match in the neighborhood of its current match in the projector. This heuristic refines matches that lie within the convergence area of the cost function (i.e. ≤ 15 pixels in Fig. 5.6(a)). *Backward matching* tests the neighbors of a camera pixel to check if they could also match its corresponding projector pixel. This heuristic tends to create new matches, i.e. improve matches that are outside the convergence area (> 15 pixels in Fig. 5.6(a)). The speedup is shown in Fig. 5.8, where the convergence is plotted as a function of the number of iterations needed with and without the use of the heuristics.

5.5.2 Match confidence and termination criteria

This section discusses a termination criteria to decide when to stop matching iterations. This is not a trivial problem due to the probabilistic nature of the

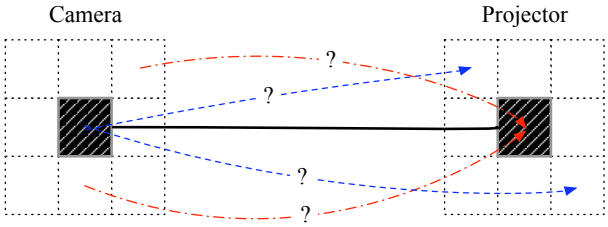


Figure 5.7. When a match is found (black solid line), two simple matching heuristics can be used : *forward matching* **(blue dashed lines) attempts to improve an existing match and** *backward matching* **(red dot-dashed lines) attempts to create neighborhood matches.**

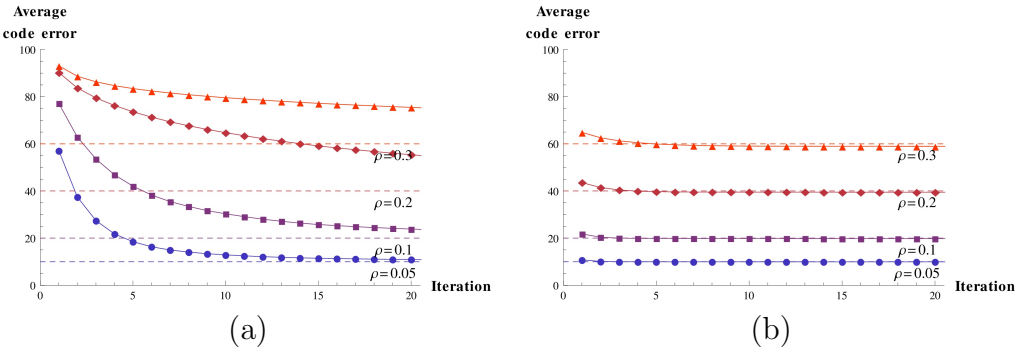


Figure 5.8. For increasing noise levels ρ, convergence of the hashing method (a) without heuristics (b) with heuristics. The dashed lines represent the theoretical lowest average code error. Convergence is much faster when applying the heuristics.

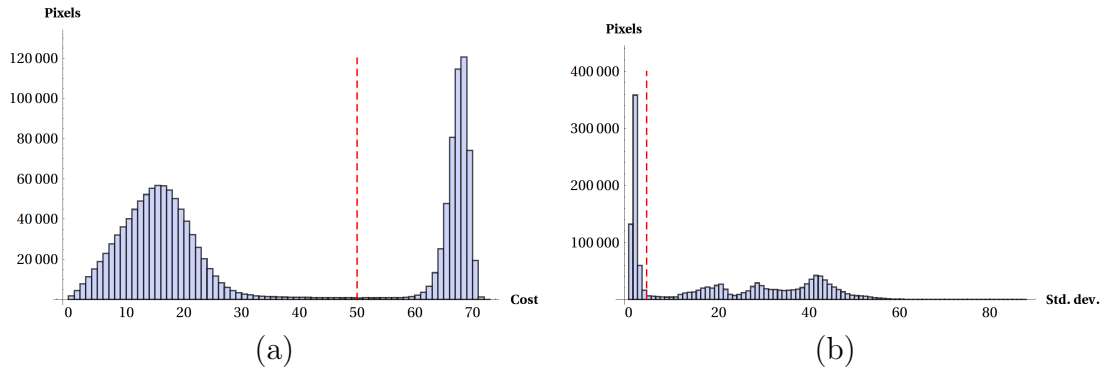


Figure 5.9. For a typical scene, (a) a histogram of match costs has two distributions centered at ρN and at a value a bit below $\frac{N}{2}$ (see text for details). (b) a histogram of standard deviation of intensities has a high peak corresponding to unlit camera pixels or low contrast regions. A threshold (indicated here by the red dashed line) cannot completely separate the long tails of the distributions.

algorithm. For instance, it can often happen that hashing improves a few matches even after there was no improvement for several iterations.

Camera pixels that see a surface area not directly illuminated by the projector should be excluded from the matching process because they produce random codes that depend on camera noise. The matching process would keep improving these matches, making a termination criteria more difficult to establish. Looking at the matching costs or standard deviations of intensity could be a good strategy to detect most of the unlit camera pixels. Fig.5.9(a) shows a histogram of the matching costs for a typical scene after 50 iterations. The matching costs are distributed in two well separated Binomial-like distributions, namely one centered at ρN and one centered below $\frac{N}{2}$ (in Fig.5.9, $N = 200$ and $\rho \approx 0.1$). The first distribution corresponds to correctly matched camera pixels. The second distribution corresponds to unlit pixels; its mean is lower than $\frac{N}{2}$, because only the minimum matching code is kept at each iteration. Fig.5.9(b) shows a histogram of the standard deviations of pixel intensities. The distribution is roughly bimodal, with the highest peak corresponding to mostly

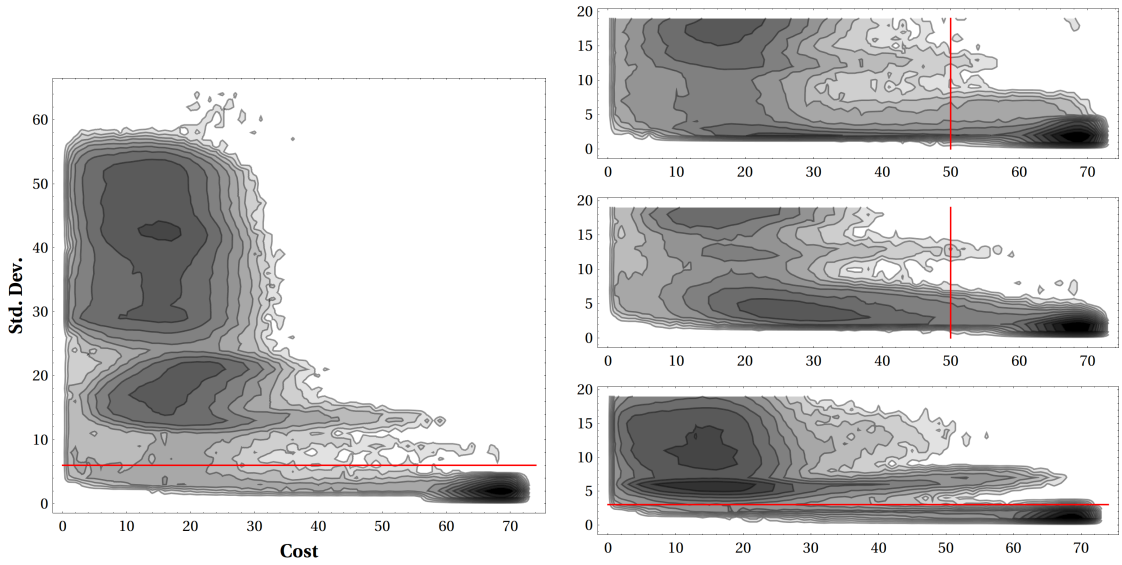


Figure 5.10. 2D log histograms of matching costs and standard deviations of intensity for the 4 scenes presented in the experimental results, namely (left) `Ball` **and (right, top to bottom)** `Games`, `Grapes & Peppers` **and** `Corner`. **The red lines show the thresholds to remove unlit camera pixels.**

unlit pixels. This narrow peak illustrates well the fact that all the patterns produce near constant indirect illumination for a given scene. Gray codes do not feature this property. The rest of the distribution is composed of lit pixels, modulated by the scene reflectance.

However, this peak also contains pixels corresponding to dark scene objects. Because of this ambiguity, we consider both criteria, as illustrated in Fig. 5.10. Because of the long tails of the distributions, there is usually no single threshold which can separate all good matches from wrong matches. For most scenes, either criteria works. For scenes with dark objects, saturated or noisy imaging conditions, one criteria might work better than the other. The red lines illustrates the thresholds we used for the different scenes. In practice, both criteria could be used at the same time.

Once the unlit camera pixels are discarded, we can iterate until only a small number of pixels are updated (say 5 pixels) for a few iterations (say 5 iterations). Very few match errors may remain, usually less than 0.01% of all pixels (20 or 30 pixels). These are typically located where strong interreflection remains, such as the intersection of two walls. There, the high code errors makes the heuristics inefficient. An exhaustive search is then performed for all matches that are not smooth with respect to their neighbors, in the hope of finding a better match. Smoothness for a camera pixel is simply checked by considering the average match of its neighbors, and verifying that it is within a threshold distance τ (we use τ=1.5). Note that this smoothness condition will also select all depth discontinuities as potential match errors, thereby subjecting them to an exhaustive search. This search is repeated until no further updates are made.

5.5.3 First results

In this section, we present the first results of our method on a real scene. The scene contains significant interreflections, depth discontinuities and out of focus regions. A more detailed comparison with other methods will be presented in Sec. 5.7.

The scene is composed of two walls, a floor and a ball (see Fig. 5.1). Our method gives good results for high and low significant bits of the correspondence map, as illustrated in Fig. 5.11. A frequency f of 128 cycles per image was used.

Furthermore, we tested our method over a range of unstructured pattern frequencies. The results for selected regions are shown in Fig. 5.12. Notice that for regions not lighted directly, random codes are expected. This is observed behind the ball (Fig. 5.12 (top right)). High-frequency patterns also improve matching on the floor near the wall.

Finally, using the best results of our method as a reference, we measured errors by varying pattern frequencies and the number of patterns used. Fig. 5.13 shows that errors are smaller with more patterns and middle frequencies. Low frequencies are

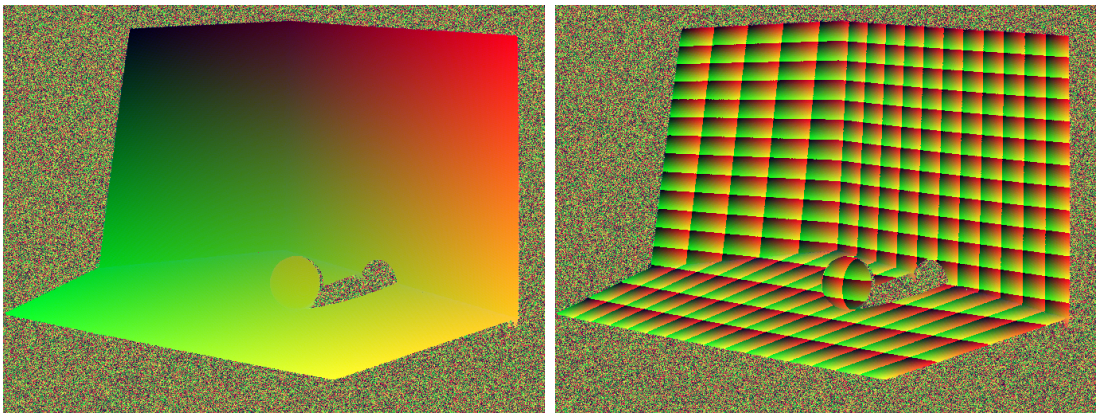


Figure 5.11. First results using unstructured light patterns. Results are shown with only the high significant bits of the correspondence map (left) and only the low significant bits (right). The correspondence map is color encoded, with red and green used for x and y coordinates respectively.

unsuitable to reduce the effects of indirect lighting, and more patterns are required to disambiguate codes locally. Very high frequencies (here 256) would be ideal to make indirect illumination near constant, assuming that the camera resolution is sufficiently high to resolve the signal. But this was not the case in our setup. The problem of indirect illumination has been reduced to a problem of camera aliasing.

5.6 Overview of the Gupta *et al.* method

This section compares our method to the method recently introduced in Gupta *et al.* [28] to address indirect illumination. Their method uses four set of codes, standard Gray codes and three other sets optimized for different illumination effects.

First, they address what they classify as long-range illumination (diffuse and specular interreflections) with the use of high-frequency patterns, generated by combining a chosen high-frequency base pattern with standard Gray codes through the XOR operation. From the captured images, the original Gray code patterns can be recovered by performing the XOR operation again with the same chosen pattern.

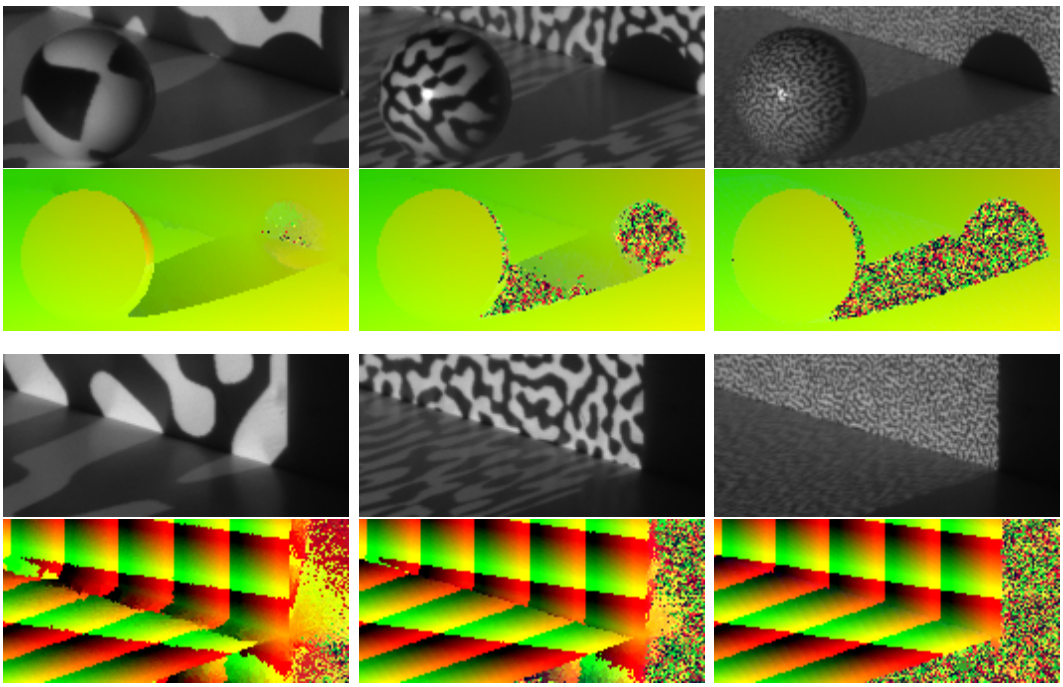


Figure 5.12. Correspondence from unstructured patterns at frequencies 8 (left), 32 (middle) and 128 (right). The effects of using higher frequency patterns are exposed on the edge of the ball and its shadow (top), and the corner of the walls (bottom).

