Titre: Title:	Systèmes de stéréovision passive dédiés à un stimulateur intra-cortical visuel
Auteur: Author:	Firas Hawi
Date:	2011
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Hawi, F. (2011). Systèmes de stéréovision passive dédiés à un stimulateur intra-cortical visuel [Mémoire de maîtrise, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/690/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/690/
Directeurs de recherche: Advisors:	Mohamad Sawan
Programme: Program:	génie électrique

UNIVERSITÉ DE MONTRÉAL

SYSTÈMES DE STÉRÉOVISION PASSIVE DÉDIÉS À UN STIMULATEUR INTRA-CORTICAL VISUEL

FIRAS HAWI

DÉPARTEMENT DE GÉNIE ÉLECTRIQUE

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE EN VUE DE L'OBTENTION

DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES

(GÉNIE ÉLECTRIQUE)

NOVEMBRE 2011

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé:

SYSTÈMES DE STÉRÉOVISION PASSIVE DÉDIÉS À UN STIMULATEUR INTRA-CORTICAL VISUEL

Présenté par: HAWI Firas

en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées

a été dûment accepté par le jury d'examen constitué de :

M. PLAMONDON Réjean, Ph. D., président

M. SAWAN Mohamad, Ph. D., membre et directeur de recherche

M. BRAULT Jean-Jules, Ph. D., membre

REMERCIEMENTS

Je tiens à remercier en premier lieu M. Mohamad Sawan, professeur à l'École Polytechnique de Montréal, pour m'avoir accueilli dans l'équipe Polystim et m'avoir offert l'opportunité de travailler sur un sujet aussi passionnant et stimulant. Je remercie également les membres du jury M. Réjean Plamondon et M. Jean-Jules Brault, professeurs à l'École Polytechnique de Montréal, pour avoir accepté de faire partie du jury d'examen de ce mémoire.

Je remercie mes parents, mon frère et mes sœurs pour leur support moral et encouragements tout au long de mes études.

J'adresse mes remerciements aussi aux membres de l'équipe Polystim qui ont toujours montré de l'initiative et de la responsabilité collective au laboratoire. Je pense à Amine, Sami, Gotham, Ali, Romain, Yushan, Arash, Robert, Ahmad, Fayçal, Réjean, Marie, Laurent et Saeid. Je tiens à remercier Roula pour m'avoir introduit au projet de l'implant intra-cortical visuel de l'équipe et Anthony pour plusieurs discussions enrichissantes.

Enfin, je remercie les techniciens Jacques Girardin et Bryan Tremblay pour leur support dans la conception de PCB et plusieurs conseils dans les travaux de soudure et autres.

RÉSUMÉ

L'objectif de ce mémoire est de concevoir un système de stéréovision passif pour inférer la profondeur de champ d'une scène. Le système est destiné à fournir une information utile aux non-voyants sur l'environnement qui les entoure. Il doit fonctionner en temps réel et garantir de bonnes précisions.

Le premier chapitre présente des méthodes de base ainsi qu'une revue de littérature qui présente différentes catégories de méthodes qu'on peut employer pour inférer la profondeur. Un système à base d'une chaîne de Markov cachée est présenté. Il permet de mettre en correspondance les pixels des images issues de deux caméras, étape essentielle pour l'inférence de profondeur en stéréovision passive. Nous analysons les points de faiblesse du système et présentons une autre méthode graphique, la propagation de conviction qui est plus stable. Une autre catégorie de méthodes qui sont plus rapides est aussi présentée. En particulier, on met l'emphase sur la corrélation de phase.

Le deuxième chapitre présente deux méthodes à base de corrélation de phase. La première méthode est une amélioration de la précision de la corrélation de phase unidimensionnelle (Enhanced Phase Only Correlation, EPOC) et la deuxième est une optimisation de la corrélation de phase bidimensionnelle (Multiple Modes Phase Only Correlation, MMPOC). La corrélation de phase bidimensionnelle est plus précise que la version unidimensionnelle, mais moins rapide. Une description d'une implémentation d'EPOC sur FPGA est fournie.

Le troisième chapitre fait une discussion sur EPOC dans un contexte graphique. Une comparaison entre différentes approches se fait dans un contexte Bayesien.

Le quatrième chapitre présente deux systèmes stéréoscopiques à base de mémoire associative quantique de Hopfield. Le premier système est une architecture sérielle qui emploie le réseau quantique de Hopfield, le deuxième système est une version parallèle de cette architecture. Le but de ce chapitre est d'expliquer qu'une mémoire associative possède la capacité d'inférer la profondeur à partir d'une paire d'images stéréoscopiques. Les points de faiblesse de ces méthodes sont comparés à ceux du cerveau humain.

ABSTRACT

The main objective of the present Master thesis is to design a passive stereovision system to be used in the cortical visual stimulators application. The system aims to provide blind patients with information on the spatial positioning of objects that are present in a given environment. Main requirements are real-time operability and high disparity estimation accuracy.

The first chapter presents different approaches to solve the correspondence search problem, a main step to infer depth with passive stereovision systems. The first system is based on a Hidden Markov Model. We review the drawbacks of the proposed system and present a more stable approach that exists in the literature, Belief Propagation. We also present faster approaches to solve the correspondence problem, with a special attention on phase correlation.

The second chapter presents two phase based approaches to solve the correspondence problem. The first system (Enhanced Phase Only Correlation, EPOC) is an enhancement of the 1D version of the Phase Only Correlation. The second system (Multiple Modes Phase Only Correlation, MMPOC) optimizes the 2D Phase Only Correlation method. The 2D Phase Only Correlation method is known to be more accurate than the 1D version, but takes a longer time to execute. A real-time implementation of the EPOC algorithm on a FPGA is described.

The third chapter discusses in more detail the EPOC system in a graphical framework. Key systems comparison is made in a Bayesian framework.

In the fourth chapter, we design two Hopfield based quantum associative memories that have the ability to infer depth from two images, the serial architecture and parallel architecture of Hopfield Quantum network. A comparison relates the drawbacks of the proposed systems with results of experimentations that were performed on the human brain.

TABLE DES MATIERES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES SIGLES ET ABRÉVIATIONS

1D	Unidimensionnel / Une dimension
2D	Bidimensionnel / Deux dimensions
1DPOC	One dimensional phase only correlation
2DPOC	Two dimensional phase only correlation
BSME	Bloc de superposition des espaces de recherche et de minimisation d'énergie
BP	Belief propagation
EPOC	Enhanced phase only correlation
FA	Forward Algorithm
FIFO	First in, first out
FPGA	Field programmable gate array
HMM	Hidden Markov Model
ICM	Iterated Conditional Modes
MMPOC	Multiple modes phase only correlation
MRF	Markov random Field
PC	Phase correlation
POC	Phase only correlation
SAD	Sum of absolute differences
SSD	Sum of squared differences

LISTE DES ANNEXES

INTRODUCTION

L'imagerie tridimensionnelle est un sujet qui réfère à une partie de plus en plus intégrante dans divers systèmes médicaux, médiatiques, industriels et logistiques. Les développements en imagerie artificielle en générale et l'imagerie tridimensionnelle en particulier ont été motivés par le besoin de créer des machines plus sensibles, plus intelligentes et plus utiles dans le cadre d'une compétition industrielle. Dans ce cadre, la stéréovision passive est un domaine fortement divergent. Le manque d'une formulation complète et unique au problème de la stéréovision passive est à l'origine de cette divergence. Loin de cet environnement compétitif, mais faisant partie de cette divergence, il y a des scientifiques qui se sont intéressés à étudier le système visuel humain dans le but d'accroître notre connaissance sur le cerveau. Jusqu'aux années 60, on croyait qu'il existait une partie du cerveau qui est chargée de faire à la fois la compréhension des objets contenus dans une scène, estimer leur dimensions et leur positionnement. Une série d'expériences (Schneider 1967) chez des animaux a révélé que l'identification des objets et leur localisation se fait dans deux parties différentes du cerveau.

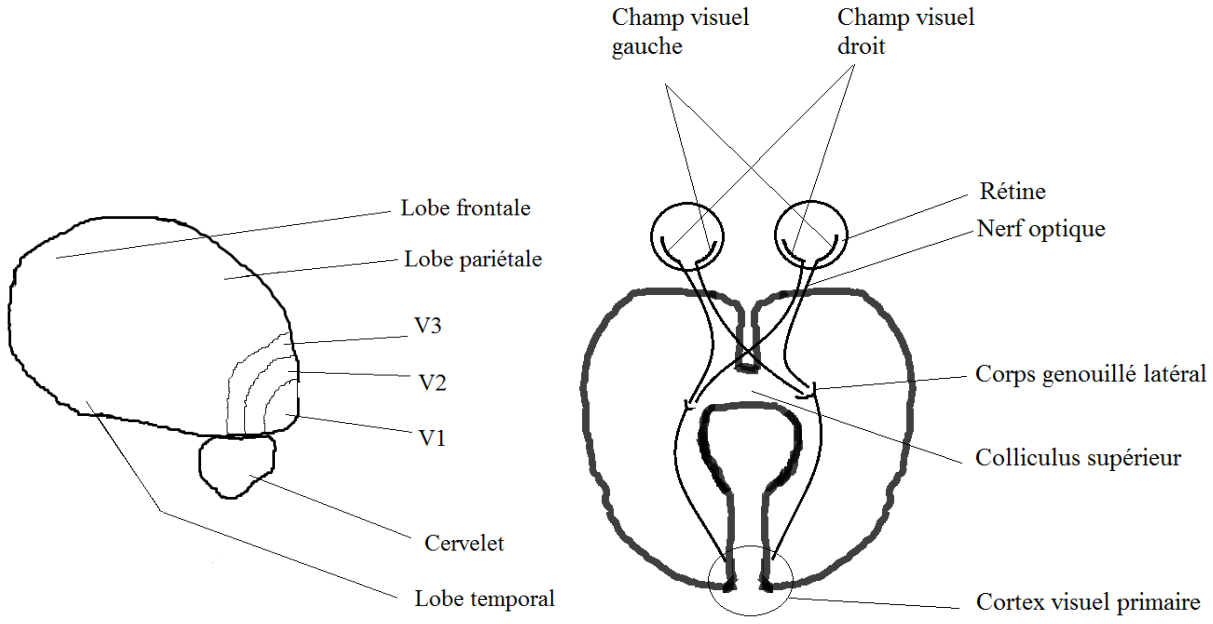


Figure 1.1 Anatomie du cerveau (a) et visualisation du parcours du nerf optique (b)

D’après (Ungerleider and Mishkin 1982), la partie du cerveau chargée de traiter le positionnement des objets dans une scène s’étale sur la région supérieure du cerveau de la région V1 jusqu’au lobe frontale et est appelée voie dorsale. La partie du cerveau chargée d’identifier les objets s’étend de la partie V1 jusqu’au lobe temporal (Figure 1.1 (a)), c’est la voie ventrale. Au niveau de la rétine, la lumière est convertie en potentiels d’actions qui sont transportés par le nerf optique au cerveau. Ce qui est remarquable dans Figure 1.1 (b), c’est que chacun des lobes (gauche et droit) du cerveau reçoit, au niveau du corps genouillé latéral, des signaux provenant des deux yeux. Le corps genouillé latérale envoie par la suite l’information visuelle à la partie V1, mais reçoit aussi de la partie V1 des signaux de façon rétroactive.

Le laboratoire Polystim est en train de développer un implant qui permettra aux non voyants d’avoir une perception visuelle de l’environnement qui les entoure. Il est connu que des stimulations électriques faites au niveau du cortex visuel engendrent la perception de points lumineux (Brindley and Lewin 1968). La position du point lumineux perçu par le patient dépend de l’endroit où la stimulation a été appliquée. Ainsi, en faisant des stimulations électriques aux endroits appropriés, on peut reconstruire le champ de vision chez l’individu. Une puce électronique implantée sur le cortex du cerveau, plus précisément sur la voie dorsale, est chargée de faire ces stimulations (ROY 1999). Elle reçoit l’image à transmettre au patient via un lien sans fil d’une unité de traitement d’image externe (Gervais 2004) et stimule des neurones responsables de la perception de la vision à travers des électrodes (Figure 1.2).