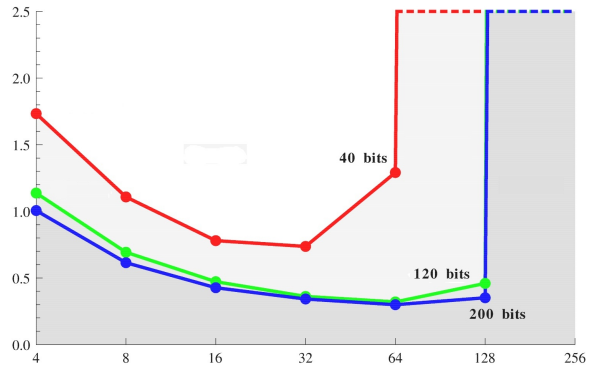


Figure 5.13. Average correspondence cost as a function of pattern frequency (4,8,...,256), for various code lengths (40,120 and 200 bits). Observe that more bits give lower errors. Low frequency patterns give slightly larger average errors because they required even more than 200 bits to disambiguate all pixels locally. High frequency patterns suffer from aliasing which makes convergence harder to achieve.

Although this pattern could be any high-frequency pattern, Gupta *et al.* use the two highest Gray code patterns to generate two sets of patterns, namely XOR-2 and XOR-4 patterns (2 and 4 correspond to the maximum stripe width in both sets). Note that this choice produces narrow but very long stripes, which is not the case in our patterns. Effects of indirect illumination could probably be reduced further by choosing a base pattern that limits the stripes in both directions.

Second, they address short-range effects (sub-surface scattering and defocus) that can severely blur the high-frequency patterns, leading to a lot of code errors during the binarization process. To avoid this, Gupta *et al.* use a set of patterns called min-SW Gray codes [25], featuring stripe widths between 8 and 32 respectively.

In [28], good correspondences are chosen if they match in at least two sets of codes. Otherwise, a camera pixel is flagged as an error. In our implementation of the method, we matched codes in x and y separately and we considered that two matches agreed if their pixel distance was less or equal to 2. Also, we used the recovered camera codes directly, without applying any filtering such as a median filter to remove

isolated noisy matches. Note that we did not address in this paper the iterative error correction process[28, 77] which captures additional patterns that include only unmatched projector pixels. While this process can be effective to decrease indirect illumination given a good error detection criteria, we argue that it should ideally not be required for robust patterns.

5.7 Experiments

In order to test the performance of our proposed method, we scanned several challenging scenes using a Gige Prosilica 1360 camera and a Samsung P400 projector. The pixel resolution of the camera and the projector were 1360×1024 and 800×600 respectively. We compared correspondence results from Gray codes, Phase-shift, the Gupta *et al.* method and our method based on unstructured light. We tested four scenes that exhibit different challenges: `Ball`, `Games`, `Grapes & Peppers` and `Corner`. Results on other scenes are available online at [2]. For our method, we used 200 patterns of frequency $f = 64$. For the `Ball` and `Corner` scenes, matches from our method were pruned (i.e. camera pixels set to black in the figure) when the intensity standard deviation observed in a camera pixel was below 3. For the `Games` and `Grapes & Peppers` scenes, this strategy eliminated too many matches because of low contrast in some areas. We instead pruned all matches with cost over 50 (see Fig. 5.10).

Ball

The `Ball` scene is similar to the scene used in Sec. 5.5.3. It is composed of two walls, a floor and a ball that creates a highlight and a depth discontinuity at its boundary. Results of all tested methods are shown in Fig. 5.14. Our method (top row) gives good results for high and low significant bits of the correspondence map. The Gupta *et al.* method (2^{nd} row) performs well, but a few pixels are discarded near the intersection of the wall and the ground, where interreflections are higher. Gray codes (3^{rd} row) fail to recover highly significant bits on the floor near the walls because of indirect

lighting. Phase-shift (last row) results are presented for 16 and 64 cycles per frame patterns. Only low significant bits are shown (i.e. no phase unwrapping is applied). It has difficulties near the walls and features a wavy matching typical of direct coding methods in the presence of indirect lighting. This artifact gets worse when using a lower frequency.

Fig. 5.15 shows results of all methods at depth discontinuities. Our method recovers correctly the edges of the ball. The Gupta *et al.* method rejects a lot of matches at discontinuities. As discussed at the end of Sec. 5.4.2, this is probably due to the instabilities of position-based codes at discontinuities because of code blending. The iterative process used by Gupta *et al.* [28, 77] would not help to recover matches at discontinuities because each iteration only removes projector pixels that were correctly matched. Unfortunately, none of the two blended codes at a discontinuity is ever matched correctly. The instability at discontinuities is also clearly visible for Gray codes (Fig. 5.15(c)) at the right edge of the ball, where a one pixel border is wrong. Gray codes also suffer from interreflections, as can be seen on the upper left edge of the ball. Phase-shift is affected by code blending at depth discontinuities. This can be seen as a blurred edge.

In order to verify that all methods perform similarly when unaffected by indirect illumination, we selected a region where indirect illumination is negligeable, namely the upper left region of the left wall, and compared the matches of all methods. At least 80% of the matches were exactly the same. All the remaining matches were within a distance of one pixel.

Games

Fig. 5.16 shows results for the `Games` scene, which exhibit a lot of sharp discontinuities. Also observe the curved surface of the cylindrical box, especially the soft edges at the sides where surface normals become perpendicular to the optical axis of the camera. There, Gray codes fail to recover correct matches. The Phase-shift method performs

better, but the floor correspondences exhibit wavy results due to light bouncing off the cylindrical and rectangular boxes. The Gupta *et al.* method performs well, but error pixels are still flagged at the top of the rectangular box, the left and right edges of the cylindrical box and because of light reflection at its bottom. Our method successfully matches all these problematic areas. Notice that matches due to reflections at the left of the scene were not pruned because their matching cost was low even though contrast was low as well.

Grapes & Peppers

Results for the `Grapes & Peppers` scene are shown in Fig. 5.17. Grapes are translucent fruits that create subsurface scattering, and peppers have very shiny surfaces. Subsurface scattering is especially challenging to high-frequency patterns because they become blurry. Our method works quite well for this difficult scene. The Gupta *et al.* method fails to match pixels at the bottom of the right pepper and a few pixels on the grapes. Phase-shift and Gray codes also work pretty well, although Gray codes fail at the edges of the peppers, due to interreflections.

Corner

The `Corner` scene was made using two highly reflective surfaces set at a 90 degree angle. Gray codes and Phase-shift badly fail to match pixels near the corner. The Gupta *et al.* method is more successful in the sense that it does not exhibit wrong matches, but it misses alot of good matches. Our method works much better in that it is able to recover all matches, even at the corner. Notice that the black tape holding the reflective material could not be matched successfully because of its very low reflectance. The quality of the results of our method can be seen in Fig. 5.19 which shows all scenes reconstructed by triangulation.

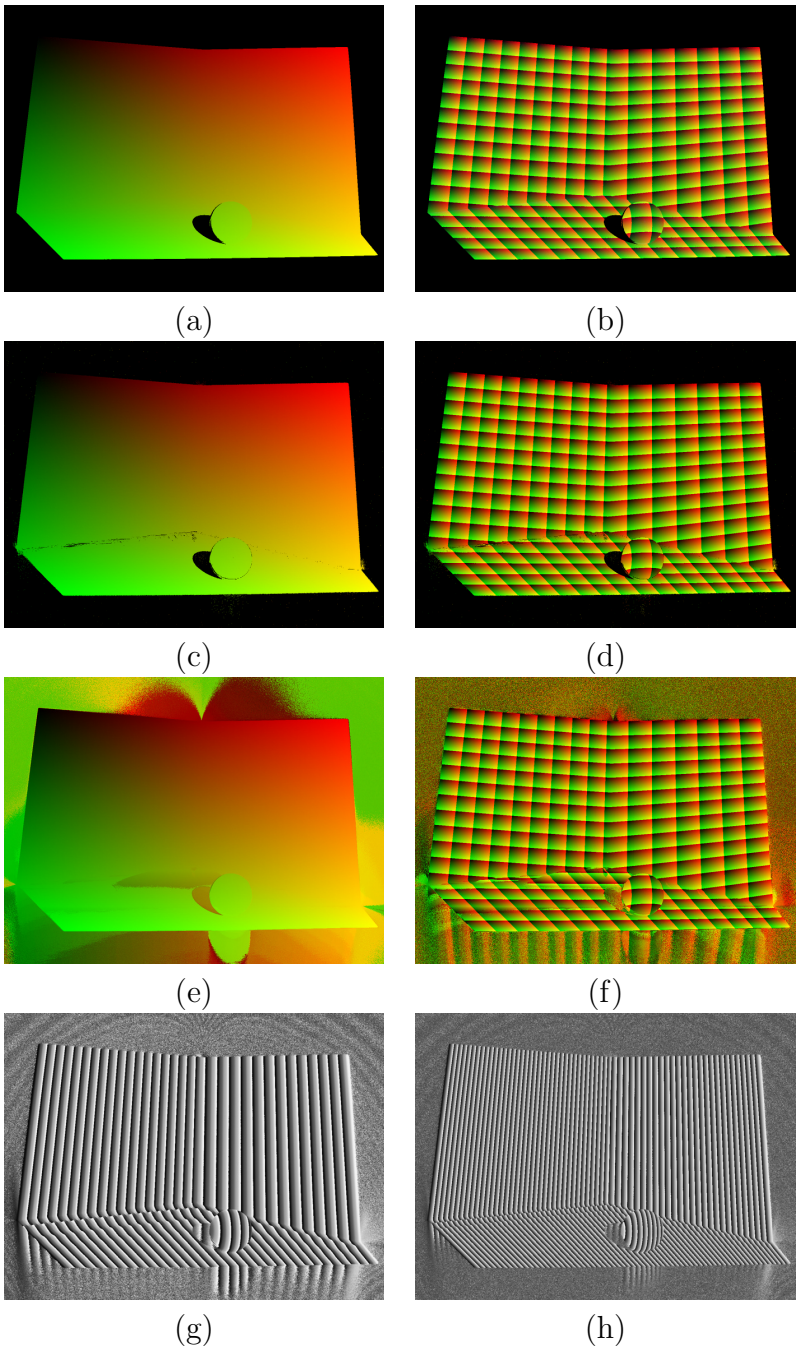


Figure 5.14. Results for the `Ball` scene using the tested methods, namely our unstructured light method (top row), the Gupta *et al.* method (2^{nd} row), Gray codes (3^{rd} row) and Phase-shift (bottom row). The right column shows the low significant bits of the correspondence map only. For Phase-shift, results are presented for a different frequency on each column.

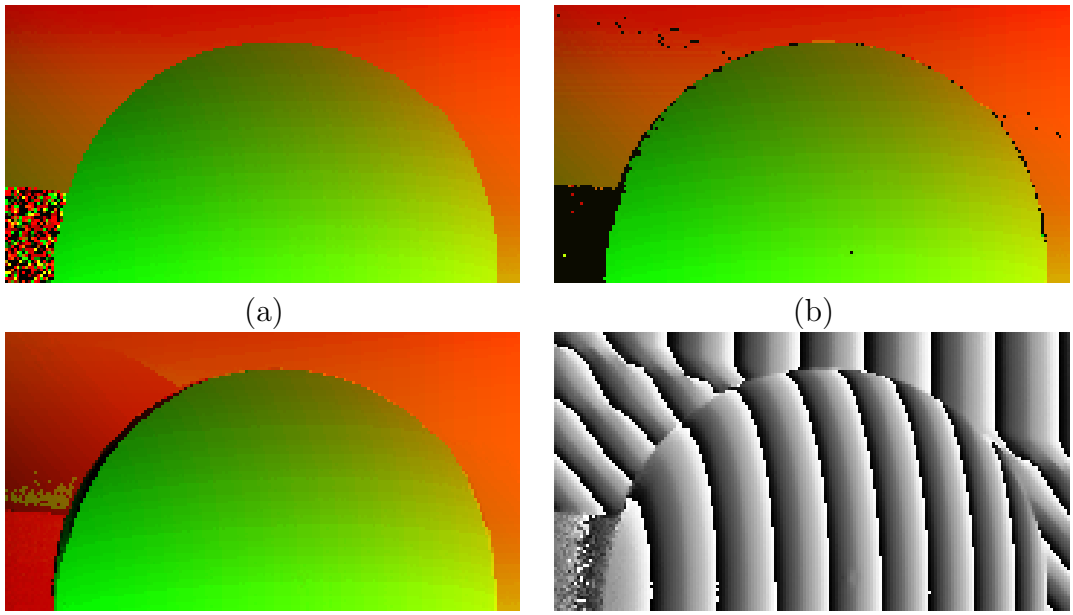


Figure 5.15. Results for a ball using all tested methods: (a) our unstructured light method (b) Gupta *et al.* (c) Gray codes (d) Phase-shift. In (b), black pixels represent matches identified as inconsistent.

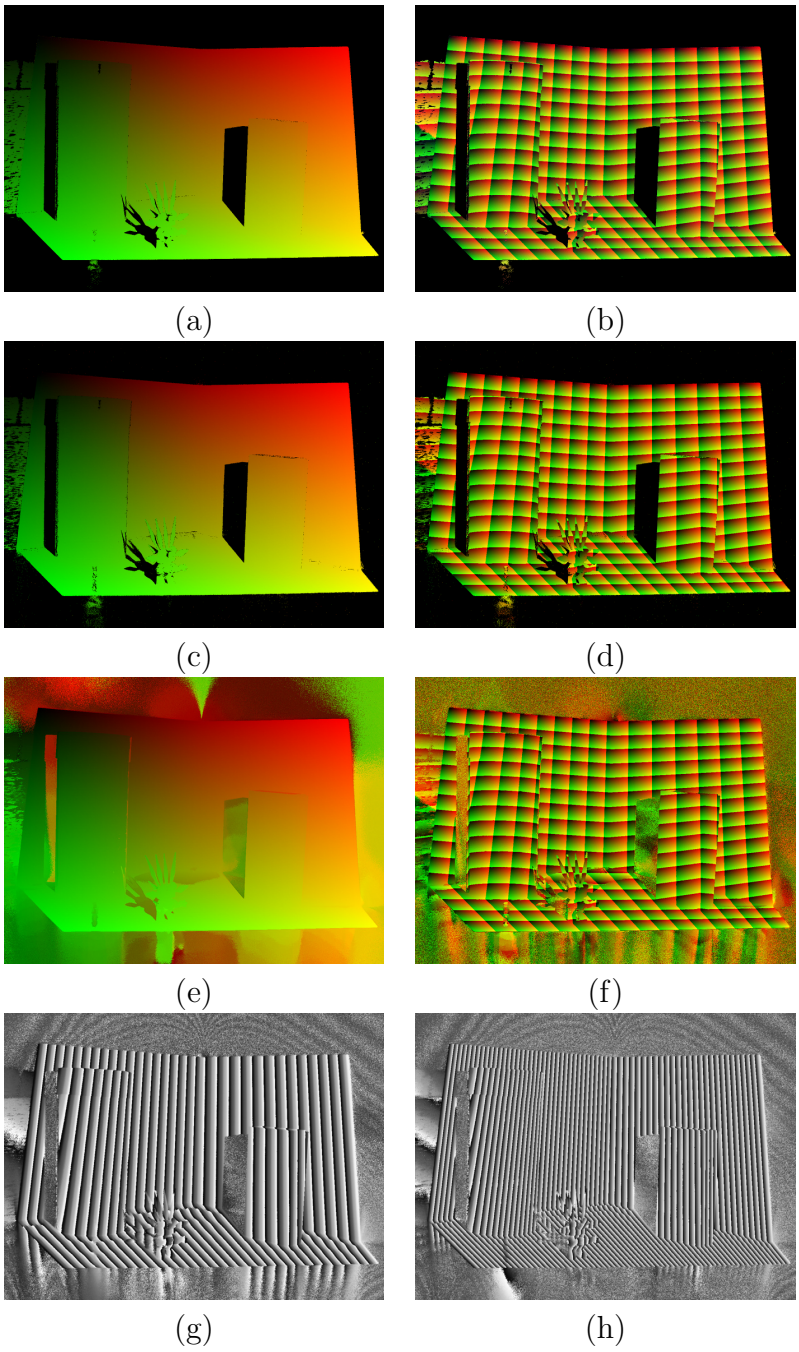


Figure 5.16. Results for the `Games` scene using the tested methods, namely our unstructured light method (top row), the Gupta *et al.* method (2^{nd} row), Gray codes (3^{rd} row) and Phase-shift (bottom row). The right column shows the low significant bits of the correspondence map only. For Phase-shift, results are presented for a different frequency on each column.

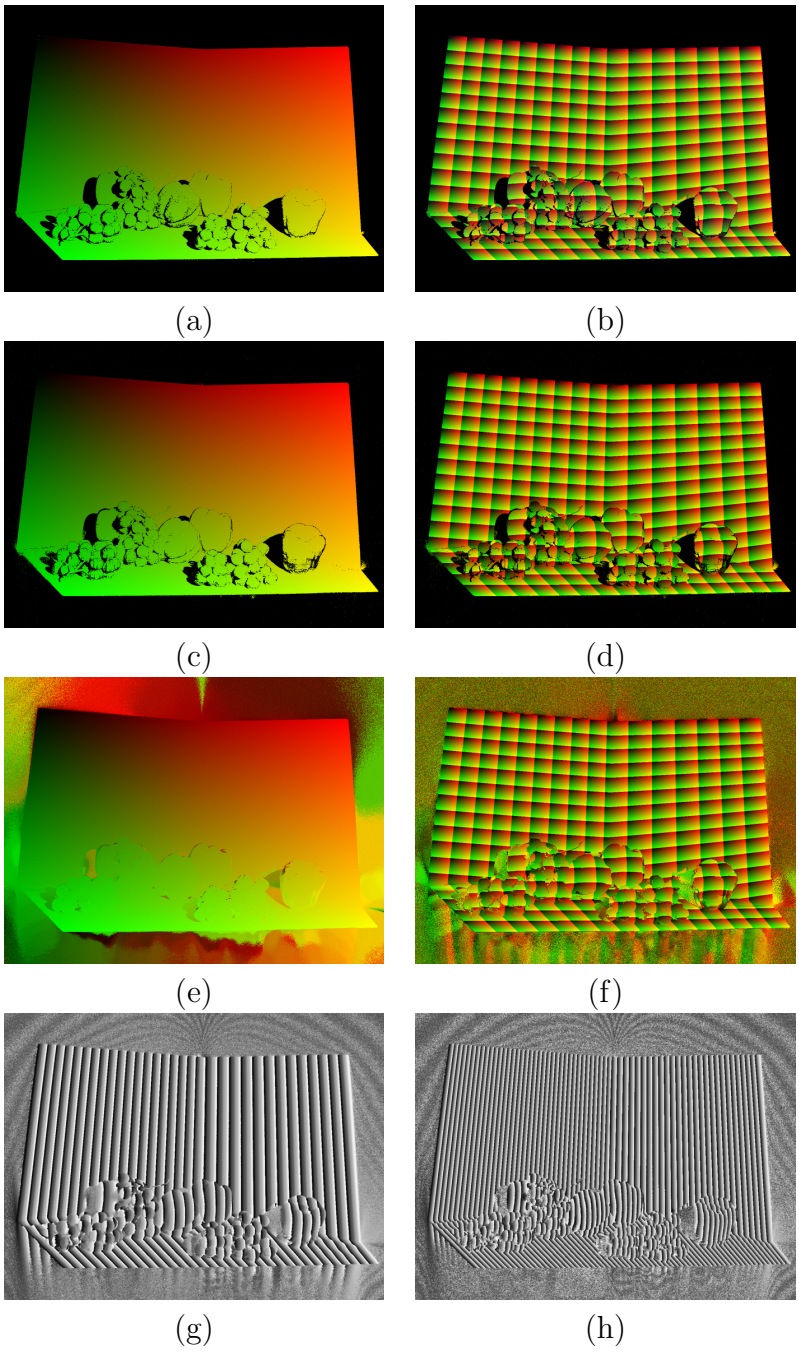


Figure 5.17. Results for the `Grapes & Peppers` scene using the tested methods, namely our unstructured light method (top row), the Gupta *et al.* method (2^{nd} row), Gray codes (3^{rd} row) and Phase-shift (bottom row). The right column shows the low significant bits of the correspondence map only. For Phase-shift, results are presented for a different frequency on each column.

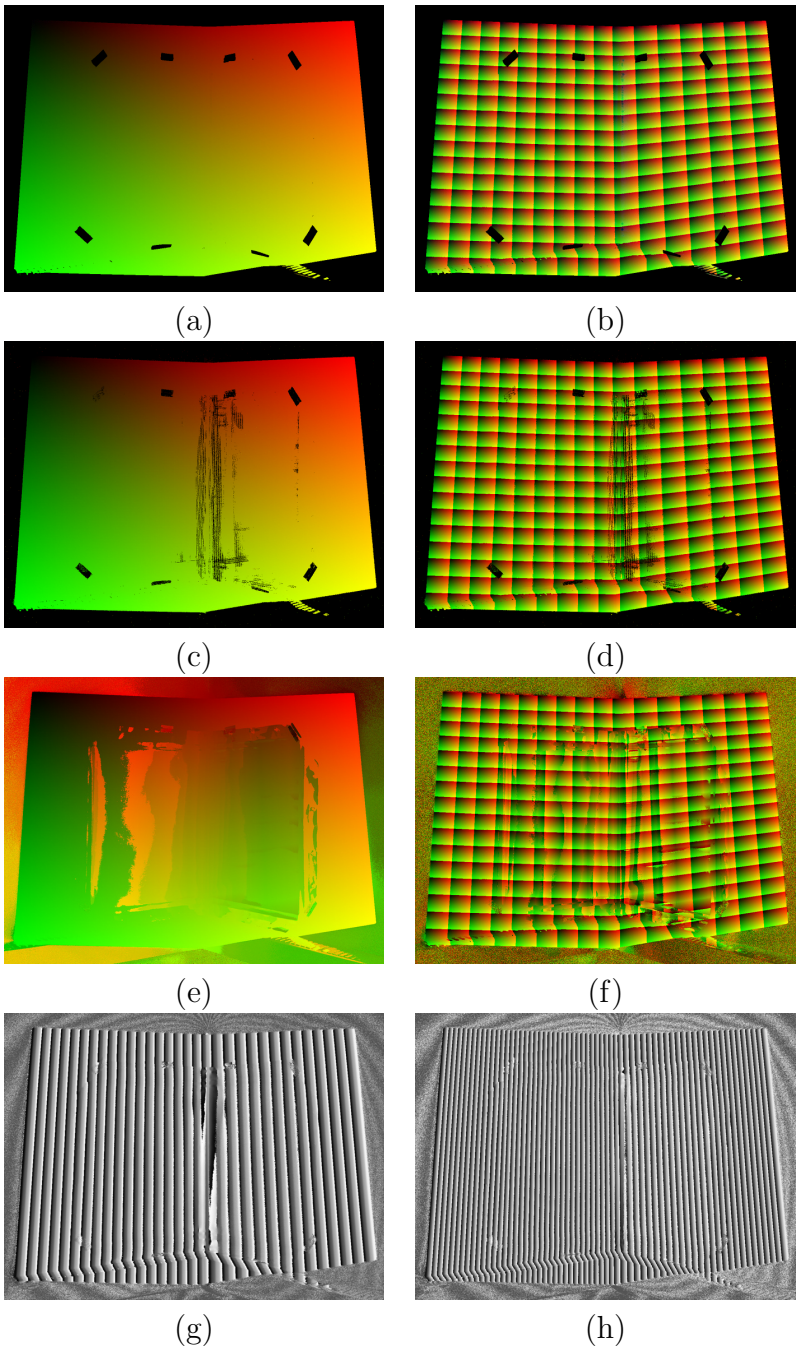
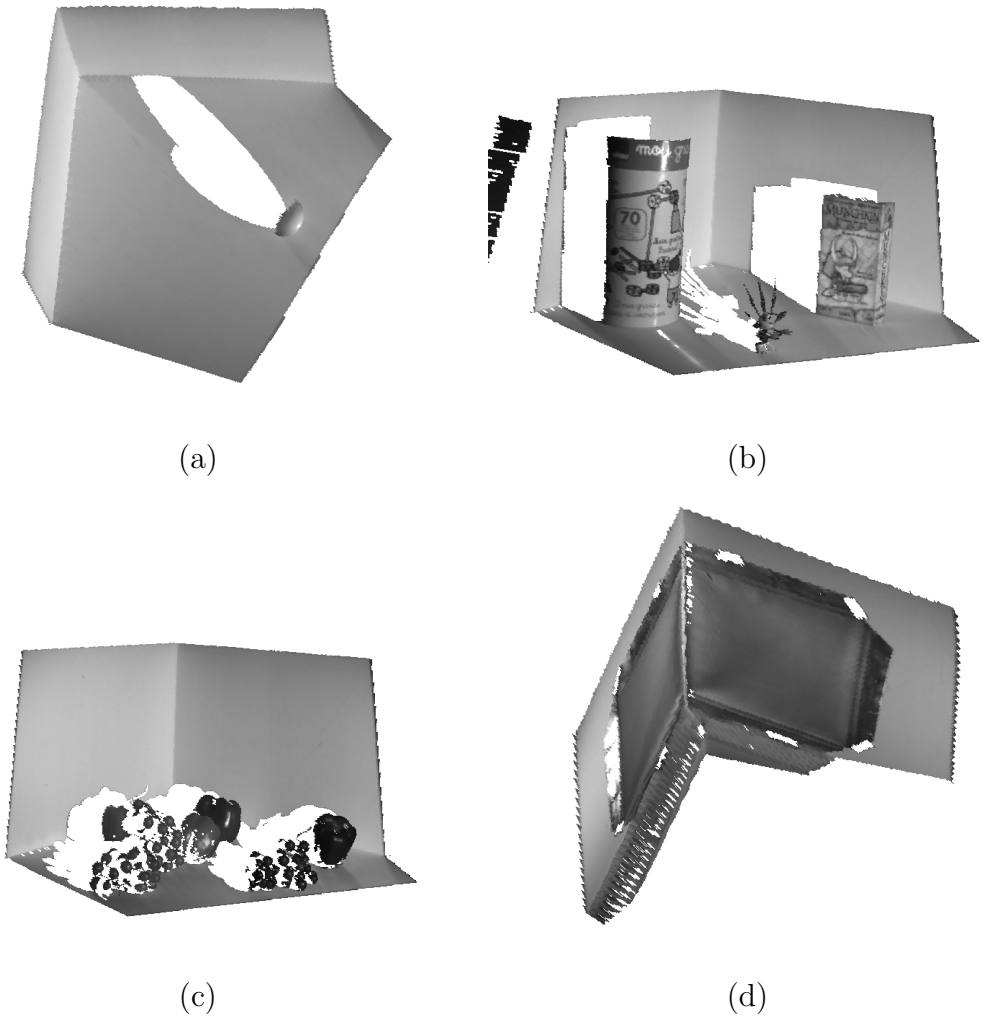


Figure 5.18. Results for the `Corner` scene using the tested methods, namely our unstructured light method (top row), the Gupta *et al.* method (2^{nd} row), Gray codes (3^{rd} row) and Phase-shift (bottom row). The right column shows the low significant bits of the correspondence map only. For Phase-shift, results are presented for a different frequency on each column.



(a) (b)

(c) (d)

Figure 5.19. Triangulation from the correspondences given by our method for the `Ball` scene (a), the `Games` scene (b), the `Grapes & Peppers` scene (c) and the `Corner` scene (d).

5.8 Conclusion

In this paper, we addressed the problem of indirect illumination in structured light systems by taking advantage of a new approach to active reconstruction that uses patterns unrelated to projector pixel position. The only constraint imposed on these unstructured light patterns is that a sequence of these patterns identifies every projector pixels by a unique code. The proposed band-pass white noise patterns are designed to reduce the effects of indirect illumination and be robust to other issues such as low camera-projector pixel ratios. Because of the high number of patterns, the method is robust to capture errors and the matching algorithm provides very good performance with respect to depth discontinuities. Future works could address the problem of estimating matches at sub-pixel precision, as well as reducing the number of patterns by increasing orthogonality between them, while still keeping their basic properties.

PARTIE III

PERCEPTION DE L'OMNISTÉRÉO

Chapitre 6

VISION BINOCULAIRE ET PLACE IDÉALE DU SPECTATEUR

Nous avons mentionné en introduction qu'une image omnistéréo ne permet pas une perception stéréo sans distorsion. Dans le présent chapitre, nous introduisons, à la section 6.1, des notions liées à la vision binoculaire humaine. À la section 6.2, nous justifions que la place idéale du spectateur est au centre de l'écran cylindrique. Ces notions seront utiles à la compréhension du chapitre 7 qui analyse si les distorsions sont perçues par un spectateur au centre de l'écran cylindrique.

6.1 Mécanisme de la vision binoculaire humaine

Nous décrivons ici brièvement le mécanisme normal de la vision binoculaire humaine. Lorsqu'un spectateur fixe un objet dans la scène, il y a convergence[1] des yeux accompagnée du phénomène d'accommodation[2]. La convergence des yeux est illustrée à la figure 6.1. L'angle défini par les deux lignes pointillées est appelé angle de convergence. Tout point à la même profondeur que cet objet est vu à la même excentricité[3]. Par contre, tout point plus rapproché ou plus éloigné que cet objet est vu à une excentricité différente par chaque oeil. C'est cette différence d'excentricité, appelée disparité, qui permet à la vision binoculaire d'estimer les profondeurs d'une scène lors de la fusion des deux images par le cerveau.

[1] Les yeux convergent de façon à ce que l'objet soit vu par la fovéa de chaque oeil. La fovéa, au centre de la rétine, est la zone où la vision des détails est la plus précise. Cette précision s'explique par la grande densité des photorécepteurs.

[2] Le phénomène d'accommodation modifie la courbure du cristallin pour qu'une image nette de l'objet se forme sur la rétine.

[3] L'excentricité rétinienne est la distance angulaire par rapport à la fovéa.