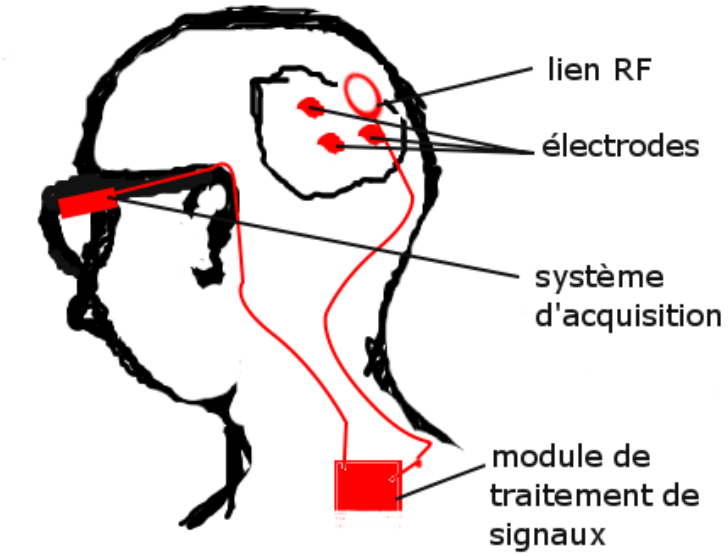
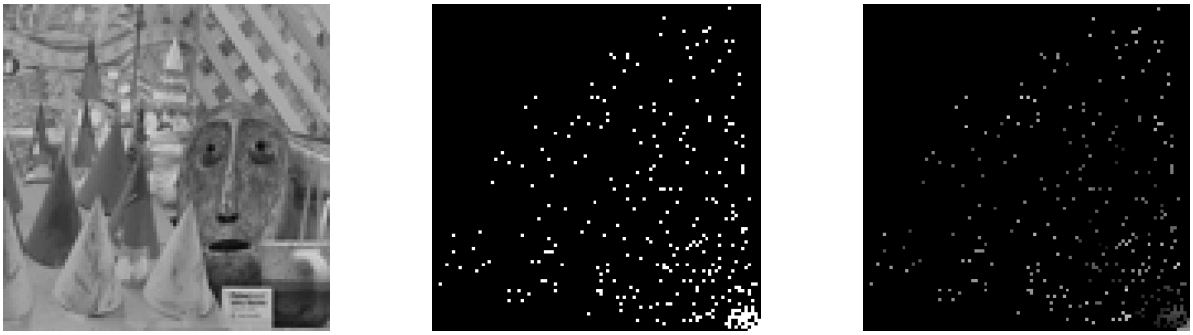


Figure 1.2 Schéma Visualisant les composantes principales du stimulateur

Les points lumineux que le patient percevra, aussi appelés phosphènes, sont repartis selon une carte qui est propre à chaque patient. La répartition des phosphènes dépend de la position des neurones qui sont en contact avec la grille d'électrodes. Une série de tests visuels effectués sur le patient après l'implant est essentielle pour pouvoir identifier l'emplacement des phosphènes stimulés. Un exemple de carte de phosphènes est visualisé dans Figure 1.3.

Jusqu'à ce jour, il n'a pas été possible de contrôler la couleur du point perçu au niveau neuronal. On transmet par la grille d'électrodes une image à niveaux de gris. En plus, l'image perçue par le patient est discontinue et difficile à traiter (Buffoni, J.Coulombe et al. 2003). Un patient qui perçoit une image comme celle visualisée dans Figure 1.3 (c) ne peut saisir l'information contenue dans la scène dans Figure 1.3 (a) avec le niveau de détail essentiel à l'estimation de la distance relative des objets et à la détection d'obstacles à proximité. En fait, l'inférence de la profondeur du champ par le cerveau se fait en mettant en correspondance des éléments de l'image saisie par l'œil de gauche et ceux de l'image saisie par l'œil de droite. Le fait que l'image transmise par l'implant est discontinue rend difficile cette mise en correspondance. Ceci a mené des chercheurs à développer des systèmes d'acquisition 3D afin de fournir aux patients une information sur la profondeur de champ qui leur sera plus utile dans la vie courante. Ainsi, au lieu de transmettre une information relative à l'intensité de la couleur des objets de la scène, on transmet une information reliée directement à la profondeur de champ. Le niveau de gris du phosphène sera élevé pour identifier des objets qui sont proches du patient et faible pour les objets éloignés. Ainsi, des systèmes actifs basés sur le principe de lumière structurée (Delia 2007) ont été développés. Ces systèmes permettent d'avoir une image de profondeur de haute précision et en temps réel.

Toutefois, les systèmes actifs comprennent des lacunes qui rendent leur utilisation peu pratique dans notre application. Parmi ces lacunes, la portée d'un système actif est reliée aux capacités énergétiques disponibles. Que ce soit un système à lumière structurée ou basé sur le calcul de temps de vol, l'onde émise doit atteindre sa cible en gardant un niveau d'amplitude considérable pour être détectée par les récepteurs après réflexion sur les objets de la scène. Pour détecter des objets de plus en plus loin, l'amplitude de l'onde émise doit être amplifiée. L'augmentation de l'amplitude de l'onde émise nécessite des capacités énergétiques plus grandes, ce qui nécessite des batteries plus grandes et plus lourdes. Par la suite, la compacité du système actif sera touchée, ainsi que sa portabilité.



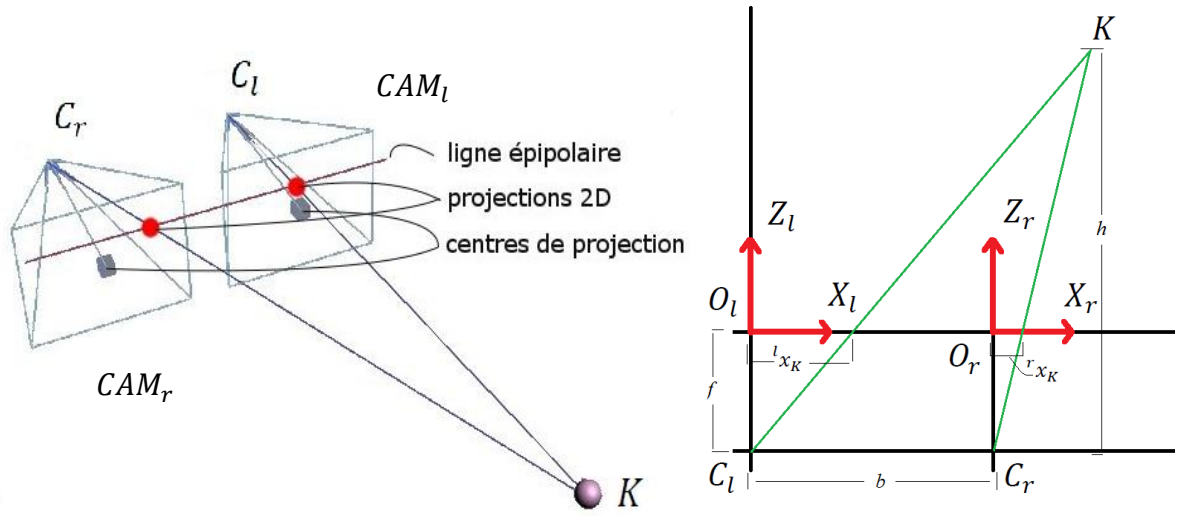
(a) (b) (c)

Figure 1.3 Dans (a), on montre une image saisie d'une camera, dans (b) une carte de 300 phosphène et dans (c) l'image (a) vue à partir des phosphènes visualisés dans (b). La procédure de génération de cartes de phosphènes est expliquée dans (Buffoni, Coulombe et al. 2003).

D'autre part, l'utilisation d'un système actif d'imagerie 3D en milieu publique n'est pas pratique. Dans le cas d'un système à lumière structurée qui envoie des ondes dans le domaine visible, le système doit projeter des patrons lumineux qui peuvent déranger d'autres personnes ou même subir une interférence par d'autres systèmes d'imagerie de même type présents dans le milieu.

Pour ces raisons, il est intéressant d'explorer l'emploi des systèmes passifs. Bien que leur utilisation soit plus acceptable en public, les systèmes de stéréovision passive éprouvent une difficulté à maintenir de bonnes précisions lorsqu'elles sont destinées à fonctionner en temps réel. Un système passif de reconstruction 3D est constitué essentiellement de plusieurs caméras qui font la capture d'une scène au même moment et d'une unité de traitement d'images dont le rôle est de savoir pour chaque pixel d'une image reçue d'une caméra donnée, son pixel correspondant dans les images fournies par les autres cameras. On appelle cette dernière étape la mise en correspondance.

L'image d'un objet dans une caméra va être placée en fonction de l'angle de vue de cette caméra. L'angle de vue est l'angle formé par le centre de projection, centre de caméra et l'objet 3D. Ainsi, si on sait de combien l'image d'un objet a été décalée entre deux vues, on peut connaître la distance qui sépare l'objet d'une caméra donnée par triangulation (Figure 1.4).



(a) (b)

Figure 1.4 Système de stéréovision passive (a) et géométrie de la stéréovision passive (b).

Les lignes épipolaires sont déduites par calibration. Elles permettent de réduire l'espace de recherche du correspondant d'un pixel donné d'une image à un vecteur unidimensionnel. Typiquement, lorsque deux caméras identiques faisant partie d'un système stéréoscopique sont parfaitement alignées horizontalement, un pixel appartenant à une ligne i de la première image doit avoir son correspondant quelque part sur la ligne i de la deuxième image comme illustré dans Figure 1.4 (a).

Si on place deux caméras pour faire la capture d'une scène de façon simultanée et qu'on observe la projection des objets de la scène sur les plans de projection des cameras (écran), on remarque que les projections sont translatées les uns aux autres d'un montant inversement proportionnel à la distance qui sépare les objets de la scène du système de caméras.

Soient deux caméras CAM_l et CAM_r ayant comme centres C_l et C_r et centres de projection O_l et O_r. On utilise les contraintes épipolaires tel qu'illustré dans Figure 1.4 (a). Supposons que ^lx_K est la position du pixel de CAM_l qui est associée au pixel ^rx_K de CAM_r. On désigne par (X_l, Z_l) le système de coordonnées centré sur O_l dans lequel est défini ^lx_K. Similairement, (X_r, Z_r) est le système de coordonnées centré sur O_r dans lequel est défini ^rx_K. Si on note par h la distance d'un objet par rapport au système de caméras et par b la distance qui sépare les deux caméras stéréoscopiques, on a d'après Figure 1.4 (b):

$$h = \frac{b}{f\left|\ {}^{l}x_K\ -\ {}^{r}x_K\ \right|} \tag{1.1}$$

Ici, f est la distance focale qui est considérée de même valeur pour les deux caméras. Elle peut être déduite par calibration. En particulier, on remarque que

$$h \propto \frac{1}{\left|\ {}^{l}x_K\ -\ {}^{r}x_K\ \right|} \tag{1.2}$$

Et on note :

$$d_K = \left|\ {}^{l}x_K\ -\ {}^{r}x_K\ \right| \tag{1.3}$$

Ici, d_K s'appelle la disparité. C'est un paramètre-clé dans tout système de stéréovision passive qu'on prend soin de définir.

La disparité est le montant de translation entre un pixel d'une image stéréoscopique et le pixel correspondant dans l'autre image. Elle est mesurée en pixels. Afin d'obtenir une image de profondeur à partir de deux caméras, il faut connaître pour chaque pixel d'une image stéréoscopique, la disparité correspondante. Pour y arriver, il faut connaître deux mesures. Ces mesures sont les positions de deux pixels qui sont les images d'un point de la scène sur les deux caméras. La position de chacun des deux pixels est mesurée dans le système de référence de la caméra qui contient le pixel en question. Dans l'exemple de Figure 1.4 (b), le système de référence de la mesure ${}^{l}x_K$ est centré sur O_l, le centre de projection de CAM_l. Alors que le système de référence de la position du pixel correspondant dans l'autre caméra, ${}^{r}x_K$, est centré sur O_r, le centre de projection de CAM_r. La disparité est la valeur absolue de la différence de ${}^{l}x_K$ et ${}^{r}x_K$.

Pour construire l'image de disparités, il est utile de formuler le problème de sorte à identifier dans les images stéréoscopiques l'information pertinente à l'inférence de disparités.