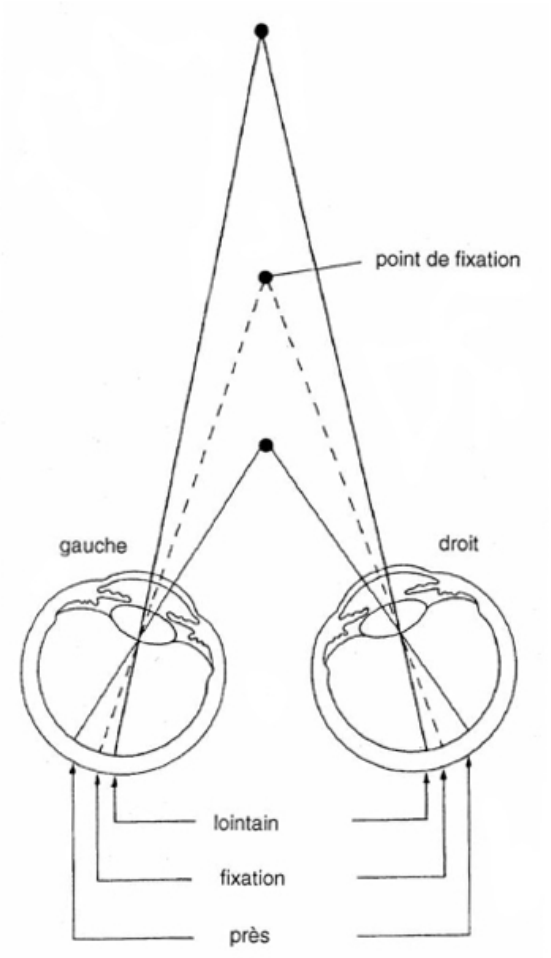


Figure 6.1. Convergence de la vision binoculaire. D'après Jean-Louis Vercher de L'Institut des Sciences du Mouvement, Université de la Méditerranée (France). Tirée de `http://acces.inrp.fr/acces/ressources/neurosciences/vision/VisionMarseille/vergence`.

Lors d'une projection stéréo ou omnistéréo, la convergence des yeux se fait de façon similaire, mais l'accommodation se fait toujours sur l'écran même si le spectateur regarde un objet (virtuel) devant ou derrière l'écran. Ceci risque de causer des maux de têtes, qui peuvent être atténués en ajustant les disparités pour que l'objet fixé soit à la profondeur de l'écran.

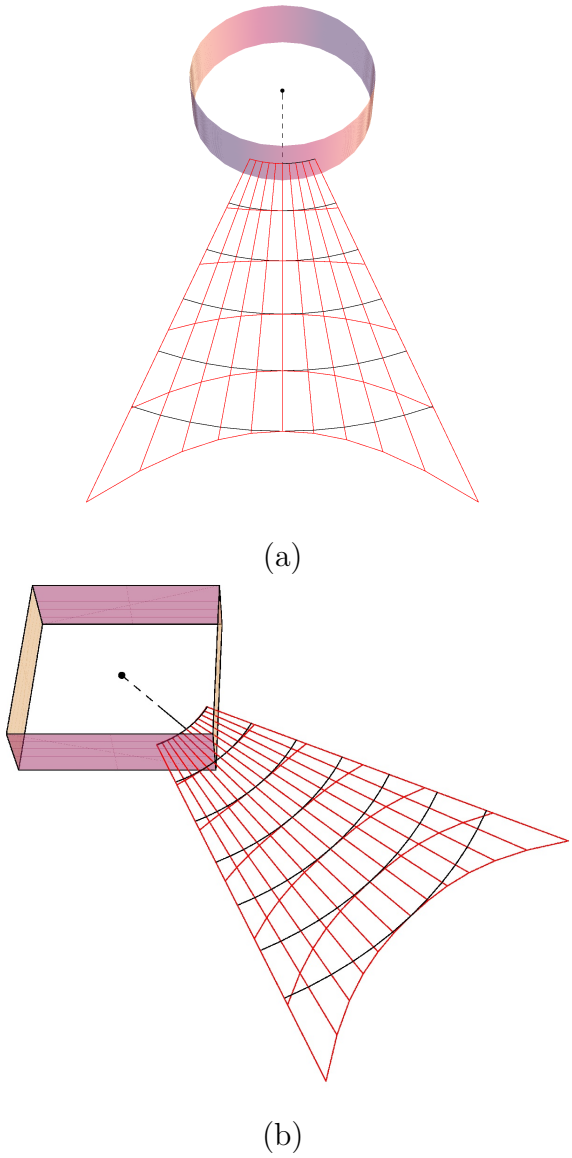
6.2 Place idéale du spectateur

Lors d'un tournage stéréo standard, deux caméras capturent l'image que verrait chaque oeil d'un spectateur face à l'écran et au centre de la salle de cinéma. Tout spectateur qui n'occupe pas cette place idéale perçoit des distorsions de profondeur [8, 70]. Dans un environnement omnistéréo, la place idéale est aussi au centre de l'écran cylindrique puisqu'elle correspond à celle du trépied lors du tournage. Mais dans un environnement omnistéréo, l'orientation du regard du spectateur est inconnue. Le tournage omnistéréo doit donc capturer des images étroites prises dans toutes les orientations, mais seule l'image au centre du champ visuel d'un spectateur centré a été capturée avec la même orientation que son regard. Toutes les images contiguës, capturées avec une orientation différente, déplacent les points sur l'écran par rapport à ce que le spectateur devrait voir. Par conséquent, la fusion des images gauche et droite crée des distorsions.

Les figures 6.2 et 6.3 montrent respectivement des exemples de distorsions omnistéréo (indiquées en rouge) pour un spectateur centré et non centré, et dont les yeux sont à la hauteur du centre vertical de l'écran. La scène (indiquée en noire) est composée d'arcs de cercle de différents rayons couvrant 60° d'angle de vue. Des lignes droites sont également ajoutées à un intervalle de 5° d'excentricité. De plus, ces figures comparent les distorsions des environnements cylindrique et CAVE. Nous verrons au chapitre qui suit comment calculer les distorsions de façon précise. Nous faisons ici trois observations à partir de ces figures. Premièrement, le spectateur

au centre perçoit principalement des distorsions de profondeur. Deuxièmement, les distorsions sont presque nulles au centre du champ visuel d'un spectateur centré, mais deviennent significatives pour un spectateur non centré. Troisièmement, il y a distorsion en forme de coin pour un spectateur non centré dans un environnement CAVE. Ceci peut justifier l'utilisation d'un écran cylindrique.

Pour un spectateur au centre, les distorsions de profondeur se situent à la périphérie du champ visuel, où l'acuité est moins précise en raison de la diminution du nombre de photorécepteurs sur la rétine hors de la fovéa [33]. Ainsi, nous avons cherché à savoir au chapitre qui suit si ces distorsions sont perçues par un spectateur au centre. Notre analyse suppose un rayon de l'écran de 230cm, comme l'écran cylindrique que nous avons construit. Nous verrons, entre autres, que les distorsions sont nulles pour des points à une distance égale au rayon de l'écran, mais qu'elles augmentent graduellement pour des points à l'avant et à l'arrière. Nous montrerons que ces distorsions sont négligeables, sauf pour des points très rapprochés à moins d'un mètre du centre par exemple. L'utilisation d'un écran au rayon plus grand diminuerait la distorsion des points plus éloignés du spectateur, mais augmenterait celle des points plus rapprochés. Il faudrait alors éviter l'utilisation de points de la scène dans un rayon plus grand qu'un mètre.



(a)

(b)

Figure 6.2. Exemples de distorsions omnistéréo pour un spectateur au centre (a) d'un écran cylindrique de 2,3m de rayon (b) d'un environnement CAVE avec des murs de 4,6m de large. La scène est composée d'arcs de cercle (noire), mais des distorsions changent leur courbure (rouge). À noter qu'il n'y a aucune distorsion au centre du champ visuel. Les lignes pointillées indiquent l'orientation du regard du spectateur.

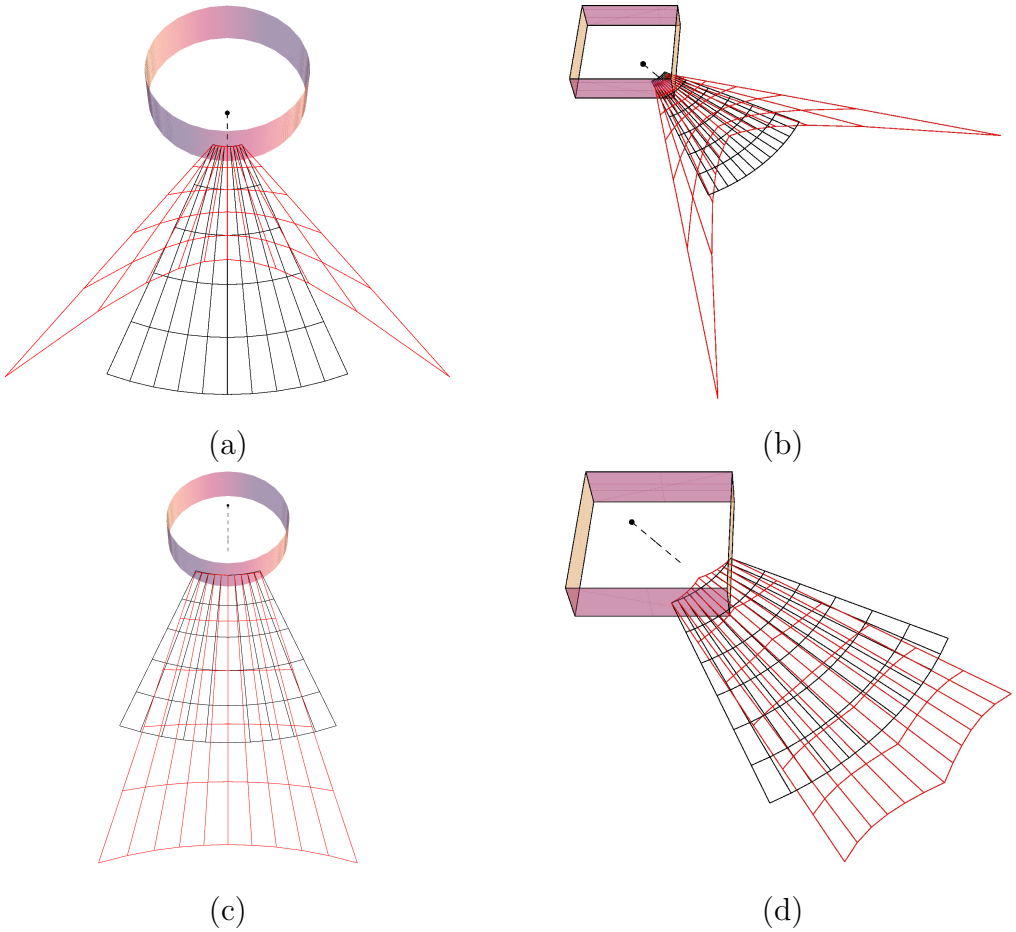


Figure 6.3. Exemples de distorsions omnistéréo pour un spectateur (a-b) près de l'écran, (c-d) plus loin de l'écran. La scène est composée d'arcs de cercle (noire), mais des distorsions changent leur courbure (rouge). À noter la distorsion en forme de coin pour les environnements CAVE. Les lignes pointillées indiquent l'orientation du regard du spectateur.

Chapitre 7

ANALYSIS OF DISPARITY DISTORTIONS IN OMNISTEREOSCOPIC DISPLAYS (ARTICLE)

Ce chapitre présente l'article suivant :

> V. Couture, M.S. Langer, S. Roy. *Analysis of Disparity Distortions in Omnistereoscopic Displays*. ACM Transactions on Applied Perception, vol. 7, n° 4 (2010), présenté au Symposium on Applied Perception in Graphics and Visualization (APGV 2010).

La littérature traitant de l'omnistéréo s'intéresse principalement au tournage et à la projection. Mais, elle n'établit aucun lien entre les distorsions stéréo causées par l'omnistéréo et les limites perceptuelles de la vision des spectateurs. Ce chapitre s'intéresse à la perception des distorsions omnistéréo, qui augmentent progressivement du centre à la périphérie du champ visuel. Plus particulièrement, nous calculons les distorsions causées par deux modèles de projection pour le rendu omnistéréo de scènes virtuelles. Le premier modèle utilise les profondeurs de la scène pour ne donner aucune erreur de disparité au centre du champ visuel. Le deuxième modèle utilise le principe des caméras-fente pour un tournage omnistéréo, tel que décrit au chapitre 2.

Nous établissons le lien entre ces distorsions et les limites d'acuité de la vision binoculaire humaine. Ces limites sont connues pour un champ de vision de 40° seulement parce que l'acuité binoculaire est généralement considérée très faible plus en périphérie. Nous étudions les erreurs de disparités horizontales afin de vérifier si les distorsions de profondeur qu'elles induisent dépassent les limites d'acuité binoculaire. Une telle comparaison vise à vérifier si ces distorsions peuvent être percues ou sont négligeables.

Nous présentons ici l'article dans sa version originale.

Abstract

An omnistereoscopic image is a pair of panoramic images that enables stereoscopic depth perception all around an observer. An omnistereo projection on a cylindrical display does not require tracking of the observer's viewing direction. However, such a display introduces stereo distortions. In this paper, we investigate two projection models for rendering 3D scenes in omnistereo. The first is designed to give zero disparity errors at the center of the visual field. The second is the well-known slit-camera model. For both models, disparity errors are shown to increase gradually in the periphery, as visual stereo acuity decreases. We use available data on human stereoscopic acuity limits to argue that depth distortions caused by these models are so small that they cannot be perceived.

7.1 Introduction

Binocular depth perception requires an observer to establish point correspondences between two images, and to use the disparity differences as a cue to relative depth of visible surfaces. In designing binocular displays such as 3D cinema, it is traditionally assumed that the baseline joining the two eyes is known relative to the screen and, in particular, that the baseline is parallel to the screen. Other methods have relaxed these assumptions though. For example, in some virtual environments such as CAVEs [17, 18], a head tracking system has been used which allows the viewer position and the viewer's orientation to be updated. These environments aim to display exact stereo images to a single observer.

Another approach is to use omnistereoscopic images, which are multi-viewpoint panoramic images that contain stereo information all around an observer [34, 37, 45, 48, 49]. Similarly to CAVEs, omnistereo images can be used for navigation in a virtual environment. However, they remove the need to track the head orientation [10, 11, 45]. An example of an omnistereo image is shown in Fig. 7.1.

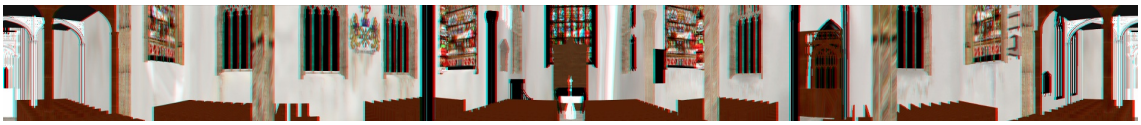


Figure 7.1. Omnistereoscopic image rendered from a 3D scene model of the Charles Church in Plymouth (UK), courtesy of Karol Kwiatek. The image is encoded in red/cyan anaglyph format.

Fig. 7.2(a) illustrates an omnistereoscopic display that consists of a cylindrical screen and an observer located at the center O. The baseline of the observer's eyes is perpendicular to the fixation point, which can be anywhere along the line through O that is perpendicular to baseline – called the "median line". In this setup, the observer is free to rotate his head, i.e. the baseline orientation, but the position of the observer is assumed to be remain at or near the center O. We note that the ratio of baseline to display radius is typically much smaller than that illustrated in the figure, so the model is less sensitive to the exact observer position.

One of the challenges of creating omnistereo images is that it is impossible to render correct stereo disparities for all observation orientations at the same time, since the correct rendered stereo disparity depends on the orientation of the observer. In this paper, we analyze the distortions that are present in omnistereo displays. We investigate two projection models for rendering omnistereo images from 3D scenes. For the first model, the disparity errors are *designed* to be zero on the median plane between the eyes regardless of which direction the observer is oriented, and to gradually increase towards the periphery of the visual field [65]. This design is motivated by the spatial acuity properties of the human visual system, in particular stereo acuity is highest in the fovea and decreases precipitously with eccentricity. We also investigate the well-known *slit-camera* model and show that it produces similar disparity errors. Moreover, we show that for both models the disparity errors are so small that they are perceptually negligible within a 20° eccentricity.

To our knowledge, this is the first attempt to connect depth distortions in omnistereo environments to known stereo vision limits of human observers. Finally, we briefly describe system implementation of the model.

A layout of this paper is as follows. In Sec. 7.2, we briefly review prior works on omnistereo imaging. In Sec. 7.3 we present a projection model that gives zero disparity error for all points on the median plane between the eyes of an observer centered in a cylindrical omnistereo display. The omnistereo distortions caused by this model are discussed in Sec. 7.4. Sec. 7.5 shows that the standard slit-camera model also causes similar distortions, at least when points are not too far from the vergence point of the cameras. Then, existing limits of human stereo acuity are discussed in Sec. 7.6 in which we argue that disparity errors for both models are too small to be perceived. Our discussion is restricted to horizontal disparities only, *i.e.* we do not address vertical disparities. Details on the implementation are presented in Sec. 7.7. We conclude in Sec. 7.8.

7.2 Previous Work

Most work on omnistereo images addresses how they can be *captured* with a stereo camera [34, 37, 45, 48, 49]. In [48, 49], a stereo pair of cameras is rotated to fully cover 360° degrees (see Fig. 7.2(b,c)). At every one or two degrees, slit-images are captured having a small horizontal field of view, whose angular width depends on the amount of rotation between consecutive frames. In practice, a set of columns (say 50) is considered for HD images. The resulting omnistereo images are usually displayed as a panorama on a small planar surface such as a monitor [49].

In contrast, this paper considers immersive environments in which omnistereo images are rendered and displayed on a cylindrical screen surrounding the viewer. To our knowledge, the only prior published method that uses a cylindrical screen for projecting rendered omnistereo images is [10], which uses the above slit-camera

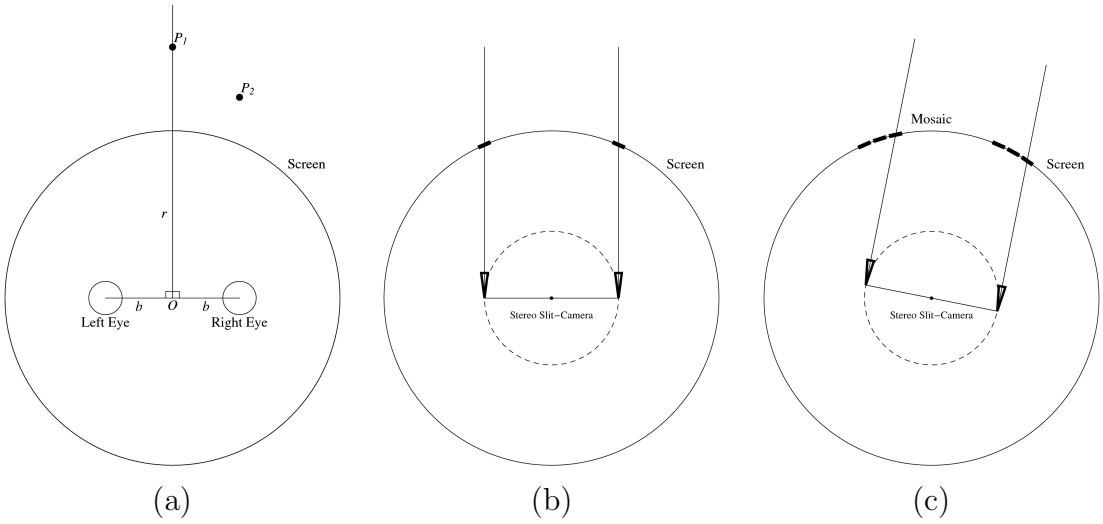
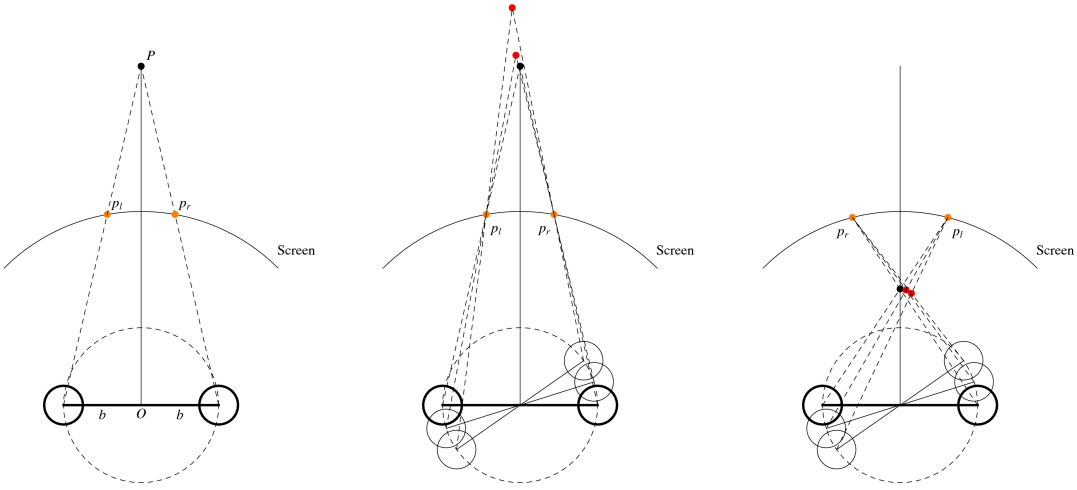


Figure 7.2. (a) Omnistereo immersive environment viewed from above: a cylindrical screen of radius r and an observer at the center of the screen. Two scene points are shown. The projection model introduced in Sec. 7.3 gives zero stereo disparity error for a point P_1 on the median line. Errors increase in the periphery of the visual field, for example, for a point P_2. (b) Previous work on omnistereo images uses a rotating stereo pair of slit-cameras verged at a specific distance. Here the distance is infinity, i.e. cameras are parallel. (c) Image slits are stitched together in a mosaicing process to cover 360°.

projection model [37, 49]. It is observed in [10] that the frame of the stereo glasses limits the view window for stereo input to the eyes, but otherwise there is no mention of the disparity information available to the observer and possible perceptual limits. The present paper is directly concerned with such perceptual limits. In particular, we investigate the resulting disparity errors to see if the depth distortions they induce are well above known detection thresholds and hence whether they can be perceived.

7.3 Median Plane Projection Model

We first describe a projection model that gives zero disparity error for *all* points on the median line between the two eyes. This model is slightly different from the



(a) (b) (c)

Figure 7.3. (a) Rendering positions p_l and p_r of a point P are computed with respect to the baseline orientation for which P lies on the median line between the eyes. Note that only part of the full circular screen is shown. (b) If the observer rotates his head, then P becomes a point in the periphery of the visual field. The rendered points p_l and p_r on the screen are triangulated again and a distortion is introduced. The triangulated P appears at a different location (see red dots). Points behind the screen appear further away in depth. (c) Points in front of the screen appear closer in depth.

standard slit-camera model in that the latter causes disparity errors on the median plane which depend on the vergence of the eyes (or cameras, in the case of image capture). See Sec. 7.5 for details on the slit-camera projection model.

For simplicity, we first present the model for the 2D case (see Fig. 7.3). Recall from Fig. 7.2(a) that P is a point in the scene, and O is the center of the screen circle of radius r. The head is centered at O and each eye is located on a circle of radius b centered at O, such that $2b$ is the *baseline* distance between the eyes. For the cylindrical display in our lab, $r = 230$ cm. For the plots and computations later

in the paper, we take $b = 3.25$ cm. Note that for illustration purposes, Fig. 7.3 uses a larger $b : r$ ratio in than in the actual lab setup.

The projection model requires known scene depths, namely we have a virtual 3D scene model that is being rendered. Given b and r, we compute for each point P the rendered screen positions p_l and p_r, that is, the positions *on the cylindrical display screen* where the rendered point P is projected for the left and right eye's image, respectively. Because it is well known that stereo acuity is highest at the center of the visual field [33], we design a projection model that gives zero disparity error for a point P when an observer is oriented so that the median line passes through P.

For any point P, we therefore render this point by assuming that the point lies in the head's median plane. Because the display is rotationally symmetric, we consider without loss of generality the eyes located at $(\pm b, 0)$ and a point $P = (0, Z)$. The screen pixel positions p_l and p_r are each computed by intersecting a line with a circle, namely a line joining the corresponding eye and P with the circle of radius r centered at the origin. For the right eye, this intersection is given by:

$$(p_{r,x}, p_{r,z}) = (\frac{b\ (Z^2 - \sqrt{\Delta})}{Z^2 + b^2},\ \frac{Z\ (b^2 + \sqrt{\Delta})}{Z^2 + b^2}) \tag{7.1}$$

where

$$\Delta = r^2(Z^2 + b^2) - Z^2 b^2. \tag{7.2}$$

The screen position for the left eye is computed similarly, using $-b$ instead, giving:

$$(p_{l,x}, p_{l,z}) = (-p_{r,x}, p_{r,z}) \tag{7.3}$$

We extend the above projection model to the 3D case by considering eyes at $(\pm b, 0, 0)$ and a point $P = (0, Y, Z)$, with the display now a vertical cylinder of radius r centered at the origin. Screen positions $p_{r,x}, p_{r,z}$ remain the same, and the vertical

screen position is given by:

$$p_{l,y} = p_{r,y} = \frac{Y p_{r,z}}{Z}. \tag{7.4}$$

A point P that lies on the median plane is projected to the correct screen positions p_l and p_r, and so in principle its 3D position can be correctly estimated by triangulation. For 3D points that are not on the median plane, triangulation errors occur that lead to small geometric distortions. The severity of these errors increases gradually with eccentricity. In the following section, we will analyze these errors.

Before doing so, we elaborate on a few assumptions of the projection model. First, when projecting a point P, were are assuming that the observer's eyes are located as in Fig. 7.3 and that the observer is fixating somewhere on the head's median plane. Our analysis does not consider disparity errors relative to stereo acuity when the observer is fixating left or right of the head's median plane. Second, at each new fixation, there is a slight shifting of center of projection (the pupil) as the eyes rotate, since the pupil is slightly displaced from the center of rotation. Since this displacement is so small relative to the baseline, we ignore it in our model. A third assumption is that the model is using a pinhole projection, and so we are ignoring blur and accommodation. As in typical stereo displays, our images are focused on the screen and this leads to a vergence-accommodation conflict [32]. However, this accommodation conflict is most significant for screens closer than 2 m and so it our setting the conflict would only arise for objects rendered to be closer than the screen.

7.4 Geometric distortions and disparity errors for median plane model

In the previous section, we discussed a model for projecting a 3D point onto a cylindrical screen such that a point is triangulated to its correct 3D position when when the point lies on the median plane of the observer. For points that are not on the medial plane, triangulation errors occur which lead to small geometric distortions.

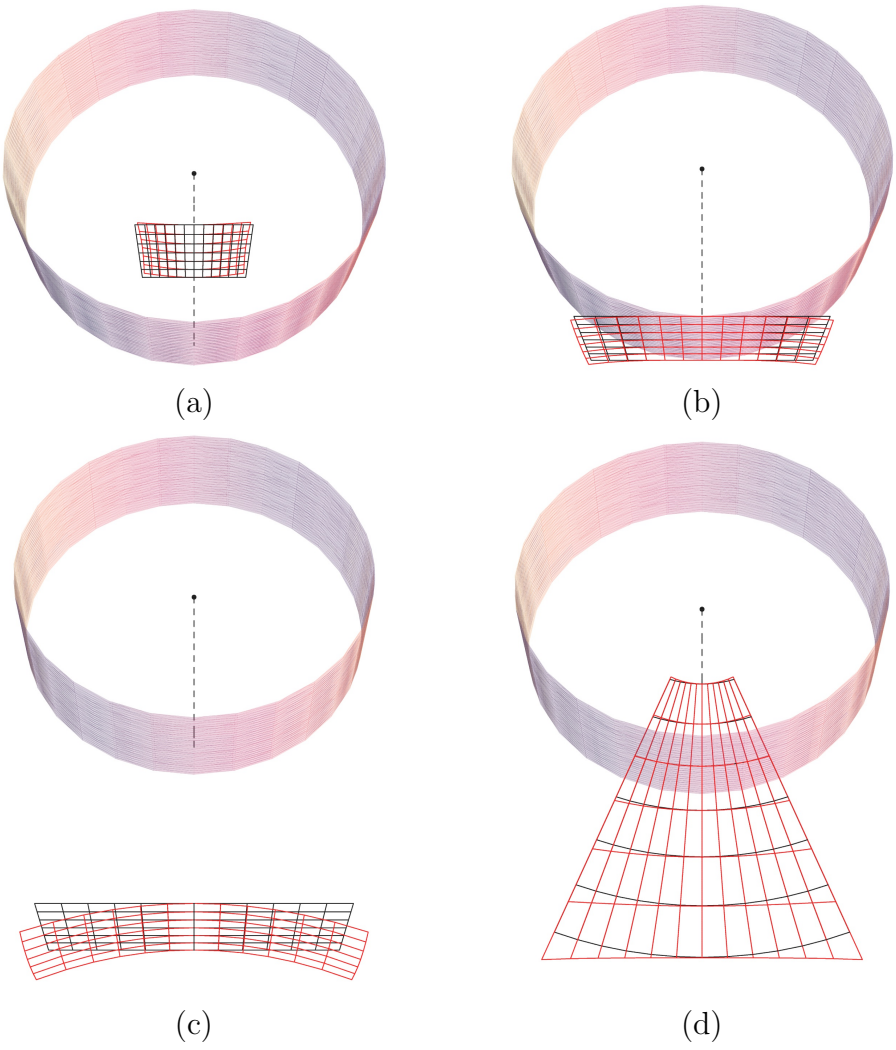


Figure 7.4. For an observer looking at a specific orientation on the cylindrical screen indicated by the dashed line, the omnistereo projection model causes distortions in the periphery of the visual field. In (a-c), a plane is shown at different depths, both its true shape (black) and its triangulated distorted shape (red). In (a), the left and right sides of the plane in front of the screen are distorted to be even closer. In (b-c), the sides of the plane are distorted to be farther way. Points on the screen itself are not distorted. Note that these planes are also slightly distorted vertically, but these distortions are very small and will not be discussed further. In (d), circular curves with varying radius (black) are shown in the x-z plane with their perceived distorted shapes (red). Black grid lines that are radial (constant direction) are drawn, but they are not visible because they are overwritten by the red grid lines because distortion of visual orientation is very small.

Fig. 7.4 illustrates the distortions that are caused by the model when the screen radius is $r = 230$ cm and the eye baseline is $2b = 6.5$ cm. In Fig. 7.4(a-c), three planar surfaces are shown in black at different depths, with the distortions shown in red. There is zero distortion at eccentricity 0°, by design. In addition, points on the screen are not distorted at all (see Fig. 7.4(d)). Errors increase gradually away from this zero-distortion locus. At large horizontal eccentricities, depths are distorted to be closer for points in front of the screen, and farther for points behind the screen.