On assume que les contraintes épipolaires sont utilisés pour réduire l'espace de recherche de 2D à 1D. Soient deux signaux S_1 et S_2 ayant chacun N échantillons. Les échantillons $s_1^n(t)$ et $s_2^n(t)$ des signaux S_1 et S_2 sont définis tels que:

$$S_i(t) = \sum_{n=1}^{N} s_i^n(t), \quad s_i^n(t) = \begin{cases} S_i(t) & si\ t = n \\ 0 & autrement \end{cases} \tag{1.4}$$

Les échantillons $s_1^n(t)$ et $s_2^n(t)$ sont regroupés dans des groupes d'échantillons $R_1^{m,a_m,b_m}(t)$ et $R_2^{m,a_m,b_m}(t)$. Dans la notation $R_i^{m,a,b}(t)$, on note par m le numéro du groupe d'échantillons, par a l'indice spatial qui définit le début du signal et par b l'indice spatial qui indique sa fin. Pour deux groupes d'échantillons successifs $R_i^{m,a_m,b_m}(t)$ et $R_i^{m+1,a_{m+1},b_{m+1}}(t)$, on a $a_{m+1} = (b_m)+1$. Ainsi, les groupes d'échantillons sont définis par :

$$R_i^{m,a_m,b_m}(t) = \sum_{n=a}^{b} s_i^n(t) \tag{1.5}$$

Au lieu de décrire le signal S_1 par une série d'échantillons, on va le décrire par des ensembles d'échantillons, tels que:

$$S_1(t) = \sum_{m=1}^{M} R_1^{m,a_m,b_m}(t) \tag{1.6}$$

Où M correspond au nombre d'ensembles d'échantillons $R_i^{m,a,b}(t)$ nécessaires pour la description du signal S_i et $M < N$.

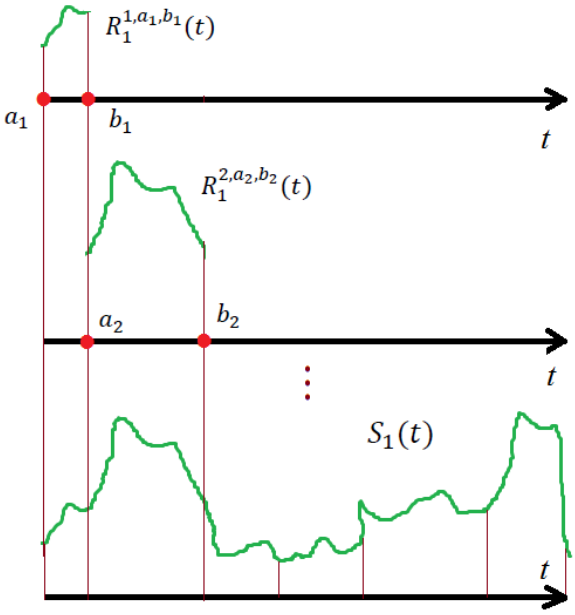


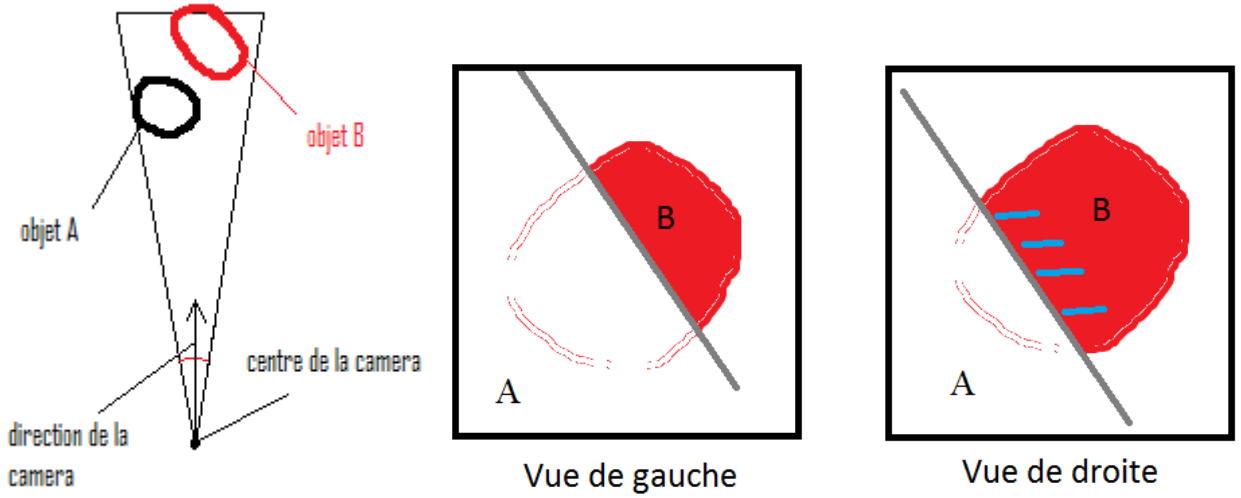
Figure 1.5 Illustration d'un signal $S_1(t)$ en régions $R_1^{m,a_m,b_m}(t)$.

Dans un problème de stéréovision passive où il y a deux signaux tirés de deux lignes épipolaires de la vue de gauche et de celle de droite, on souhaite savoir les paramètres de la transformée qui permet de passer de S_2 à S_1 et qui peut prendre la forme suivante :

$$S_1 = T(S_2) \tag{1.7}$$

$$\sum_{m=1}^{M} R_1^{m,a_m,b_m}(t) = \sum_{m=1}^{M} \alpha^m R_2^{m,a_m,b_m}\left(\frac{t - \beta^m \gamma^m}{\gamma^m}\right) \tag{1.8}$$

Le paramètre α sert à prendre en compte de l'occlusion. Si un objet est en partie caché par un autre, des régions de cet objet seront visibles à une camera mais pas à l'autre. Dans ce cas, la région qui est invisible se fait attribuer le paramètre $\alpha^m = 0$, la région visible aura un $\alpha^m = 1$. Figure 1.14 illustre une région en occlusion.



(a) (b) (c)

Figure 1.6 Illustration de l'obstruction d'un objet (B) par un autre objet (A) par rapport à une caméra donnée (a). Dans (b) et (c) est illustré l'effet de l'occlusion. Les deux images stéréoscopiques d'un objet partiellement caché ne peuvent pas être obtenues l'une de l'autre par une simple translation. Dans (c), la zone hachurée n'est pas visible dans (b).

Le paramètre γ est l'étalement spatial. Un objet qui est oblique à l'angle de vue du système stéréoscopique, i.e. qui est étendu sur plusieurs niveaux de profondeur, aura une image dans une vue qui est étalée sur une région de largeur différente de celle de son image dans l'autre vue comme montré dans Figure 1.7. L'introduction de ce paramètre se fait avec l'hypothèse que les objets de la scène sont plats.

Le paramètre β est le paramètre essentiel. Il s'agit du montant de translation entre deux pixels ou groupes de pixels. Il s'agit aussi de la disparité. Ce paramètre se rattache directement à la profondeur d'un objet par rapport au système stéréoscopique. Un objet qui est proche a une disparité qui est grande alors qu'un objet éloigné possède une disparité qui est petite.

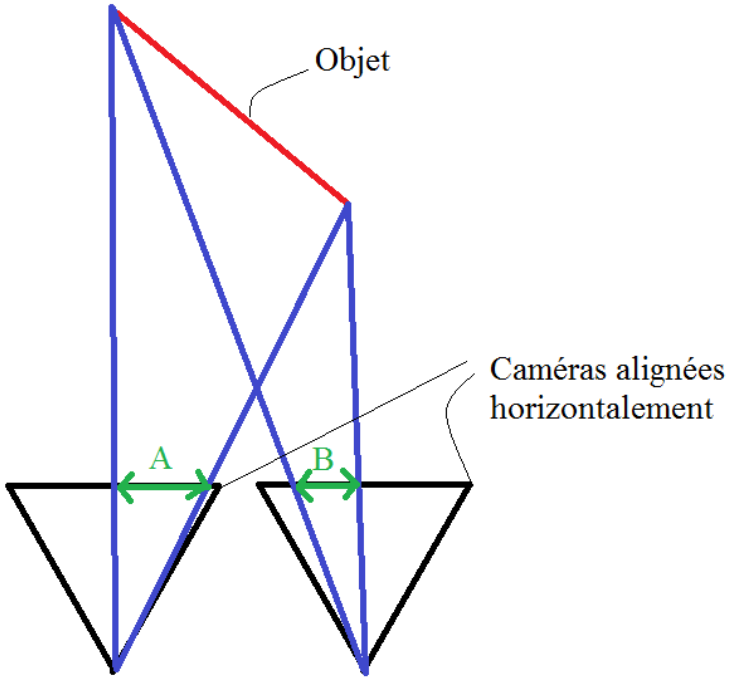


Figure 1.7 Illustration de la configuration d'un objet de la scène qui est oblique par rapport au système de stéréovision binoculaire passif. La largeur A de la projection de l'objet sur une caméra est différente de la largeur B de la projection de cet objet sur la deuxième caméra.

En plus d'estimer les paramètres α, β et γ, on doit estimer les étendues des régions $R_i^{m,a,b}(t)$ et prendre en compte le bruit de cameras et bruit d'échantillonnage. C'est-à-dire, il faut estimer les paramètres a et b de chaque ensemble d'échantillons, ainsi que le nombre d'ensemble d'échantillons M. L'estimation des étendues des régions peut se faire de plusieurs manières, dépendamment du contexte algorithmique du système de stéréovision passive. Avant de présenter un exemple simple d'estimation des régions, précisons qu'une région est l'image d'une portion d'un objet tel que les couleurs qui constituent cette portion d'objet sont très similaires. Ainsi, on fait l'hypothèse que deux pixels voisins qui appartiennent au même objet auront éventuellement des couleurs semblables. Ayant donné un pixel de référence, la segmentation a

pour but d'extraire d'une fenêtre donnée tous les pixels qui appartiennent au même objet à lequel appartient ce pixel de référence. En prenant la valeur de la couleur du pixel d'intérêt comme couleur de référence, on parcourt toute la fenêtre d'étude pour ne retenir que les pixels dont la couleur est à une distance qui est inferieure à un certain seuil arbitraire μ de la couleur de référence, i.e. :

$$\left|f(p_i) - f(p_j)\right| < \mu \qquad (1.9)$$

Où p_i est le pixel de référence, normalement situé au centre de la fenêtre d'étude de l'image stéréoscopique de référence. Le terme $p_j$avec $j \in I$, I désignant l'espace des indices de tous les pixels appartenant à la fenêtre d'étude, réfère au pixel d'indice j. La fonction $f(.)$ retourne la couleur (pour simplifier, on peut utiliser une échelle de gris) du pixel en paramètre. La valeur du seuil μ dépend de la luminosité de la scène et peut être déterminé expérimentalement.

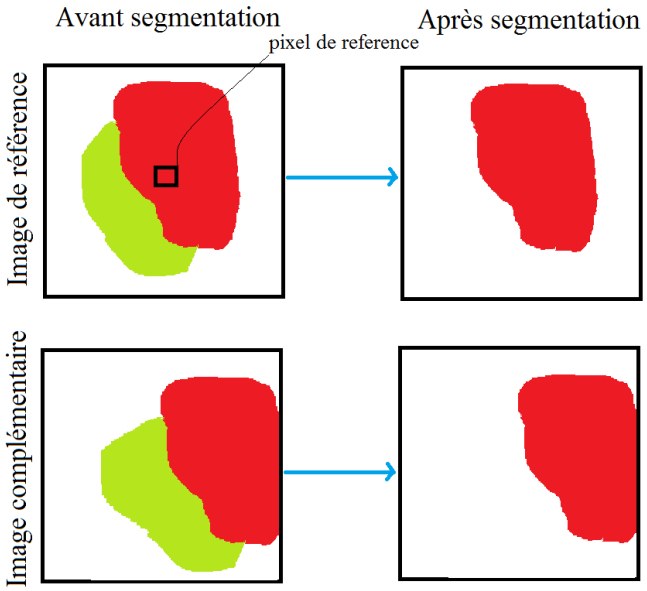


Figure 1.8 Extraction d'objet rapide en se basant sur la couleur. À gauche on voit l'image avant extraction d'objet avec en encadré le pixel de référence, à droite, le résultat après extraction.

Figure 1.8 illustre le mécanisme utilisé pour l'extraction rapide de l'objet, le pixel de référence est encadré dans l'image de référence. La valeur de la couleur utilisée pour faire l'extraction d'objets dans la deuxième image est la même que celle utilisée pour la première (image de référence).

L'extraction se fait ainsi en associant à tous les pixels identifiés en tant que n'appartenant pas à l'objet qui contient le pixel de référence la valeur 0, et pour tous les pixels qui appartiennent à cet objet de référence, on garde la valeur qui leur est déjà associée, i.e. la valeur qui reflète leur couleur (en fond de gris).

Vu le nombre de paramètres, la formulation en (1.8) semble difficile à résoudre, surtout en temps réel. La résolution du problème de mise en correspondance se fait par des approximations à cette formulation ou des reformulations plus simples.