For a 3D point P in the periphery, the rendering positions p_l and p_r generate vertical disparities and do not triangulate to a unique point. Vertical disparities can also arise when the observer is not located at the true center of projection, similar to traditional flat stereo displays [30, 75]. For the plots of Fig. 7.4(a-c), the triangulated point was computed using a least squares fit.

In Fig. 7.4(d), black circular curves with varying radius are shown in the horizontal plane passing by the two eyes. The distortions in red are almost entirely in depth, rather than in direction. In particular, black grid lines are drawn every 5° in eccentricity, but they are hidden by the red grid lines because distortion of visual direction is near zero.

Note that points that are far away from the observer undergo larger depth distortions, but these distortions are not necessarily perceivable. The reason is that the visual system measures disparity, which depends on inverse depth. Large absolute errors in triangulated depth might still produce small disparity changes.

Fig. 7.5 shows the disparity errors for points within a 40° field of view and at various distances from the observer. The disparity errors are zero at an eccentricity of 0° (all curves) and for points on the screen (i.e. blue curve, 230 cm). The disparity errors are in the range of 0 to 6 arcmin up to 20 degrees in eccentricity, whereas the visual direction distortions are less than 0.1 arcsec (data not shown). Again note that the sign of disparity error is opposite for points in front and behind the screen. This

is consistent with Fig. 7.3 and plots in Fig. 7.4 which show that points closer than the screen appear even closer and points behind the screen appear farther.

In Sec. 7.6, we examine whether these depth errors are perceivable. We consider the disparity errors of the triangulated points and compare these disparity errors to the disparity detection thresholds in human vision.

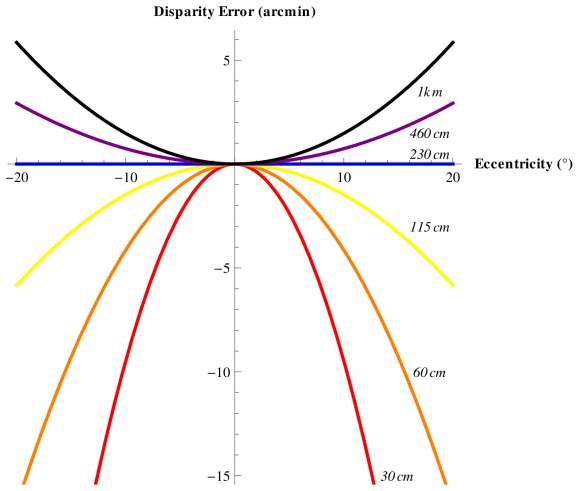


Figure 7.5. For an observer looking at a specific orientation, omnistereo projection distorts points in the periphery of the visual field. This plot shows the computed disparity distortions for points within a 40° field of view at various distances from the observer. Distortion is zero at an eccentricity of 0° and for points on the screen.

7.5 Slit-Camera Projection Model

This section compares distortions of the projection model described in Sec. 7.3 to the more standard slit-camera model.We assume that the cameras have a baseline of width $2b$ centered at and rotating about O. For simplicity, we also suppose for the remainder of this section that the optical axis of both slit-cameras intersect at infinity, i.e. that both cameras are parallel.

Ignoring occlusions, a point P on the median line but not located at infinity is then captured by a V-shaped baseline (see Fig. 7.7(a)). In practice, this means that P is not captured in the same left and right stereo frame. This creates distortion even for points on the median line, in contrast to the median line model presented in Sec. 7.3(b) (see Fig. 7.7(b)).

As a projection model, the screen position p_r is computed by intersecting a line joining the right eye $(b\cos(\alpha), b\sin(\alpha))$ and $P = (0, Z)$, and the circle of radius r centered at the origin. Angle α is given by:

$$\alpha = \frac{\pi}{2} - \arccos(\frac{b}{Z}). \tag{7.5}$$

Fig. 7.6 shows the disparity errors. Observe that the errors are large for very close points to the observer even for an eccentricity of 0°. However, if scene points are limited to points further away than 60 cm, the slit-camera model gives near zero disparity errors.

7.6 Disparity errors versus stereo acuity

On the one hand, since geometric distortions increase with eccentricity, one might expect these distortions to be perceivable at large eccentricities. On the other hand, since the resolution of the visual system decreases with eccentricity, one might expect the distortions not to be perceived. This raises the question of how large the distortions are in comparison to known visual stereo acuity limits, especially in the periphery.

While it is generally agreed that human stereo vision is worse in the periphery, relatively little is known about how performance falls off with eccentricity. Most studies of stereo in peripheral vision only consider eccentricities up to about 10 degrees [33], and classical experiments consider only very simple local tasks such as depth discrimination of thin isolated vertical lines.

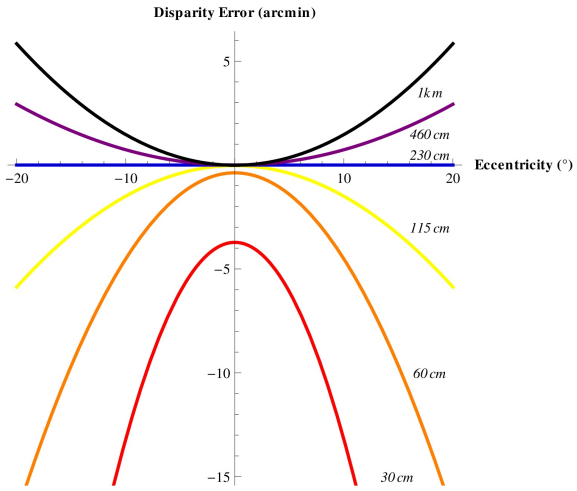


Figure 7.6. Disparity errors for the slit-camera model for points in a 40° field of view at various distances from the observer. Similarly to the median line model, errors are zero for points on the screen (230 cm). However, very near points may be distorted at some distances even if eccentricity is 0°. (In this plot, the eyes are converging at infinity.)

More recent studies have measured perception of more global properties of scene geometry, namely sensitivity to disparity corrugations in the periphery. For example, [54] tested eccentricities up to 20 degrees and, for each eccentricity, they measure the detection threshold of sinusoids of disparity corrugation. They used short presentation times (500 ms), and a fixation point at the center of an annulus that was itself filled with a random dot pattern. They found that peak sensitivity to corrugations was a bandpass function and that for greater eccentricities, peak sensitivity occurred at lower spatial frequencies of the disparity corrugation. The peak detection thresholds themselves increased with eccentricity. For 0, 3.5, 7, 13, and 21 degrees eccentricity, the peak thresholds were about 0.03, 0.3, 0.5, 2, and 5 arc minutes of disparity, respectively. We emphasize that "peak" here refers to the corrugation spatial frequency that was most easily detected.

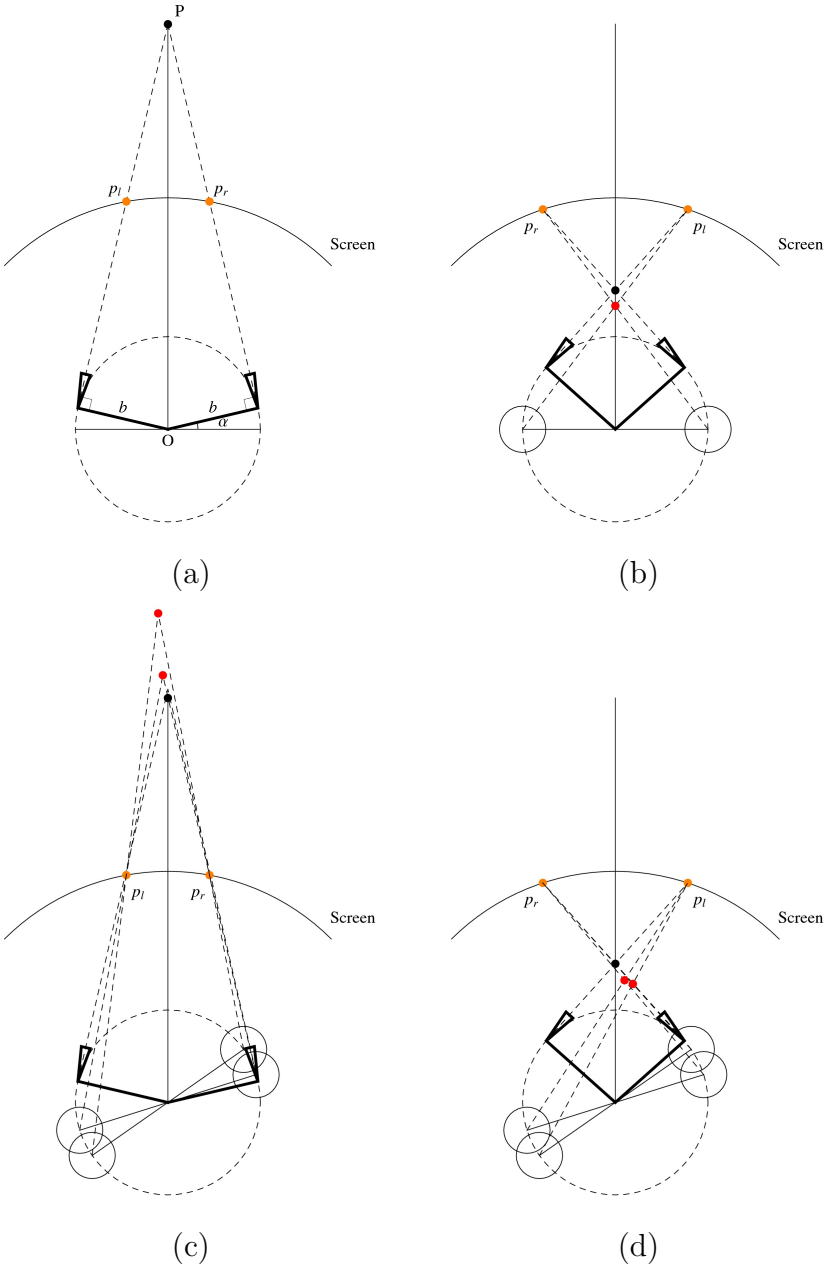


Figure 7.7. (a) The slit-camera projection model projects a point P on the circular screen w.r.t. the position of the left and right cameras having optical axis intersecting P. (b) Point P may appear distorted even if it lies on the median line of the observer. (c-d) Similarly to Fig. 7.3(b-c), P appears at a different location (see red dots) if in the periphery.

The peak thresholds just mentioned are quantitatively similar to the distortions in Fig. 7.5 and Fig. 7.6, provided that the rendered points are at least 1m away from the viewer (yellow curve and above). Of course, one should not attach too great a significance to the similarities of the data – the psychophysical data are dependent on the details of the experiment (observers' task, stimulus, presentation time, definition of threshold, etc). Nonetheless, the similarities do suggest that the geometric distortions introduced by both projection models are at or below the detection threshold, and hence may not be of significance to human observers.

As an aside, the reader may also be interested in knowing why human stereo acuity worsens in the periphery. There is evidence that the high detection thresholds mentioned above are due mainly to the lower resolution of the luminance signal in the left and right eye image, rather than to limitation in stereo processing per se. The luminance signal decreases in the periphery because of factors such as poorer optics, sparser retinal sampling, and greater pooling of photoreceptors by each ganglion cell [9]. Stereo performance worsens in the periphery but no worse than one would expect from the worsening input luminance signals [7, 31]. Indeed, at sufficiently low luminance spatial frequencies – or, equivalently, low dot densities if one is using random dot stereograms – detection thresholds for disparity corrugations in the fovea are similar to those in the periphery. See Fig. 6 of [7], for example.

Finally, we should mention that our discussion of stereo acuity in human vision is far from complete. In particular, we have considered acuity limits on horizontal disparities only. Vertical disparities are also often present and appear to be treated differently than horizontal disparities by the human visual system, for example, in the pooling of information across the visual field [3]. A more complete study of the effects of disparity distortions should consider vertical disparities. For recent reviews of some of the relevant literature, see for example [30, 58].

7.7 Implementation

This section describes a rotation method that leads a point P to be rendered at the screen positions p_l, p_r. The method is similar in flavour to [65] and can be applied with both the median plane and the slit-camera projection models.

We first present the method in 2D. As shown in Fig. 7.8 for the right eye, the method rotates P by θ to get P_r which is rendered with respect to the origin O. Note that angle θ is of opposite sign for the left eye. Screen points p_l and p_r coincide ($\theta = 0$) for scene points P that lie on the screen. Angle θ can be computed by

$$\theta = \arctan(p_{r,z}, p_{r,x}). \tag{7.6}$$

The 3D extension is a rotation θ within the epipolar plane defined by the two eyes and point P.

We implement the method using a vertex shader that rotates each vertex by its corresponding θ, using a positive rotation for the left view and a negative rotation for the right view. For the setup in our lab and for points farther than 100 cm from the observer, the magnitude of rotation θ is less than 1°. Note that this is much less than the rotational shears that are shown in Fig. 7.8, where the baseline is exaggerated.

The two sheared scenes, i.e. for the left and right eyes, are each rendered such that the center of projection is at O. In our vertex shader implementation, the diffuse reflection term is computed using the unmodified vertex and light positions and normals. The specular term for point P is computed by assuming the head is oriented such that P is on the medial plane.

Note that because a vertex shader is applied on vertices and not pixels, the rotational model tends to distort long edges in low tessellated scenes, as only the endpoints (the vertices) are moved correctly. Hence, the rotational model works best for a highly tessellated scene.

One caveat is that we are assuming that any point P that is visible to the both eyes for the real observer will also be visible in both of the rendered images. Our implementation does not guarantee this condition is met, however, since a point at another depth and off the optical axis could in principle be rotated such that it occludes point P in one of the two images.

Finally, we note that the projection model of Sec. 7.3 can be implemented in other ways. For example, rather than rotating about O, a translation parallel to the baseline could be used. This would give rise to a shearing of medial plane that is parallel to the baseline. Since rotations are typically small (less than 1°), this new shear would be near identical to the one produced by rotation. Hence the distortions would be similar as well.

The rotational projection model was tested in a 230 cm radius cylindrical screen, with a height of 150 cm. Four projectors were used to cover half the screen (180 degrees), with neighboring projectors overlapping. Lighttwist [68, 69], an Open Source multi-projector system, automatically aligns the projectors from the point of view of a camera, here at the center of the cylinder screen, without actually reconstructing the screen in 3D as in [56]. High pixel resolution and contrast was achieved at an affordable cost by the use of HD projectors.

For polarized stereo projection, the number of projectors is doubled to eight. The light of the projectors for the left eye is polarized horizontally, and vertically for the right eye. A special screen maintains light polarization, and observers must wear appropriate filtering glasses. Real-time navigation was also successfully achieved by having a rendering computer connected to each projector, synchronized by a master computer that multicasts the joystick input. Navigation was controlled by a single observer.

In practice, the observer might be located off-center, especially if more than one observer is allowed in the omnistereo environment. In this case, perspective distortions arise when the viewer is far from the assumed center of projection [8, 70].

7.8 Conclusion

This paper presented a projection model for rendering omnistereo images from 3D scenes such that the 3D distortions of the scene are zero for points P that are in the center of the field of view, that is, on the head's median plane. The method assumes the observer is standing at the center of the cylindrical screen. 3D distortions were computed and compared to available stereo acuity measurements. The disparity errors from the projection model were found to be near threshold for detection of disparity corrugation at all eccentricities up to 20 degrees. Future work will analyse perspective distortions that result in omnistereo 3D cinema when the viewer is far from the assumed center of projection.