Les systèmes de pointe sont aujourd'hui comparés à l'aide de bases de données, dont la plus importante est celle de *Middlebury* (Scharstein and Szeliski 2001). Dans cette base de données, il y a des paires d'images stéréoscopiques qui sont fournies à la communauté. *Middlebury* dispose pour chaque paire d'image stéréoscopique d'une scène, une image de profondeur calculée à partir d'un système de stéréovision active à base de lumière structurée. Ces images de profondeur sont converties en images de disparités. Les chercheurs téléchargent les paires d'images stéréoscopiques pour construire les images de disparités correspondantes utilisant leurs propres algorithmes. Ces images de profondeur sont soumises à *Middlebury* qui va calculer des taux d'erreurs pour chaque image de disparité soumise en utilisant ses propres images de disparité calculés par la méthode de lumière structurée comme référence. Le taux d'erreur d'une image de disparité $S_{prop}(i,j)$ qui contient N pixels est obtenu en utilisant l'image de disparité $S_{idéale}(i,j)$ déduite par un système à base de lumière structurée comme suit:

$$\epsilon = \frac{\sum_{(i,j)} g\left(\left|S_{prop}(i,j) - S_{idéale}(i,j)\right|, \theta\right)}{N} \times 100\ (\%) \tag{1.10}$$

Avec

$$g(\psi, \theta) = \begin{cases} 1\ si\ \psi > \theta \\ 0\ si\ \psi < \theta \end{cases} \tag{1.11}$$

Où θ est un seuil qui vaut normalement 1. Dans ce présent mémoire, Tous les taux d'erreurs sont basés sur l'équation (1.10) et admettent la valeur du seuil $\theta = 1$.

Middlebury calcule trois taux d'erreurs par image, le premier est calculé sur les régions qui ne sont pas en occlusion, le deuxième est calculé sur les frontières d'objets, incluant éventuellement des régions en occlusion et le troisième taux d'erreur est calculé sur toute l'image. Afin de comparer les algorithmes des chercheurs, on requiert de faire la reconstruction 3D de 4 paires

d'images spécifiques. Les taux d'erreurs sont ainsi comparés. La base de données de *Middlebury* a été éliminatoire au fil du temps. Les 4 paires d'images utilisées pour comparer les algorithmes ont été changées, de sorte que les évaluations sur la base de données la plus récente soient faites sur les systèmes de pointe selon les versions de la base de données qui précèdent. *Middlebury* n'évalue pas le temps d'exécution des algorithmes, mais seulement le taux d'erreur sur les disparités.

Nous nous intéressons à mettre en œuvre un dispositif qui peut fournir aux patients une information sur la distance qui les séparent des objets présents dans le milieu, avec la capacité de faire la distinction entre les objets qui sont proches et ceux qui sont à l'arrière plan. Ainsi, on ne s'intéresse pas à une mesure métrique, mais plutôt à une mesure qui se rattache directement au positionnement relatif des objets. Il est possible de voire dans l'équation (1.1) que la disparité est directement reliée à la distance h. Notons que b et f sont des constantes, donc l'information essentielle au calcul de la distance relative est la disparité. L'obtention de la distance à partir de la disparité nécessite une division, donc plus de ressources, et la connaissance de la distance focale f qui requiert une routine de calibration. Nous avons donc choisi de construire un système dont le but est de générer une carte de disparités afin de l'envoyer au patient à travers l'implant intra-cortical.

Dans un système de stéréovision passive, on peut avoir à calculer la valeur de disparité au niveau de chacun des pixels de l'image, dans ce cas on parle de *stéréovision dense*. Dans une application qui requiert de calculer des valeurs de disparité au niveau de seulement quelques pixels de l'image, on parle de *stéréovision dispersée.* Notre application est une application de stéréovision dispersée puisque nous voulons calculer la disparité au niveau de quelques centaines de phosphènes reparties dans l'image (Figure 1.3 (b)).

Pour définir les exigences de précision, il faut se reposer sur des critères cohérents. L'erreur de précision sur les niveaux de gris qui reflètent la profondeur de champ ne doit pas empêcher le patient à classifier ensemble les phosphènes qui correspondent au même objet. Cette mesure est de nature esthétique puisque cette information est destinée à être interprétée par le cerveau et non pas par une machine qui requiert un taux d'erreur calculable. Pour attribuer un nombre à cette exigence de précision, on se réfère à la base de donnée *Middlebury* (Scharstein and Szeliski 2001). La raison c'est qu'en plus d'être une référence mondiale en stéréovision, cette base de

données a été éliminatoire pour contenir aujourd'hui les méthodes de pointe en stéréovision binoculaire passive, c.à.d. les méthodes qui ont les plus petits taux d'erreurs de reconstruction 3D avec des scènes qui contiennent des occlusions. La plus pire méthode a actuellement une erreur de 20% (Scharstein and Szeliski 2001). Ainsi, nous posons comme exigence que l'algorithme doit avoir un taux d'erreur supérieur à 20%. En d'autres termes, notre système doit fournir des images de meilleure qualité que la plus pire méthode retenue par *Middlebury.* Ce taux d'erreur est calculé en utilisant l'équation (1.10) avec $\theta = 1$.

Une autre exigence est la capacité du système à détecter les bordures des objets à différents niveaux de disparité. Une dernière exigence est l'opérabilité en temps réel. Dans les applications vidéo, un taux de rafraichissement de 30 images par seconde est suffisant.

Plusieurs méthodes existent pour résoudre le problème de mise en correspondance. Il est utile d'identifier deux catégories de celles-ci. La première catégorie regroupe des méthodes appelées méthodes *globales*. Elle comprend des algorithmes à haute complexité qui permettent d'avoir de bonnes précisions de localisation d'un objet dans une scène (Scharstein and Pal 2007) au prix d'un temps de calcul élevé (Kim, Park et al. 2009). Elles sont surtout constituées des méthodes graphiques (Bishop 2006) tels que la propagation de conviction (Weinman, Tran et al. 2008). Ces méthodes sont itératives et consomment beaucoup de mémoire, leur implémentation en matériel sous leur version actuelle n'est donc pas recommandée.

La deuxième catégorie est celle des méthodes *locales*. Les méthodes *locales* sont basées sur des notions de base en analyse des signaux. Bien que leur opérabilité en temps réel fût démontrée (Morikawa, Katsumata et al. 1999), leurs précisions ne sont toutefois pas encourageantes (Hirschmuller, Innocent et al. 2002). En fait, sous leurs versions brutes, leurs précisions se détériorent rapidement lorsqu'il y a des effets d'occlusions.

D'une part, les méthodes locales sont plus pratiques à implémenter en temps réel, et des implémentations sur FPGA ont vu le jour (Niitsuma and Maruyama 2010). Alors qu'une implémentation d'une méthode globale en temps réel peut exiger des ressources plus dispendieuses en matériel et en énergie (GPU), ainsi que des compromis sur le nombre possible de niveaux de disparités (Wang, Liao et al. 2007).

D'autre part, les meilleures méthodes (en termes de taux d'erreur) selon *Middlebury* sont présentement de nature graphique. Des méthodes basées sur la cross-corrélation ont été éliminées

du classement de *Middlebury* malgré leurs performances temporelles supérieures. Une forme plus robuste de cross-corrélation est la corrélation de phase (Kuglin and Hines 1975). Récemment, une version améliorée de la corrélation de phase avait subit le test de *Middlebury* et n'avait pas été retenue (Alba and Arce-Santana 2009), avec un taux d'erreur moyen de 27.6%.

Dans ce mémoire, nous sommes intéressés par les méthodes locales parce qu'on veut un système qui fonctionne en temps réel et qui soit portable. Nous étudions quand même les méthodes graphiques pour tirer profit de leurs avantages et identifier leurs points de faiblesse dans les applications de stéréovision *dispersée*. Nous arrivons à augmenter la précision de la corrélation de phase unidimensionnelle pour aboutir à un taux d'erreur significativement meilleure que celui de la dernière version existante (Alba and Arce-Santana 2009). Nous optimisons aussi la corrélation de phase bidimensionnelle pour améliorer son temps d'exécution. Une implémentation sur FPGA prouve l'opérabilité en temps réel d'un des algorithmes élaborés dans une application dédiée à l'implant intra-cortical visuel.

La corrélation et les méthodes graphiques sont dérivées de formulations mathématiques qui n'ont pas nécessairement de relation directe avec le fonctionnement cérébrale ou avec des architectures neuronales présentes dans le cerveau. Dans notre application, nous sommes en quelque sorte concernés, parce que l'algorithme élaboré va être utilisé dans un implant qui vise à remplacer une partie du cerveau, qui est celle responsable de l'inférence de profondeur. C'est donc favorable de situer notre travail dans le contexte du fonctionnement cérébrale. Nous montrons que la corrélation de phase explique comment une mémoire associative permet d'inférer la profondeur. Ainsi, deux autres systèmes permettant de résoudre le problème de mise en correspondance en utilisant des mémoires associatives sont introduits.

Dans le chapitre 1, une revue de littérature est présentée ainsi qu'un système de base basé sur une chaîne de Markov cachée. Ce chapitre couvre quelques méthodes globales et locales. La propagation de conviction y est présentée, ainsi que la corrélation de phase. Les points de force et de faiblesse des méthodes globales et locales sont exhibés.

Dans chapitre 2, la corrélation de phase est présentée en plus de détails. La version améliorée de la corrélation de phase unidimensionnelle (Enhanced Phase Only Correlation – EPOC) y est expliquée, ainsi que la version optimisée de la corrélation de phase bidimensionnelle (Multiple Modes Phase Only correlation – MMPOC). Les deux architectures sont comparées. EPOC est

implémentée sur FPGA. Ce chapitre représente un article soumit pour publication dans la revue Computer Vision and Image Understanding (Hawi and Sawan 2011).

Dans chapitre 3, une discussion plus approfondie de l'algorithme EPOC est présentée. Une comparaison entre EPOC et un autre système de la même classe de méthodes qui est présent dans la littérature (Alba and Arce-Santana 2009) est faite.

Dans chapitre 4, deux systèmes à base de mémoire associative de Hoppfield sont utilisés pour inférer la profondeur. Ces systèmes servent à démontrer l'importance de la phase dans l'inférence de profondeur dans une mémoire associative. Ainsi, la corrélation de phase sera mieux mise dans le contexte neuronal.

CHAPITRE 1 SYSTÈMES DE BASE ET REVUE DE LITTERATURE

Dans ce premier chapitre, nous présentons des méthodes utilisées pour résoudre le problème de mise en correspondance en stéréovision passive. Nous avons fait la distinction de méthodes globales et méthodes locales. Dans la première partie de ce chapitre, nous introduisons les méthodes graphiques qui sont des méthodes globales. Ainsi, un système à base d'une chaîne de Markov cachée ou HMM (pour Hidden Markov Model) sera présenté, suivi d'un model plus complet régi par la propagation de conviction, ou BP (Belief Propagation). Dans la deuxième partie, nous introduisons les méthodes locales, surtout la corrélation de phase. La capacité des deux groupes de méthodes, globales et locales, à répondre aux exigences de notre application particulière sera mise en question dans la conclusion.

1.1 Méthodes globales

En stéréovision passive, les méthodes globales, principalement de nature graphique sont à l'origine des états de l'art selon *Middlebury* (Scharstein and Szeliski 2001). Nous commençons par reformuler le problème de mise en correspondance d'une manière plus adaptée aux méthodes graphiques. Ensuite, nous présentons une méthode de construction de cartes de disparités en se basant sur une chaîne de Markov cachée avant de faire une ouverture sur la littérature et présenter une méthode plus robuste, la propagation de conviction.

1.1.1 Adaptation du problème au contexte graphique

Dans l'introduction, nous avons formulé d'une manière générale le problème de mise en correspondance. Dans la formulation faite dans l'équation (1.8), on doit regrouper les pixels en des régions, chercher les montant de translation et d'étalement spatiale au niveau de chaque région et faire correspondre les régions $R_1^{m,a_m,b_m}(t)$ de la première image stéréoscopique avec les régions $R_2^{m,a_m,b_m}(t)$ de la deuxième image stéréoscopique en prenant en compte de l'occlusion. Afin de réduire le nombre de variables et obtenir une formulation plus simple à résoudre dans un contexte graphique, nous simplifions dans ce qui suit cette formulation. En particulier, on doit fixer à priori les régions $R_1^{m,a_m,b_m}(t)$ de la première image en définissant leur étendus. Pour ceci, on pose comme contrainte que $b_m = a_m + 1$. En d'autres termes, chaque groupe de pixels sera restreint à un seul pixel. De cette manière, on aurait éliminé le problème du

regroupement des pixels en des régions. Par contre, on aurait augmenté le nombre d'évaluations de sorte que les paramètres de translation et d'étalement spatiale auront à être estimés au niveau de chaque pixel et non pas au niveau de seulement quelques régions. Mais aussi, vu que le paramètre de l'étalement spatial γ^m est défini sur plusieurs pixels (par définition même de l'étalement spatial), et que nous avons restreint les régions $R_i^{m,a_m,b_m}(t)$ à un pixel chacun, nous pouvons éliminer ce paramètre. Ainsi, on présente la nouvelle définition des régions:

$$R_i^{m,m,m+1}(t) = s_i^m(t) \tag{1.12}$$

Une nouvelle formulation du problème peut être écrite sous la forme suivante:

$$\sum_{m=1}^{M} s_1^m(t) = \sum_{m=1}^{M} \alpha^m s_2^m(t - \beta^m) \tag{1.13}$$

Où $M = N$, N étant le nombre de pixels sur une ligne quelconque d'une image stéréoscopique. Ainsi, le problème sera de trouver pour une série de pixels successifs $s_1^m(t)$, la série équivalente $s_2^m(t - \beta^m)$ en trouvant les valeurs du paramètre de translation β^m qui permettent de faire correspondre chaque pixel de la première image à un pixel de l'autre image. Dans cette formulation, nous prenons aussi en compte de l'occlusion avec le paramètre α^m.

Concentrons-nous sur les montants de translation β^m et mettons le terme α^m à part pour le moment. Ces montants de translation sont les inconnues du système. Les entités observables sont s_1^m et s_2^m, i.e. les pixels des deux images stéréoscopiques. L'équation (1.13) dit que si on dispose d'une ligne d'une image $s_2^m(t)$ et les valeurs de disparités correspondantes, représentées par les montants de translation $\beta^m(t)$ au niveau de chaque pixel de la ligne $S_2(t)$, on peut déduire le contenu de la ligne épipolaire de l'autre image $S_1(t)$. De façon équivalente, on peut déduire $S_2(t)$ à partir de $S_1(t)$ et de β^m. Nous rappelons qu'on utilise les contraintes épipolaires et que l'espace de recherche est restreint de ce fait à deux lignes de pixels $S_1(t)$ et $S_2(t)$ au lieu de deux images. Ainsi, on peut identifier dans cette formulation un processus caché (β^m) qui prend en entrée une ligne d'une image stéréoscopique et en sortie, il produit le contenu de l'autre image stéréoscopique. Le problème est de trouver ce processus caché, en ayant en notre disposition le contenu des deux lignes épipoliares $S_1(t)$ et $S_2(t)$. Ce problème ressemble à un problème de prédiction d'un processus caché qui réside derrière l'observation d'une séquence observable, un problème de HMM.

1.1.2 Inférence de profondeur avec une chaîne de Markov cachée

Dans une chaîne de Markov, normalement on dispose de données observables et on peut déduire une matrice de transition pour prédire les échantillons futurs d'une séquence de données. Avec les chaînes de Markov cachées, le processus derrière la production d'une séquence observable est caché. Chaque état caché peut produire un état observable donné avec une certaine probabilité. Une bonne revue des chaînes de HMM est donnée dans (Cappé, Moulines et al. 2005).

Dans cette partie nous faisons la conception d'un algorithme de résolution du problème de mise en correspondance en utilisant un HMM, nous implémentons l'algorithme sur Matlab et analysons des résultats.

Avant tout, nous simplifions encore plus la formulation faite dans (1.13) pour aboutir à la forme :

$$\sum_{m=1}^{M} s_1^m(t_1^m) = \sum_{m=1}^{M} s_2^m(t_2^m) \tag{1.14}$$

Nous avons substitué les variables $t_1^m = t$ et $t_2^m = t - \beta^m$. De cette façon, chercher le s_2^m qui correspond à un certain s_1^m revient à chercher l'indice spatiale t_2^m de la ligne S_2 qui correspond à l'indice t_1^m de la ligne S_1. Les signaux S_1 et S_2 sont codés sur niveaux de gris, c.à.d que s_1^m et s_2^m peuvent prendre des valeurs dans $\{1,2, \dots k, \dots K\}$ où K est le nombre de niveaux de gris, et vaut typiquement 255. On omet aussi α^m par simplification.

Le processus caché que nous souhaitons dévoiler est l'assignation qui relie chaque t_2^m à un t_1^v. L'idée est d'adapter la chaîne de Markov au contenu de $s_1^m(t_1^m)$, $t_1^m = m$ et $m = \{1, \dots, M\}$, et durant l'opération (recherche du correspondant), pour chaque échantillon présenté de la deuxième image s_2^m, on doit savoir quel est le t_2^m correspondant. Ainsi, les états cachés sont les t_2^m et les observations sont les s_2^m.

Notre HMM est constitué essentiellement d'une matrice de transition $A = [a_{ij}]$ où a_{ij} représente la probabilité de transition d'un état t_2^i à un état t_2^j et d'une matrice d'observation B dont les éléments $b_m(k)$ représentent la probabilité d'observer la couleur k alors qu'on est dans un état caché t_2^m. Dans notre chaîne de Markov, nous faisons l'hypothèse qu'un état courant est dépendant seulement de l'état précédent, ainsi

$$P(t_2^m | t_2^1) = P(t_2^m | t_2^{m-1}) \tag{1.15}$$

Une autre hypothèse dit que l'observation d'une couleur est dépendante de l'état courant seulement, indépendamment des observations et états précédents:

$$P(s_1^m | s_1^{m-1}, t_2^1) = P(s_1^m | t_2^m) \tag{1.16}$$

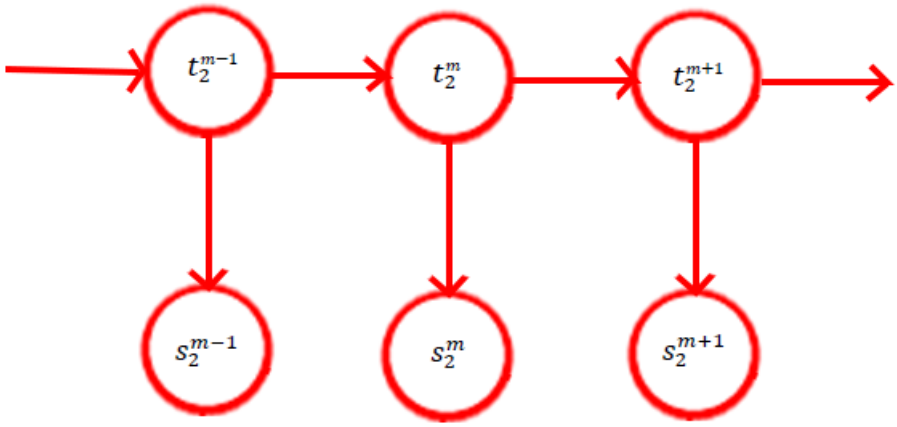


Figure 1.9 HMM utilisé dans le système stéréoscopique. Les états cachés sont t_2^m. Nous observons le contenu de la deuxième image s_2^m et on veut inférer les t_2^m en ayant en notre disposition s_1^m.

Si un objet se situe à l'infini, on a $t_2^m = t_1^m$, c.à.d. que $t_2^m - t_1^m = 0$. Lorsque l'objet s'approche, on aura $t_2^m - t_1^m > 0$ si S_2 est définie dans la vue de gauche. Le cas où $t_2^m - t_1^m < 0$ est impossible. Ainsi, les éléments de la matrice de transition sont déterminés comme suit :

$$a_{ij} = \begin{cases} 1/T & si\ i \leq j \\ 0\ autrement \end{cases} \tag{1.17}$$

Où T est un facteur de normalisation déterminé de sorte que $\sum_{i=1}^{N} \sum_{j=1}^{N} a_{ij} = 1$.

La matrice d'observation est déterminée comme suit :

$$b_m(k) = e^{-(\frac{s_1^m - k}{\gamma})^2} \tag{1.18}$$

Ici, γ est un paramètre arbitraire et sert à régler le degré de similitude entre une couleur s_1^m de la première image et une couleur de la deuxième image qui vient d'être observée k (k est définie sur S_2).

En définissant la matrice d'observation et la matrice de transition, notre HMM a été défini. Nous procédons par expliquer comment opérer dans ce système. Nous voulons savoir c'est quoi la

séquence d'états t_2^m qui a généré la séquence de couleur qu'on observe dans la deuxième image s_2^m. Cette étape est appelée le décodage et est effectuée à l'aide de l'algorithme de Viterbi (Viterbi 1967). On définit $\delta_m(v)$ par :

$$\delta_m(v) = \max_{t_2^1, t_2^2, \ldots, t_2^{m-1}} P(t_2^1 t_2^2 t_2^3 \ldots t_2^m = t_1^v, s_2^1 s_2^2 s_2^3 \ldots s_2^m) \tag{1.19}$$

Ayant inféré la probabilité maximale sur les valeurs des états cachés précédents $t_2^1 \ldots t_2^{m-1}$, c'est la probabilité que l'état courant (l'indice courant défini sur la ligne S_2) corresponde à un certain indice t_1^v de la première ligne, en ayant en disposition aussi les observations $s_2^1 \ldots s_2^m$. Cette probabilité est estimée séquentiellement. On commence par l'initialiser:

$$\delta_1(v) = \pi_{t_2^1} b_v(s_2^1) \tag{1.20}$$

Ici, $\pi_{t_2^1}$ est la probabilité de l'état initiale. Elle est choisie uniforme sur tous les états cachés possibles.

Par la suite, les $\delta_m(v)$ sont estimés récursivement:

$$\delta_m(v) = \max_{1 \le i \le N} [\delta_{m-1}(i) \mathrm{a}_{iv}] b_v(s_2^m) \tag{1.21}$$

Ainsi, la détermination d'un état caché se fait en fonction de la distribution $\delta_m(v)$:

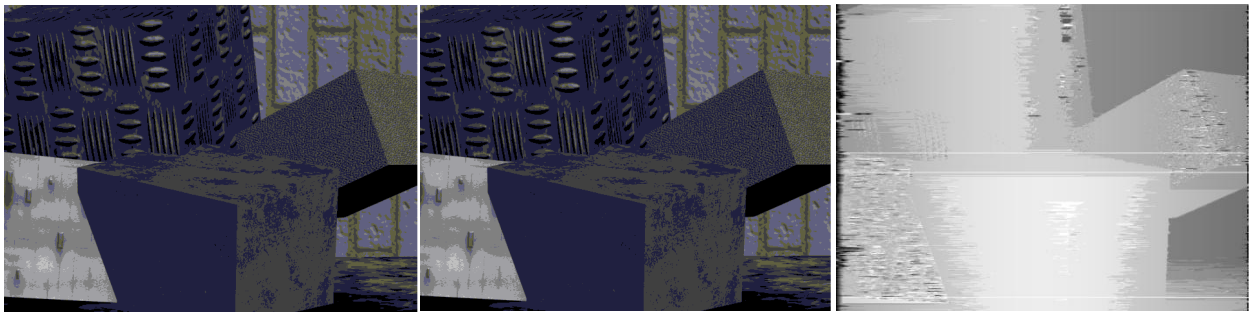
$$t_2^m = \arg \max_{1 \le i \le N} [\delta_m(i)] \tag{1.22}$$

La détermination de t_2^m conduit à la détermination de la disparité β^m puisque $t_2^m = t - \beta^m$.

Cette procédure est appliquée ligne par ligne. Nous avons construit une scène en 3D sous 3Ds max Studio, et avons fait la capture de la scène à partir de deux caméras alignées horizontalement de sorte à pouvoir utiliser les lignes épipolaires sans avoir à faire du calibrage. Les résultats sont visualisés dans Figure 1.10.

Nous pouvons observer des lacunes dans l'utilisation d'un tel système. Dans Figure 1.10 (c), l'image de profondeur est erronée dans la partie inférieure du dernier bloc à droite. Cette erreur qui s'étale sur toute la partie inférieure du bloc en question est causée par le fait que la propagation de messages a été faite de gauche à droite. Ainsi, une erreur sur la disparité à un certain endroit donnée, affecte les valeurs de disparité sur les pixels qui sont à droite du pixel erroné. L'étendu de l'erreur est dépendante du contenu de la scène, et des valeurs de disparités idéales et disparités erronées. Elle survient surtout lorsqu'il y a des effets d'occlusion. Vu que la

propagation des messages a été faite de gauche à droite, une autre propagation de messages de droite à gauche peut éliminer certaines erreurs de ce type. Dans la partie suivante, nous allons voire comment résoudre ce problème avec des méthodes graphiques plus sophistiquées, mais qui sont cette fois-ci tirées de la littérature.