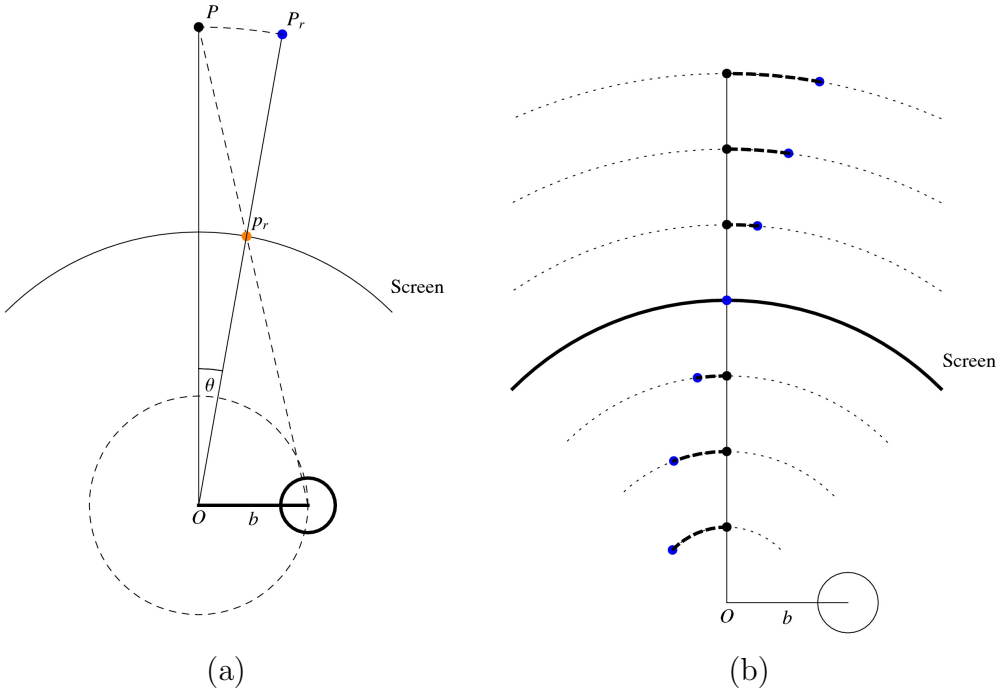


Figure 7.8. (a) Left and right views are rendered from O. Disparities between the two views are created by modifying the position of scene points by a rotation around O. The magnitude of rotation depends on the distance from the eye to the point. Sign of rotation is opposite for the two eyes. The angle of rotation θ is given in Eq. 7.6. Baseline b is typically much smaller than screen diameter and as a result rotations are typically less than 1°. (b) Each point P is relocated by a rotation on a circle of radius $|P|$, and rendered with respect to O. These rotations roughly shear the scene in cylindrical coordinates. This shear produces zero distortion if P is on the observer's medial plane.

DISCUSSION ET CONCLUSION

Le cinéma omnistéréo s'inscrit dans la lignée du cinéma immersif. Il vise à permettre une perception stéréo des profondeurs tout autour de spectateurs. Cette thèse par articles apporte trois contributions en rapport à des aspects du tournage, de la projection et de la perception du cinéma omnistéréo.

Tournage et création de vidéos omnistéréo par assemblage d'images

Le tournage et la création d'images omnistéréo supposaient jusqu'à maintenant une scène statique. Nous avons développé la première méthode de création de vidéos omnistéréo pour des scènes dynamiques. Cette méthode superpose des groupes d'images de façon à créer une vidéo 360° qui peut jouer en boucle sans coupure de mouvement. Nous utilisons un simple dégradé à l'intérieur des zones de superposition. Nous avons mis sur pied une expérience psychophysique qui visait à vérifier la visibilité de ces dégradés. Cette expérience nous permet de conclure que notre méthode fonctionne pour des mouvements sans structure isolée, comme des courants d'eau, mais qu'elle produit des duplications visibles pour des structures isolées en mouvement, comme des branches au vent. Des travaux futurs pourraient remplacer le simple dégradé par une transition optimale entre les groupes d'images de façon à éviter toute duplication [40]. L'optimisation pourrait être indépendante pour chaque zone de chevauchement, ce qui contraste avec les méthodes existantes qui doivent considérer l'entièreté de la séquence vidéo [4, 57].

De plus, notre méthode et les méthodes existantes pour le tournage omnistéréo supposent un trépied qui reste à la même position. Il n'y a donc pas de travelling possible comme au cinéma traditionnel. Des recherches futures pourraient appliquer des notions de vision par ordinateur pour simuler un travelling en créant des vues intermédiaires entre deux images ou vidéos omnistéréo capturées à différents endroits.

Nous avons également proposé une méthode pour le tournage de mouvements non répétitifs, ce qui ouvre la voie au tournage de films omnistéréo. Cette méthode suppose des mouvements à l'intérieur du champ de vision d'une caméra stéréo. Nous l'avons testée en tournant un court métrage à partir d'images de fond omnistéréo fixes. Des travaux futures pourraient développer cette méthode pour permettre l'utilisation d'un fond vidéo omnistéréo dynamique et un tournage moins contraignant.

Finalement, une autre avenue serait l'application de notre observation clé sur la parallaxe à un système composé de plus de deux caméras, par exemple quatre caméras munies de fisheyes. En effet, nous avons vu que deux fisheyes ne peuvent pas capturer en omnistéréo, mais il serait intéressant d'explorer les possibilités d'un système à plus de deux fisheyes, qui pourrait mener à une capture omnistéréo réelle.

Projection sur écran omnistéréo à l'aide de lumière non structurée

Les systèmes multi-projecteur permettent de réaliser à un moindre coût une projection immersive et à grande échelle. Nous avons vu qu'un alignement des projecteurs peut être automatisé par l'utilisation d'une caméra et de motifs lumineux. Les motifs de lumière structurée encodent la position de chaque pixel d'un projecteur, ce qui facilite la mise en correspondance caméra-projecteur pour retrouver la forme d'une scène, par exemple un écran. Cependant, l'utilisation de motifs basse fréquence illumine de larges régions dans la scène créant, entre autres, des sources de lumière secondaires, appelées illumination indirecte. Des méthodes existantes éliminent les basses fréquences, mais conduisent à des ambiguïtés dues à un signal périodique ou à la perte de cohérence locale nécessaire à une robustesse à différents problèmes standard comme le flou ou le désalignement des pixels caméra-projecteur.

Nous proposons de nouveaux motifs basés sur le principe des motifs non structurés qui n'encodent pas directement la position des pixels d'un projecteur, et dont la seule contrainte est qu'une séquence suffisamment grande de motifs identifie chaque pixel de façon unique. Nos motifs sont générés en appliquant un filtre passe-bande sur du

bruit blanc dans le domaine fréquentiel. Ce filtre permet de créer des images dans lesquelles la taille des régions blanches et noires est limitée, ce qui réduit l'effet de l'illumination indirecte. Ces motifs conservent aussi certaines propriétés que nous considérons essentielles pour une robustesse maximale, c'est-à-dire une similitude entre les codes voisins et aucune ambiguïté entre les codes. Puisque les motifs n'encodent pas directement la position des pixels du projecteur, la correspondance entre un code observé dans un pixel de la caméra doit être trouvée dans tous les codes du projecteur. Par exemple, pour un projecteur HD, chaque pixel de caméra doit procéder à une recherche dans environ 2 millions de codes. Pour rendre cette recherche efficace, nous utilisons une méthode probabiliste de hachage itératif avec des heuristiques que nous avons développées pour accélérer la convergence.

Nos résultats ont montré que notre méthode donne d'excellents résultats pour des scènes complexes susceptibles de produire de l'illumination indirecte et d'avoir des discontinuités de profondeur. Notre méthode semble mieux fonctionner que celle de Gupta *et al.*, qui ne réussit pas à retrouver une correspondance correcte lorsque l'illumination indirecte est grande, et qui donne des résultats instables aux discontinuités de profondeur. En ce sens, notre méthode pourrait devenir un standard dans le domaine de la reconstruction active.

De plus, notre méthode ne nécessite aucun calibrage photométrique ou géométrique de la caméra et du projecteur. Cependant, elle requiert un grand nombre de motifs, typiquement entre 100 et 200. Des recherches futures pourraient diminuer le nombre de motifs requis tout en gardant leurs propriétés de base. Une autre avenue serait d'établir une correspondance sous-pixel, c'est-à-dire d'associer un pixel de caméra à une position fractionnaire dans le projecteur. Des méthodes existantes, comme la méthode de déphasage, permettent déjà une telle correspondance, mais elles dépendent de la qualité d'estimation des paramètres photométriques. Nous pourrions prendre avantage du fait que notre méthode profite de l'ajout d'un nombre illimité de motifs pour parvenir à une précision sous-pixel sans calibrage photométrique.

Perception des distorsions de profondeur

Pour un spectateur au centre de l'écran cylindrique, nous avons vu que les distorsions de profondeur causées par deux modèles de projection omnistéréo sont minimes au centre du champ visuel et s'accentue à la périphérie. Cependant, nous avons montré que ces distorsions de profondeur sont du même ordre que les limites d'acuité du système visuel humain. Les distorsions de profondeur sont considérées négligeables pour un spectateur au centre de notre écran cylindrique, en autant que les points de la scène virtuelle sont au-delà d'un mètre autour du spectateur.

Il est à noter que les données d'acuité visuelle que nous avons utilisées dépendent des paramètres de l'expérience psychophysique décrite dans [54]. Ces paramètres, comme le temps d'affichage ou la fréquence du stimulus, ne sont pas nécessairement les mêmes lors du visionnement d'un film, par exemple. De plus, le contexte d'un visionnement diffère de celui d'une expérience psychophysique où un observateur doit accomplir une tâche de détection. Néanmoins, les similitudes entre les seuils de détection visuelle et les distorsions omnistéréo suggèrent que celles-ci sont négligeables. Nous avons montré également que, pour un spectateur non centré, les distorsions deviennent non nulles même au centre du champ visuel. Mais il en est de même pour un spectateur qui n'est pas assis au centre d'une salle de cinéma [8, 70].

La méthode de création de vidéos omnistéréo présentée au chapitre 2 s'éloigne du modèle omnistéréo traditionnel, puisqu'elle utilise des images complètes et non des fentes. Cette méthode peut introduire même au centre du champ visuel de légères distorsions de profondeur, qui s'accentuent avec l'élargissement de l'angle de vue des caméras. Plus particulièrement, elle introduit des distorsions lorsqu'une zone de chevauchement entre deux groupes d'images se situe au centre du champ visuel.

Finalement, des recherches futures pourraient s'intéresser à l'analyse des distorsions verticales, en particulier pour des écrans plus hauts ou des dômes.

RÉFÉRENCES

[1] http://vision3d.iro.umontreal.ca/en/projects/omnistereo/.

[2] http://vision3d.iro.umontreal.ca/en/projects/unstructured-light-scanning/.

[3] W. Adams, J.P. Frisby, D. Buckley, J. Garding, S.D. Hippisley-Cox et J. Porrill : Pooling of vertical disparities by the human visual system. *Perception*, 25(2):165–176, 1996.

[4] A. Agarwala, K.C. Zheng, C. Pal, M. Agrawala, M. Cohen, B. Curless, D. Salesin et R. Szeliski : Panoramic video textures. *ACM Transactions on Graphics*, 24(3):821–827, 2005.

[5] A. Andoni : Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. Dans *IEEE Symposium on Foundations of Computer Science*, pages 459–468, 2006.

[6] A. Andoni et P. Indyk : Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Communications of the ACM*, 51(1): 117–122, 2008.

[7] M.S. Banks, S. Gepshtein et M.S. Landy : Why is spatial stereoresolution so low? *The Journal of Neuroscience*, 24(9):2077–2089, mars 2004.

[8] M.S. Banks, R.T. Held et A.R. Girshick : Perception of 3-d layout in stereo displays. *Information Display*, 25(1):12–16, 2009.

[9] M.S. Banks, A.B. Sekuler et S.J. Anderson : Peripheral spatial vision: limits imposed by optics, photoreceptors, and receptor pooling. *Journal of the Optical Society of America A*, 8(11):1775–1787, 1991.

[10] P. Bourke : Synthetic stereoscopic panoramic images. *Lecture Notes in Computer Science (VSMM)*, 4270:147–155, 2006.

[11] P. Bourke : Omni-directional stereoscopic fisheye images for immersive hemispherical dome environments. Dans *Computer Games and Allied Technology (CGAT)*, pages 136–143, Singapore, mai 2009.

[12] K.L. Boyer et A.C. Kak : Color-encoded structured light for rapid active ranging. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 9(1):14–28, janvier 1987.

[13] R.N. Bracewell : *The Fourier Transform and Its Applications*. McGraw-Hill, 1965.

[14] D. Caspi, N. Kiryati et J. Shamir : Range imaging with adaptive color structured light. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, Jan 1998.

[15] V. Couture, M. S. Langer et S. Roy : Capturing non-periodic omnistereo motions. Dans *Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras (OMNIVIS)*, Zaragoza, Spain, 2010.

[16] V. Couture, M. S. Langer et S. Roy : Panoramic stereo video textures. Dans *IEEE International Conference on Computer Vision (ICCV)*, Barcelona, Spain, novembre 2011.

[17] C. Cruz-Neira, D.J. Sandin et T.A. DeFanti : Surround-screen projection-based virtual reality: the design and implementation of the cave. Dans *ACM Conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 135–142, New York, NY, USA, 1993.

[18] C. Cruz-Neira, D.J. Sandin, T.A. DeFanti, R.V. Kenyon et J.C. Hart : The cave: audio visual experience automatic virtual environment. *Communications of the ACM*, 35(6):64–72, 1992.

[19] J. Davis, D. Nehab, R. Ramamoorthi et S. Rusinkiewicz : Spacetime stereo: A unifying framework for depth from triangulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(2):296–302, février 2005.

[20] W.J. Dixon et A. M. Mood : A method for obtaining and analyzing sensitivity data. *Journal of the American Statistical Association*, 43:109–126, 1948.

[21] G. Doretto, A. Chiuso, Y.N. Wu et S. Soatto : Dynamic textures. *International Journal of Computer Vision*, 51(2):91–109, février 2003.

[22] M.A. Fischler et R.C. Bolles : Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[23] A. Gionis, P. Indyk et R. Motwani : Similarity search in high dimensions via hashing. Dans *International Conference on Very Large Data Bases (VLDB)*, pages 518–529, San Francisco, CA, USA, 1999.

[24] J. Gluckman, S.K. Nayar et K.J. Thoresz : Real-time omnidirectional and panoramic stereo. Dans *DARPA Image Understanding Workshop*, pages 299–303. Morgan Kaufmann, 1998.

[25] L. Goddyn et P. Gvozdjak : Binary gray codes with long bit runs. *Electronic Journal of Combinatorics*, 10:27, 2003.

[26] S.J. Gortler, R. Grzeszczuk, R. Szeliski et M.F. Cohen : The lumigraph. Dans *ACM Conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 43–54, New York, NY, USA, 1996.

[27] J. Gühring : Dense 3-d surface acquisition by structured light using off-the-shelf components. *Videometrics and Optical Methods for 3D Shape Measurement*, janvier 2001.