(a) (b) (c)

Figure 1.10 Visualisation de deux images stéréoscopiques en niveaux de gris simulés sur 3ds Max Studio dans (a) (vue de gauche) et (b) (vue de droite). Dans (c), l'image de profondeur obtenue déduite à partir du système à base de HMM que nous avons développé. Les régions claires réfèrent à de petites distances par rapport au système de cameras simulé.

1.1.3 Propagation de conviction

Dans la partie précédente, nous avons présenté un modèle graphique que nous avons utilisé en faisant propager les messages selon une seule direction, de gauche à droite. Nous avons remarqué que cette approche entraîne la propagation de l'erreur dans cette même direction. La propagation de conviction, ou BP (Belief Propagation) vient au secoure pour pallier ce déficit. La propagation de conviction (Weinman, Tran et al. 2008) opère sur un graphe Markovien non directionnel tel qu'illustré dans Figure 1.11. Pour construire une image de disparité de dimensions $M \times N$, nous avons besoin de $M \times N$ nœuds qui sont organisés de sorte que chaque nœud soit relié à ces 4 plus proches voisins. Ainsi, un graphe peut être reconstruit en faisant des réplications de la structure visualisée dans Figure 1.11. Chaque nœud du graphe correspond à un pixel dans l'image de référence. Au niveau de chaque nœud, il existe un vecteur d'état qui contient les valeurs de disparités possibles au niveau du pixel correspondant avec une distribution de probabilité qui indique quel état (valeur de disparité) est la plus probable. Avec la propagation de conviction, les messages sont passés de chaque nœud à ses 4 voisins les plus proches de façon itérative. Ainsi, la

propagation de messages n'est pas faite dans une seule direction donnée, mais dans toutes les directions à chaque itération, au niveau de chaque nœud. On utilise la formulation élaborée dans (1.13) et on se propose d'inférer les montants de translation ou disparités β^m.

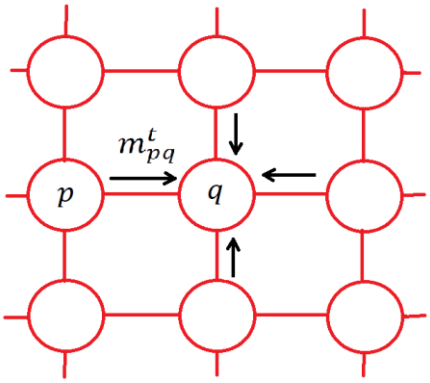


Figure 1.11 Illustration d'un graphe Markovien non directionnel et de la propagation d'un message m_{pq}^t d'un nœud p à un nœud q.

Pour décrire comment fonctionne la propagation de conviction avec un graphe non directionnel dans la résolution du problème de mise en correspondance, on définit les notations suivantes:

Soit P l'ensemble des pixels et p un pixel donné de cet ensemble tel que $p \in P$.

Soit L l'ensemble des valeurs d'état et f l'assignation qui lie un état (disparité) $f_p \in L$ à chaque pixel p.

On émet l'hypothèse que les valeurs d'état, soient les disparités, varient de façon lisse. Les changements brusques sont permis aux frontières des objets de la scène. La qualité de l'assignation des disparités aux pixels d'une image stéréoscopique peut être décrite par une fonction d'énergie qu'on souhaite minimiser:

$$E(f) = \sum_{(p,q)\epsilon N} V(f_p, f_q) + \sum_{p\epsilon P} D_p(f_p) \qquad (1.23)$$

Ici, N représente les arrêtes dans un graphe dont les nœuds sont reliés à quatre arrêtes comme dans Figure 1.11. $V(f_p, f_q)$ est le coût de discontinuité, il représente le coût d'assignation de f_p et f_q à deux pixels voisins. $D_p(f_p)$ est le coût dépendant directement des donnés (intensité de couleur), c'est le coût d'assignation de la disparité f_p au pixel p.

Un exemple simple qui montre une forme que peut prendre le coût de discontinuité est:

$$V(f_p, f_q) = |f_p - f_q| \tag{1.24}$$

La fonction de coût $D_p(f_p)$ peut prendre la forme :

$$D_p(f_p) = |s_1\ (p) - s_2\ (p - f_p)| \tag{1.25}$$

Où $s_1\ (p)$ et $s_2\ (p)$ réfèrent au contenu de deux lignes épipolaires de deux images stéréoscopiques.

La propagation de conviction fonctionne en faisant passer des messages entre les nœuds du graphe markovien. Soit m_{pq}^t le message que le nœud p envoie à un nœud voisin q au temps t. On initialise tous les messages à 0, on aura ainsi $m_{pq}^0 = 0$. À chaque itération, les messages passés évoluent selon:

$$m_{pq}^t(f_q) = \min_{f_p}(V(f_p, f_q) + D_p(f_p) + \sum_{s\epsilon N(p)\backslash q} m_{sp}^{t-1}(f_p)) \tag{1.26}$$

Ici, $s\epsilon N(p)\backslash q$ veut dire pour tout s appartenant au voisinage de p mais différent de q. $\sum_{s\epsilon N(p)\backslash q} m_{sp}^{t-1}(f_p))$ est la somme de tous les messages reçus au nœud p autre que ceux partant du nœud q à l'itération précédente.

Après T itérations, on calcule un vecteur de conviction au niveau de chaque nœud de la façon suivante :

$$b_q(f_q) = D_q(f_q) + \sum_{p\epsilon N(q)} m_{pq}^T \tag{1.27}$$

On remarque que b_q est une fonction de f_q. On calcule alors la valeur de f_q qui minimise $b_q$et on la note f_q^*.

Le temps d'exécution de cette procédure est de l'ordre de $O(nk^2T)$. Toutefois, c'est possible d'avoir des performances plus importantes en faisant des simplifications.

Pour optimiser la propagation de conviction, écrivons l'équation (1.26) sous la forme:

$$m_{pq}^t(f_q) = \min_{f_p}(V(f_p, f_q) + h(f_p)) \tag{1.28}$$

Avec :

$$h(f_p) = D_p(f_p) + \sum_{s \epsilon N(p) \setminus q} m_{sp}^{t-1}(f_p)) \tag{1.29}$$

Notre but ici est de rendre le temps d'extraction de la disparité d'un pixel donné dans un ordre $O(k)$ plutôt que $O(k^2)$ en faisant attention au fait que la partie $V(f_p, f_q)$ dépend de la différence entre f_p et f_q et donc l'information nécessaire est contenue dans un seul nombre qui est la différence de ces deux disparités et non pas dans une paire de nombres.

Pour commencer, on prend le modèle de Potts pour définir la fonction $V(f_p, f_q)$. Ainsi,

$$V(f_p, f_q) = \begin{cases} 0 & si\ f_p = f_q \\ d & autrement \end{cases} \tag{1.30}$$

Dans ce cas, l'équation (3.5) peut être écrite sous la forme:

$$m_{pq}^t(f_q) = min\left\{h(f_q), \min_{f_p} h(f_p) + d\right\} \tag{1.31}$$

Sous cette forme, la minimisation en fonction de f_p se fait indépendamment de f_q. Ainsi, on minimise $h(f_p)$ selon f_p une fois avant de choisir le minimum entre $h(f_q)$ et $\min_{f_p} h(f_p) + d$. Cette opération prend ainsi un temps de l'ordre de $O(nkT)$. Il existe d'autres manières pour optimiser la propagation de conviction, mais cet exemple montre que la simplification d'un ordre de $O(nk^2T)$ à un ordre de $O(nkT)$ a été faite au coût d'une diminution de la résolution de la fonction d'énergie $V(f_p, f_q)$. Intuitivement, on s'attend à une diminution de précision et augmentation du taux d'erreur.

La détermination de l'image de profondeur par la propagation de conviction nécessite d'opérer sur les images stéréoscopiques au complet, même si seuls quelques pixels sont requis par l'application. En pratique, la convergence vers une solution relativement stable nécessite 100 itérations (Felzenszwalb and Huttenlocher 2006).

1.1.4 Discussion

Dans cette partie nous avons présenté deux méthodes graphiques. La première est une méthode simple basée sur un HMM. Elle est régie par une seule itération. Dans cette itération, nous faisons une propagation de messages de gauche à droite et nous avons remarqué que l'erreur se propage selon le même sens de la propagation de messages. Une solution peut être de faire une

double propagation, la première de gauche à droite et la deuxième de droite à gauche, et retenir la solution la plus plausible. Toutefois, ca ne permet pas d'éliminer totalement le problème de propagation de l'erreur sur plusieurs pixels. Par exemple, si sur une ligne épipolaire, l'inférence de disparités est erronée en deux pixels, les pixels qui se situent entre les deux pixels erronés sont susceptibles à l'erreur malgré la double propagation. D'autre part, le système que nous avons construit est simple, est constitué d'un simple HMM avec une seule itération, ce qui faciliterait la tâche de l'implémenter en temps réel.

La deuxième méthode, BP, est tirée de la littérature et est à l'origine de plusieurs des meilleures méthodes connues à la base de données *Middleburry* aujourd'hui (Scharstein and Szeliski 2002). La différence entre BP et HMM est que la propagation de messages avec BP se fait dans tous les sens au niveau de chaque nœud. En plus, ca requiert de faire vers des dizaines d'itérations pour aboutir à la convergence et minimiser la fonction d'erreur. L'inférence d'un vecteur d'état au niveau d'un certain nœud est dépendant des autres vecteurs d'états aux autres nœuds du graphe. Lorsque le nombre d'itérations augmente, le vecteur d'état à un certain nœud sera affecté par les vecteurs d'état des nœuds qui sont de plus en plus lointains. Cette dépendance entre les vecteurs d'états est à l'origine de la nomination de cette famille de méthodes, «méthodes globales». Cette dépendance est problématique lorsque l'application est une application de *stéréovision dispersée*. En *stéréovision dispersée*, on a besoin de calculer la disparité au niveau de seulement quelques pixels, et l'emploi d'une méthode globale, tel que BP, exige qu'on infère des vecteurs d'états partout au voisinage du nœud au niveau duquel on veut inférer la disparité, ce qui veut dire que des vecteurs d'états auront à être inférés au niveau des pixels à lesquels l'application ne requiert pas de calculer la disparité. Ceci devient problématique avec BP vu que le nombre d'itérations peut atteindre une centaine. Le temps d'exécution de BP dans une application de stéréovision dispersée peut être le même que dans une application de stéréovision dense dans laquelle on veut inférer la disparité au niveau de tous les nœuds. Dans la partie suivante, nous allons voir des méthodes qui permettent de réduire le temps d'exécution d'un algorithme en fonction du nombre d'estimés de disparités requis par l'application.

1.2 Méthodes locales

Les méthodes locales sont basées surtout sur la corrélation. La corrélation est un outil classique qui permet de mesurer la similitude entre deux signaux et même de calculer le montant de

translation entre ces deux signaux. Dans cette partie, nous présentons la corrélation et la corrélation de phase. Nous présentons des outils qui sont utilisés pour améliorer la résolution et la précision de ces méthodes. Nous présentons aussi des méthodes de stéréovision locales à base de phase qui sont présentes dans la littérature.