[28] M. Gupta, A. Agrawal, A. Veeraraghavan et S.G. Narasimhan : Structured light 3d scanning in the presence of global illumination. Dans *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 713–720, 2011.

[29] R.I. Hartley et A. Zisserman : *Multiple View Geometry in Computer Vision*. Cambridge University Press, second édition, 2004.

[30] R.T. Held et M.S. Banks : Misperceptions in stereoscopic displays: a vision science perspective. Dans *Symposium on Applied perception in graphics and visualization (APGV)*, pages 23–32, New York, NY, USA, 2008.

[31] R.F. Hess, F.A.A. Kingdom et L.R. Ziegler : On the relationship between the spatial channels for luminance and disparity processing. *Vision Research*, 39(3):559–568, février 1999.

[32] D.M. Hoffman, A.R. Girshick, K. Akeley et M.S. Banks : Vergence-accommodation conflicts hinder visual performance and cause visual fatigue. *Journal of Vision*, 8(3):33, 2008.

[33] I.P. Howard et B.J. Rogers : *Seeing in Depth*. Oxford University Press, USA, 2002.

[34] H. Huang et Y.-P. Hung : Panoramic stereo imaging system with automatic disparity warping and seaming. *Graphical Models and Image Processing*, 60(3): 196–208, 1998.

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

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

[37] H. Ishiguro, M. Yamamoto et S. Tsuji : Omni-directional stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 14(2): 257–262, 1992.

[38] F. A. Kingdom et N. Prins : *Psychophysics: A Practical Introduction*. Academic Press, 2010.

[39] A. Kushnir et N. Kiryati : Shape from unstructured light. Dans *3DTV07*, pages 1–4, 2007.

[40] Vivek Kwatra, Arno Schödl, Irfan Essa, Greg Turk et Aaron Bobick : Graphcut textures: Image and video synthesis using graph cuts. *ACM Transactions on Graphics*, 22(3):277–286, juillet 2003.

[41] M. Levoy et P. Hanrahan : Light field rendering. Dans *ACM Conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 31–42, New York, NY, USA, 1996.

[42] H.C. Longuet Higgins et K. Prazdny : The interpretation of a moving retinal image. *Royal Society of London*, B-208:385–397, 1980.

[43] D. Marquardt : An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics (SIAP)*, 11:431–441, 1963.

[44] B. Mendiburu : *3D Movie Making: Stereoscopic Digital Cinema from Script to Screen*. Focal Press, 2009.

[45] T. Naemura, M. Kaneko et H. Harashima : Multi-user immersive stereo. Dans *IEEE International Conference on Image Processing*, volume 1, page 903, Los Alamitos, CA, USA, 1998.

[46] S.K. Nayar, A. Krishnan, M.D. Grossberg et R. Raskar : Fast separation of direct and global components of a scene using high frequency illumination. *ACM Transactions on Graphics*, 25:935–944, 2006.

[47] P. Peers, D.K. Mahajan, B. Lamond, A. Ghosh, W. Matusik, R. Ramamoorthi et P. Debevec : Compressive light transport sensing. Dans *ACM Transactions on Graphics*, volume 28, pages 3:1–3:18, New York, NY, USA, février 2009.

[48] S. Peleg et M. Ben-Ezra : Stereo panorama with a single camera. Dans *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, page 1395, Los Alamitos, CA, USA, 1999.

[49] S. Peleg, M. Ben-Ezra et Y. Pritch : Omnistereo: Panoramic stereo imaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 23(3):279–290, 2001.

[50] P. Pérez, M. Gangnet et A. Blake : Poisson image editing. Dans *ACM Conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 313–318, New York, NY, USA, 2003.

[51] P. Perez, M. Gangnet et A. Blake : Poisson image editing. Dans *ACM Conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 313–318, New York, NY, USA, 2003.

[52] Point Grey Research : *Ladybug3*, 2008.

[53] J. Posdamer et M. Altschuler : Surface measurement by space-encoded projected beam systems. *Computer Graphics and Image Processing*, janvier 1982.

[54] S.J.D. Prince et B.J. Rogers : Sensitivity to disparity corrugations in peripheral vision. *Vision Research*, 38(17):2533–2537, 1998.

[55] M. Proesmans, L. Van Gool et A. Oosterlinck : One-shot active 3d shape acquisition. Dans *International Conference on Pattern Recognition*, volume 3, pages 336 – 340 vol.3, 1996.

[56] R. Raskar, M.S. Brown, R. Yang, W.-C. Chen, G. Welch, H. Towles, B. Seales et H. Fuchs : Multi-projector displays using camera-based registration. Dans *IEEE Visualization Conference*, Washington, DC, USA, 1999.

[57] A. Rav-Acha, Y. Pritch, D. Lischinski et S. Peleg : Dynamosaicing: Mosaicing of dynamic scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(10):1789–1801, 2007.

[58] J. Read et B. Cumming : Does depth perception require vertical-disparity detectors? *Journal of Vision*, 10(6), 2010.

[59] J. Salvi, J. Batlle et E. Mouaddib : A robust-coded pattern projection for dynamic 3d scene measurement. *Pattern Recognition Letters*, janvier 1998.

[60] J. Salvi, J. Pagès et J. Batlle : Pattern codification strategies in structured light systems. *Pattern Recognition*, 37:827–849, 2004.

[61] D. Scharstein et R. Szeliski : High-accuracy stereo depth maps using structured light. Dans *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 195–202, juin 2003.

[62] A. Schödl, R. Szeliski, D.H. Salesin et I. Essa : Video textures. Dans *ACM Conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 489–498, New York, NY, USA, 2000.

[63] S.M. Seitz, A. Kalai et H.Y. Shum : Omnivergent stereo. *International Journal of Computer Vision*, 48(3):159–172, juillet 2002.

[64] J. Shi et C. Tomasi : Good features to track. Dans *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 593–600, 1994.

[65] A. Simon, R.C. Smith et R.R. Pawlicki : Omnistereo for panoramic virtual environment display systems. Dans *IEEE Virtual Reality Conference*, page 67, Los Alamitos, CA, USA, 2004.

[66] N. Snavely, S.M. Seitz et R. Szeliski : Photo tourism: Exploring photo collections in 3d. Dans *ACM Conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 835–846, New York, NY, USA, 2006.

[67] R. Szeliski : *Computer Vision : Algorithms and Applications*. Springer-Verlag New York Inc, 2010. voir pp. 398-400.

[68] J.-P. Tardif et S. Roy : A mrf formulation for coded structured light. Dans *International Conference on 3-D Digital Imaging and Modeling (3DIM)*, pages 22–29, Washington, DC, USA, 2005.

[69] J.-P. Tardif, S. Roy et M. Trudeau : Multi-projectors for arbitrary surfaces without explicit calibration nor reconstruction. Dans *International Conference on 3-D Digital Imaging and Modeling (3DIM)*, pages 217–224, Los Alamitos, CA, USA, 2003.

[70] D. Vishwanath, A.R. Girshick et M.S. Banks : Why pictures look right when viewed from the wrong place. *Nature Neuroscience*, 8(10):1401–1410, septembre 2005.

[71] P. Vuylsteke et A. Oosterlinck : Range image acquisition with a single binary-encoded light pattern. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 12(2):148–164, février 1990.

[72] G. B. Wetherill et H. Levitt : Sequential estimation of points on a psychometric function. *The British journal of Mathematical and Statistical Psychology*, 18:1–10, 1965.

[73] Y. Wexler, A.W. Fitzgibbon et A. Zisserman : Learning epipolar geometry from image sequences. Dans *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, page 209, Los Alamitos, CA, USA, 2003.

[74] H. Woeste : *Mastering Digital Panoramic Photography*. Rocky Nook, 2009.

[75] A. Woods, T. Docherty et R. Koch : Image distortions in stereoscopic video systems. Dans *Stereoscopic Displays and Applications IV*, volume 1915, pages 36–48, 1993.

[76] C. Wust et D. Capson : Surface profile measurement using color fringe projection. *Machine Vision and Applications*, Janvier 1991.

[77] Yi Xu et D.G. Aliaga : An adaptive correspondence algorithm for modeling scenes with strong interreflections. *IEEE Transactions on Visualization and Computer Graphics*, 15:465–480, 2009.

[78] L. Zhang, B. Curless et S.M. Seitz : Rapid shape acquisition using color structured light and multi-pass dynamic programming. Dans *International Symposium on 3D Data Processing Visualization and Transmission (3DPVT)*, pages 24 – 36, 2002.

[79] S. Zhang et S. Yau : High-speed three-dimensional shape measurement system using a modified two-plus-one phase-shifting algorithm. *Optical Engineering*, Jan 2007.

[80] S. Zhang et S.-T. Yau : High-resolution, real-time 3d absolute coordinate measurement based on a phase-shifting method. *Optics Express*, 14(7): 2644–2649, 2006.

Annexe A

CALCUL DE LA PARALLAXE

Soit un pixel p en une position générale dans une caméra de référence située à 0° (voir Fig. A.1). Le pixel p est ensuite projeté à des distances de z_{min} et l'infini pour obtenir les points 3D de la scène $P_{z_{min}}$ et P_∞. Nous utilisons $z_{min} = 2$m. Ces points $P_{z_{min}}$ et P_∞ sont ensuite reprojetés à différentes orientations de caméra θ dans un intervalle allant de $-FOV$ à FOV, où FOV est l'angle de vue de la caméra. La *parallaxe* de p à l'orientation θ est définie comme étant la distance pixel dans l'image entre ces deux reprojections. La parallaxe dépend donc de p et θ. Dans les figures qui suivent, nous ne considérons que les valeurs de θ pour lesquelles les deux reprojections tombent à l'intérieur de l'image.

Par construction, il n'y pas de parallaxe lorsque $\theta = 0$ puisqu'il s'agit de la position de la caméra de référence. Aussi, la Fig. A.1 montre qu'il n'y a aucune parallaxe *horizontale* à l'orientation $\theta = \theta_0$ lorsque la caméra se retrouve sur la ligne qui passe par $P_{Z_{min}}$ et P_∞. Cependant, il est à noter qu'il y a parallaxe *verticale* lorsque cette ligne n'est pas coplanaire avec la trajectoire circulaire des caméras. L'angle θ_0 se calcule à partir des deux relations suivantes :

$$\gamma = \alpha_r + \beta \tag{A.1}$$

et

$$\pi = \alpha_s + \beta + \gamma \tag{A.2}$$

où α_r est l'orientation angulaire de p par rapport à l'axe optique de la caméra de référence à $\theta = 0°$, et α_s est l'orientation angulaire de la ligne passant par P_∞ et p par rapport à l'axe optique de la seconde caméra à $\theta = \theta_o$. Subtituant (A.1) dans

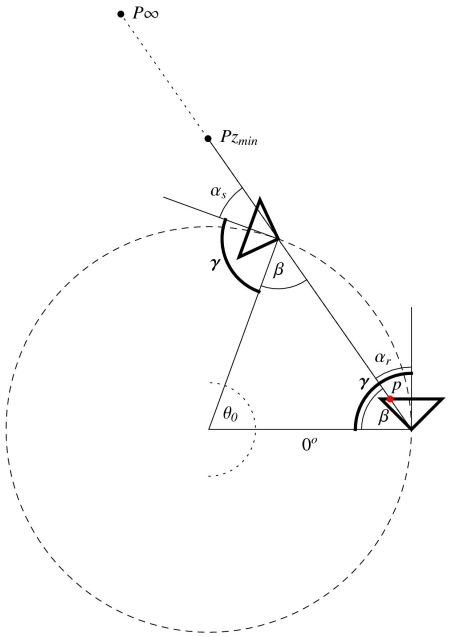


Figure A.1. La parallaxe est définie en prenant un pixel p dans une caméra de référence et en le projetant aux profondeurs Z_{min} et l'infini pour obtenir les points $P_{Z_{min}}$ et P_{∞}. Ces deux points 3D sont ensuite reprojetés dans le plan image de la caméra à une orientation θ relative à la caméra de référence, la parallaxe étant la distance image entre ces deux reprojections. Dans l'exemple montré, l'angle de vue est de 90°. Il n'y a pas de parallaxe horizontale à $\theta = 0^{\circ}$ et à $\theta = \theta_0$, c'est-à-dire lorsque la caméra se retrouve sur la ligne qui passe par $P_{Z_{min}}$ et P_{∞}. Voir l'équation A.3 pour une dérivation de θ_0. Il est à noter que $P_{Z_{min}}$ et P_{∞} ne sont pas à l'échelle par rapport au reste de la figure.

(A.2) donne :

$$\begin{aligned}
\pi = \alpha_s + \beta + \alpha_r + \beta &\Rightarrow \pi - 2\beta = \alpha_r + \alpha_s \\
&\Rightarrow \theta_o = \alpha_r + \alpha_s \\
&\Rightarrow \theta_o = \alpha_r + \pi - \beta - \gamma && \text{à partir de (2)} \\
&\Rightarrow \theta_o = 2\alpha_r + \pi - 2\gamma && \text{à partir de (1)}
\end{aligned}$$

Finalement, on obtient que

$$\theta_o = 2\alpha_r + \nu \tag{A.3}$$

où ν est la vergence stéréo, définie comme l'angle entre l'axe optique des deux caméras qui forment la paire stéréo.

Il est à noter que $\theta_0 = FOV$ lorsque $\alpha_r = \frac{FOV}{2}$, ce qui mène à l'observation clé du chapitre 1.

Annexe B

AUTRES RÉSULTATS DE L'EXPÉRIENCE PSYCHOPHYSIQUE

À la section 2.5.2, nous avons mentionné que deux participants n'avaient pas réussi le test de vision stéréo et que leurs données n'avaient pas été incluses dans les résultats de l'article. La figure B.1 montre leurs résultats. Bien que leurs seuils soient légèrement plus haut, nous pouvons tout de même observer qu'ils ont réussi à détecter les zones de chevauchement, du moins pour les scènes de végétations (`Flowers` et `Bush`). Ceci confirme que la détection des zones de chevauchements est essentiellement une tâche monoculaire.

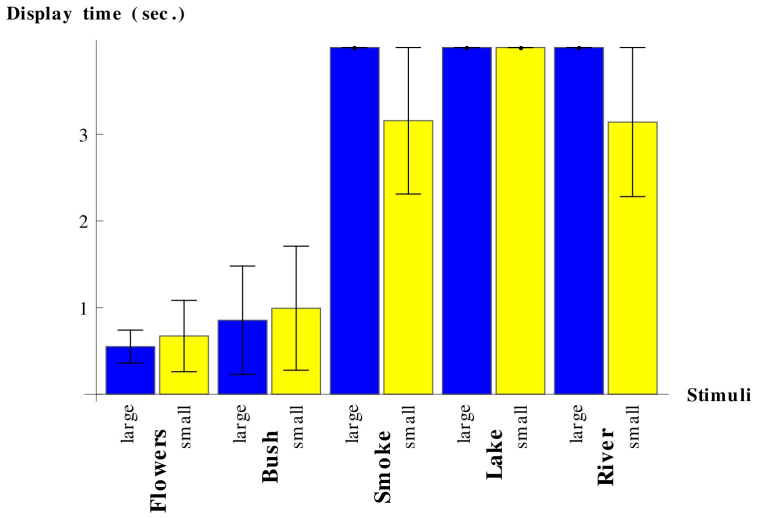


Figure B.1. Seuils de détection pour l'expérience B des deux participants qui n'ont pas réussi le test de vision stéréo.