1.2.1 Convolution, corrélation et corrélation de phase

La convolution $y(t)$ de deux signaux $u(t)$ et $v(t)$, définie dans le domaine continu, est donnée par :

$$y(t) = v(t) * u(t) = u(t) * v(t) = \int_{-\infty}^{\infty} u(\tau)v(t-\tau)d\tau \tag{1.32}$$

La convolution est associative, commutative et distributive. Dans le domaine discret, elle est donnée par:

$$y(n) = \sum_{m=-\infty}^{\infty} u(m)v(n-m) \tag{1.33}$$

Quant à la corrélation $z_{u,v}(t)$, elle peut être exprimée comme étant une convolution :

$$z_{u,v}(t) = u(t) * conj(v(-t)) = \int_{-\infty}^{\infty} u(\tau)conj(v(t-\tau))d\tau \tag{1.34}$$

On note que la corrélation n'est pas commutative. Elle peut être appliquée sur des signaux réels discrets par:

$$z_{u,v}(n) = \sum_{m=-\infty}^{\infty} u(m+n)v(m) \tag{1.35}$$

En traitement d'images, c'est utile de généraliser la convolution et la corrélation sur le domaine bidimensionnel :

$$y(n_1, n_2) = \sum_{m_1=-\infty}^{\infty} \sum_{m_2=-\infty}^{\infty} u(m_1, m_2)v(n_1 - m_1, n_2 - m_2) \tag{1.36}$$

$$z_{u,v}(n_1, n_2) = \sum_{m_1=-\infty}^{\infty} \sum_{m_2=-\infty}^{\infty} u(n_1 + m_1, n_2 + m_2)v(m_1, m_2) \tag{1.37}$$

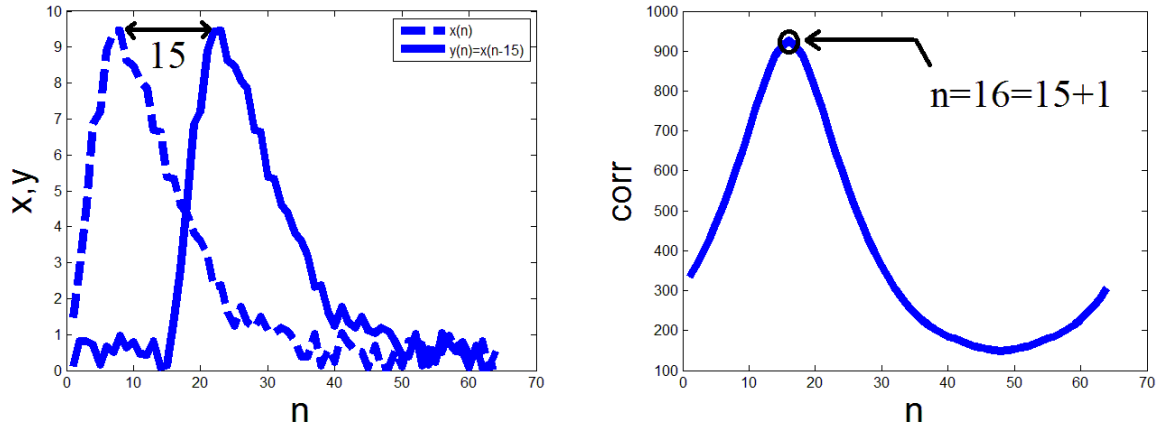
La transformée de Fourier de la convolution de $v(t)$ avec $u(t)$ est le produit de leurs transformées de Fourier $V(f)$ et $U(f)$

$$FFT\{v(t) * u(t)\} = V(f)U(f) \tag{1.38}$$

Pour les signaux réels, la transformée de Fourier d'une corrélation s'exprime selon :

$$z_{u,v}(t) \leftrightarrow U(f)conj(V(f)) \tag{1.39}$$

La corrélation peut être utilisée pour calculer le montant de translation entre deux signaux (Figure 1.12). Pour ceci, il suffit de chercher le maximum sur l'espace de recherche défini par la corrélation. L'indice qui correspond au maximum correspond au montant de translation additionné de la valeur 1. En fait, le premier indice dans l'espace de recherche correspond à un montant de translation nul.



(a) (b)

Figure 1.12 Signal x et son instance y translatée de 15 unités dans (a) et la corrélation $z_{y,x}$ dans (b) qui montre un pic à la valeur de n qui correspond au montant de translation additionné de 1.

La corrélation de phase unidimensionnelle, notée $c_{u,v}(t)$, est obtenue en normalisant le spectre de la cross-corrélation unidimensionnelle entre deux signaux. Elle est exprimée comme suit :

$$c_{u,v}(t) = IFFT\{U(f)conj(V(f))/|U(f)conj(V(f))|\} \tag{1.40}$$

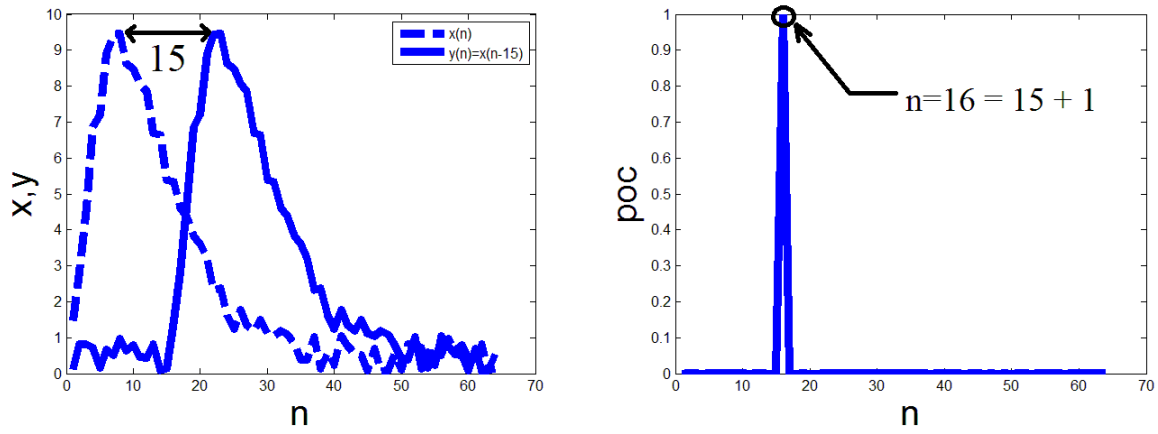
Comme montré dans Figure 1.13, la corrélation de phase conduit à un espace de recherche plus concentré sur la valeur de translation recherchée, alors que l'espace de recherche de la corrélation classique montré dans Figure 1.12 est plus dispersée autour de la valeur de translation.

Si $y(t)$ est obtenu de $x(t)$ par une simple translation tel que $y(t) = x(t-a)$, on aura que $Y(f) = X(f)e^{j2\pi af}$. La corrélation de phase entre x et y est $c_{y,x}(t) = IFFT\{e^{j2\pi af}\} = \delta(t-a)$ en vertu de l'équation (1.40), $\delta(t)$ est l'impulsion de Dirac. Ainsi, une maximisation sur l'espace de recherche $c_{y,x}(t)$ conduit à l'inférence de la valeur de translation a entre $x(t)$ et $y(t)$. Quelque soit la forme de $y(t)$, assumant que son spectre n'a pas une composante nulle, si ce signal est obtenu par simple translation de $x(t)$, la forme de $c_{y,x}(t)$ sera une impulsion de Dirac translatée de l'origine du même montant de translation qui existe entre $x(t)$ et $y(t)$.

Nous pouvons définir de façon similaire la corrélation de phase bidimensionnelle généralisée au traitement d'images :

$$c_{u,v}(n_1, n_2) = IFFT\{U(\Omega_1, \Omega_2)conj(V(\Omega_1, \Omega_2))/|U(\Omega_1, \Omega_2)conj(V(\Omega_1, \Omega_2))|\} \quad (1.41)$$

Où (n_1, n_2) sont les indices spatiaux qui sont les indices d'une image dans le domaine discret et (Ω_1, Ω_2) sont les indices discrets dans le domaine fréquentiel. La corrélation de phase est reprise dans le chapitre 2.



(a) (b)

Figure 1.13 Signal x et son instance y translatée de 15 unités dans (a) et la corrélation de phase $c_{y,x}$ dans (b).

En pratique, avant de faire la transformée de Fourier d'un signal donné, on le multiplie par une fenêtre prédéfinie. Ce prétraitement permet d'éliminer les effets de bords lors du calcul du spectre d'un signal. Les effets de bords sont présents lorsque la portion du signal retenue pour effectuer la transformée de Fourier comprend de hautes amplitudes au début et à la fin de cette portion.

Soit un signal $x(t)$ défini dans le domaine du temps (ou de l'espace spatial) continu. Afin de faire l'analyse numérique du signal, la lecture est faite à des instants discrets. Dans la phase acquisition, on implémente alors un interrupteur qui fait une lecture de la valeur instantanée du signal à chaque instant nT (secondes) où $n = 0,1,2,$ ... est un entier et T est une constante appelée la période d'échantillonnage. Un nombre d'échantillons fixe est pris en compte à chaque fois qu'on veut faire une manipulation numérique du signal. Théoriquement, ceci revient à multiplier le signal discrétisé $x(nT)$ par une fenêtre qui est d'amplitude 1 lorsqu'on est sur une région du signal qu'on veut retenir pour manipulation et 0 si on est sur une région du signal qui sort du domaine d'étude. Ceci revient à multiplier le signal par une courbe rectangulaire qui définit le domaine d'étude. Or, une multiplication dans le domaine temporel (ou spatial) correspond à une convolution dans le domaine fréquentiel :

$$F\{x(t)y(t)\} = \frac{1}{2\pi}X(f) * Y(f) \tag{1.42}$$

Où $X(f) = FFT\big(x(t)\big)$ et $Y(f) = FFT\big(y(t)\big)$.

Ainsi, la multiplication du signal continu par un signal rectangulaire équivaut à faire une convolution du spectre du signal d'étude idéal par une fonction qui est le spectre du signal rectangulaire. Soit $v(t)$ un signal rectangulaire (Figure 1.14). On a :

$$v(t) = \Pi_{T/2}(\mathrm{t}) = u\left(t + \frac{T}{2}\right) - u(t - \frac{T}{2}) \tag{1.43}$$

Où $u(t)$ est la fonction échelon définie par :

$$u(t) = \begin{cases} 0, & t < 0 \\ 1, & t \geq 0 \end{cases} \tag{1.44}$$

On peut prouver que :

$$V(j\omega) = F\{v(t)\} = TSa(\pi fT) \tag{1.45}$$

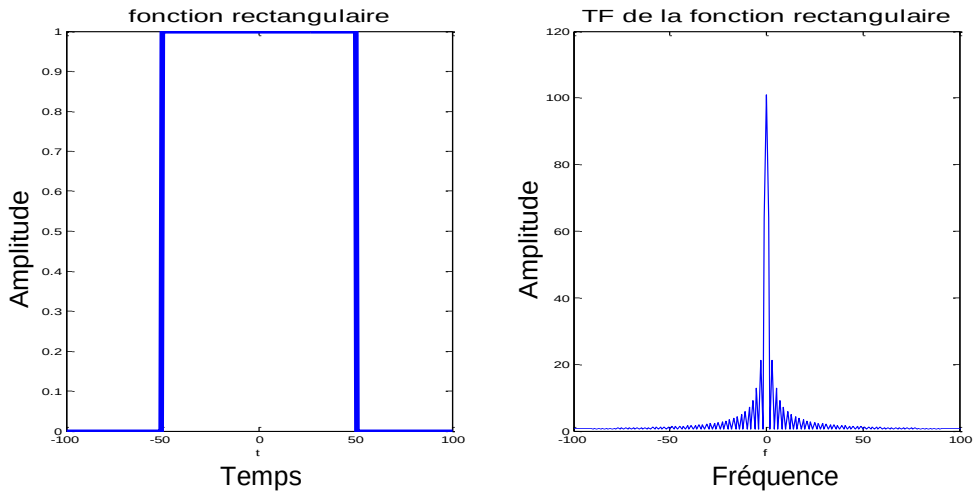


Figure 1.14-Fonction rectangulaire (gauche) et sa transformée de Fourrier (droite)

Où $Sa(t)$ désigne la fonction sinus cardinal.

On sait que la fonction sinus cardinal comprend des lobes qu'on veut éliminer (Figure 1.14). Pour ceci, on cherche une fonction dont le spectre ne comprend pas de lobes et on multiplie le signal d'étude par cette fonction avant d'effectuer la transformée de fourrier. En pratique, c'est difficile de trouver une fonction dont le spectre ne comprend pas de lobes, on cherche alors une fonction dont le spectre comprend des lobes à faibles amplitudes. Plusieurs fenêtres peuvent être utilisées dont la fenêtre de Hanning et la fenêtre de Hamming.

La fenêtre de Hanning est exprimée par :

$$v_{hann}(t) = v_1(t)\Pi_{T/2}(t) \tag{1.46}$$

Où $\Pi_{T/2}(t)$ est la fonction rectangulaire et $v_1(t)$ est définie par :

$$v_1(t) = \frac{A}{2}\left\{1 + \cos\left(\frac{2\pi t}{T}\right)\right\} \tag{1.47}$$

Son spectre est donné par :

$$V_{hann}(f) = \frac{A}{2}\frac{\sin(T\pi f)}{\pi f(1 - T^2 f^2)} \tag{1.48}$$

La fenêtre de Hanning ainsi que son spectre sont données dans Figure 1.15. La fenêtre de Hamming est exprimée par:

$$v_{hamm}(t) = 0.54 - 0.46\cos\left(\frac{2\pi t}{T}\right) \quad (1.49)$$

Son spectre est donné par :

$$V_{hamm}(f) = \frac{\sin\ \pi f T(0.54 - 0.08f^2T^2)}{\pi f(1 - T^2f^2)} \quad (1.50)$$

Le spectre de la fenêtre de Hamming (Figure 1.16) comprend des lobes dont l'amplitude vaut un cinquième de celles des lobes de la fonction Hanning.

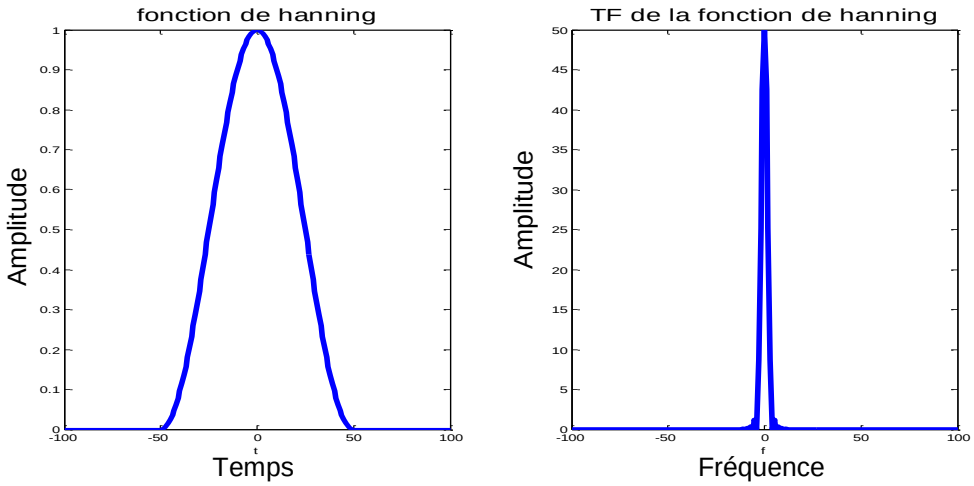


Figure 1.15-Fonction de Hanning (gauche) et sa transformée de Fourrier (droite)

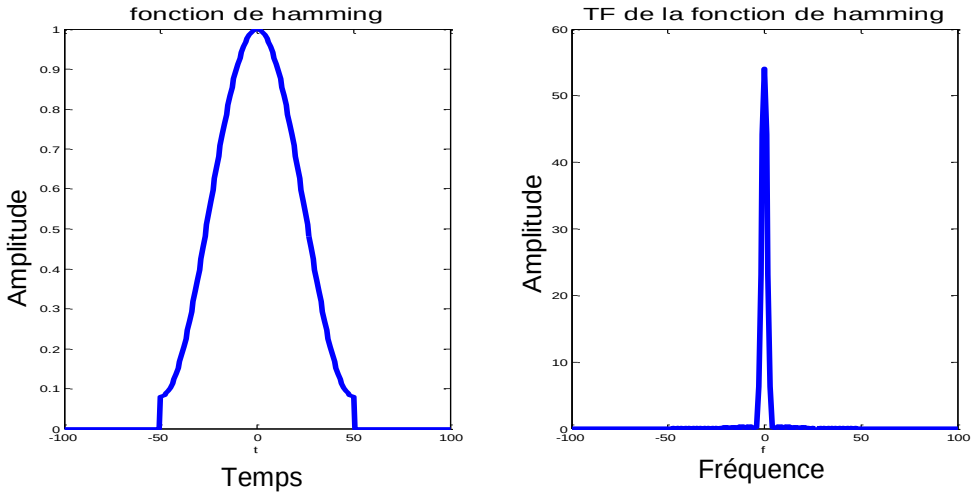


Figure 1.16-Fonction de Hamming (gauche) et sa transformée de Fourrier(droite)

La corrélation nécessite d'effectuer la transformée de Fourier et la transformée de Fourier inverse en plus des multiplications et divisions. Or les ressources matérielles et temporelles requises par

le calcul de la transformée de Fourrier augmentent avec le nombre d'échantillons considérés du signal d'étude. Ceci peut être problématique lorsqu'on veut appliquer la corrélation sur de grandes images. Afin de résoudre ce problème, on procède en créant des copies des images d'étude avec moins de résolution, réduisant ainsi la taille de l'image et par la suite du domaine d'étude. Si l'image est trop grande, on peut avoir besoin de faire cette réduction de résolution plusieurs fois, par étapes en créant ainsi ce qu'on appelle une pyramide.

Il y a plusieurs manières de créer une pyramide, toutes dépendent de la nature de l'application, du temps d'exécution de l'algorithme souhaité, des dimensions des objets d'intérêt dans une scène et de la tolérance à l'erreur voulue.

Une pyramide est tout d'abord constituée de plusieurs images qui sont des répliques d'une image originale sous échantillonnée à plusieurs niveaux. Si on note l'image originale par image de niveau 1, les images sous échantillonnées qui en découlent peuvent être notées image niveau 2, image niveau 3, etc… où l'image de niveau N est obtenue de l'image de niveau N-1 par une procédure de sous échantillonnage.

Le sous échantillonnage peut se faire de plusieurs manières. On peut par exemple créer chaque pixel de l'image de niveau N d'une pyramide en faisant la convolution de l'image de niveau N-1 par une fenêtre et en faisant ensuite la compression de cette image (en pratique, on ne fait la convolution qu'aux points qu'on veut retenir dans l'image d'hiérarchie supérieure). Normalement, cette compression se fait en retenant un pixel sur 4, soit le pixel du coté supérieur gauche d'un carré composé de quatre pixels adjacents. L'image de niveau N aura une longueur et une largeur égales aux moitiés de celles de l'image de niveau N-1 (Figure 1.17).

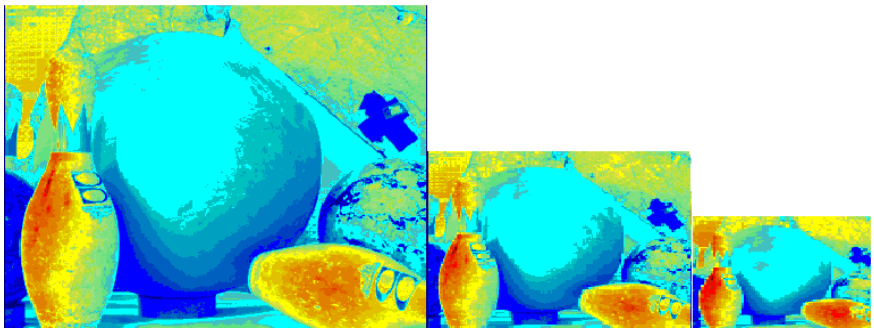


Figure 1.17-Pyramide obtenue avec une fenêtre de compression de dimensions 2×2

La fenêtre avec laquelle on fait la convolution peut s'agir entre autres d'une gaussienne, d'un cosinus surélevé, une fonction de Hamming ou de Hanning ou bien même la fonction uniforme dont tous les éléments sont égaux à une constante ϕ tel que :

$$\phi = \frac{1}{N} \tag{1.51}$$

Où N désigne le nombre de pixels qui appartiennent à la fenêtre de convolution.

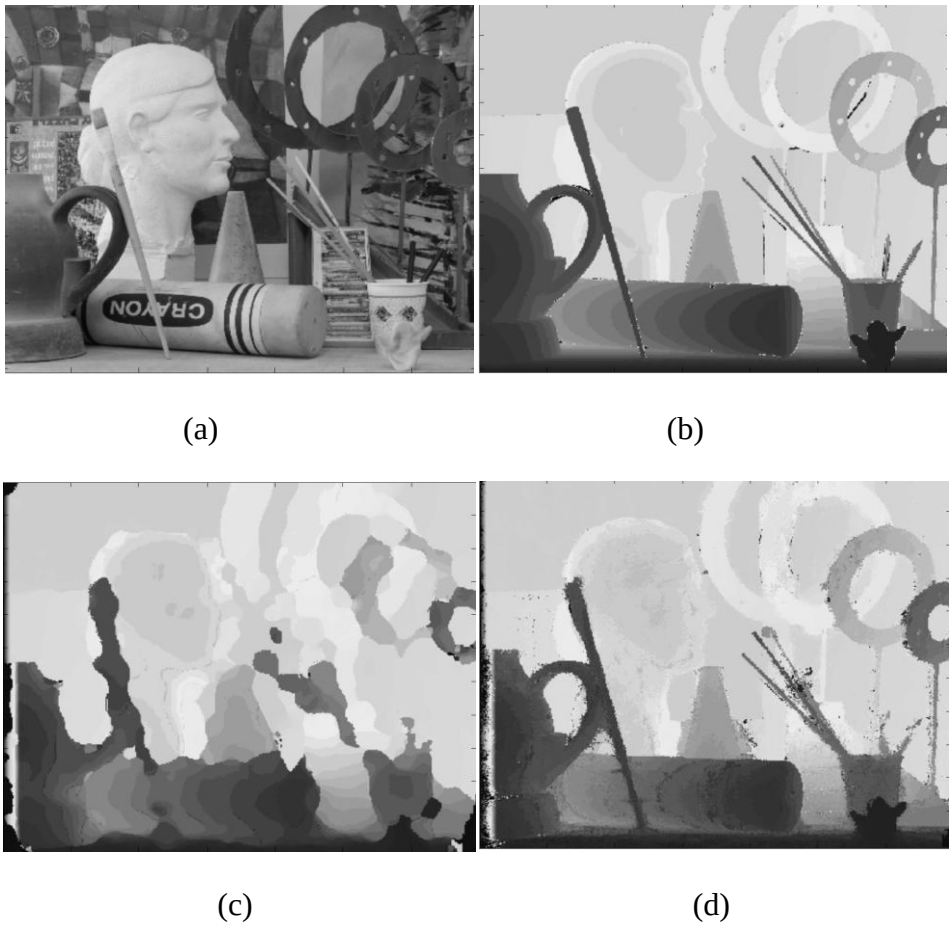
Une relation que doit respecter la forme de la fenêtre est:

$$\int_{\frac{-N_1}{2}}^{\frac{N_1}{2}} \int_{\frac{-N_2}{2}}^{\frac{N_2}{2}} f(n_1, n_2)\, dn_1 dn_2 = 1 \tag{1.52}$$

Où $f(n_1, n_2)$ dénote la valeur que prend la fenêtre de dimensions $N_1 \times N_2$ aux coordonnées (n_1, n_2).

Des précautions doivent être prises lorsque la taille de la fenêtre de convolution et celle utilisée pour la compression ne sont pas de mêmes dimensions ou bien doivent ne pas être de même dimensions. Par exemple, si on utilise une gaussienne pour construire une image d'hiérarchie supérieure, la taille de la fenêtre gaussienne doit être supérieure à la fenêtre de convolution sinon, on risque de négliger l'information que fournit les pixels à rejeter durant la compression. Lorsque la taille de la fenêtre gaussienne augmente, la variance de la gaussienne augmente, et on aura un effet de filtrage passe bas. Il faudra alors choisir une fenêtre gaussienne dont la taille n'est pas trop grande ni trop petite.

Nous avons simulé un système de stéréovision passive à base de la corrélation de phase bidimensionnelle. Une fenêtre de Hanning a été utilisée pour compenser les effets de bords. Des pyramides d'images de profondeur 6 ont été utilisées pour garantir la couverture de la plage de disparités nécessaire. Les résultats avec et sans segmentation sont illustrés dans Figure 1.18.



(a) (b)

(c) (d)

Figure 1.18 Illustration d'une image tirée d'une paire d'image stéréoscopique (a) figurant dans la base de données de *Middlebury* (Scharstein and Szeliski 2001) et le résultat considéré idéale tiré à partir de la méthode de lumière structurée (b) (Scharstein and Pal 2007). Dans (c) est visualisée une carte de disparités obtenue avec la corrélation de phase bidimensionnelle sans segmentation et dans (d) le résultat obtenu avec segmentation.

1.2.2 Version ALBA de la corrélation de phase

La version ALBA (Alba and Arce-Santana 2009) de la corrélation de phase unidimensionnelle considère une formulation du problème de la stéréovision selon laquelle les objets vus dans une image sont obtenus en faisant des simples translations des objets de l'autre image.

$$\sum_{m=1}^{M} R_1^{m,a_m,b_m}(t) = \sum_{m=1}^{M} \alpha^m R_2^{m,a_m,b_m}(t-\beta^m) \tag{1.53}$$

Les disparités sont estimées un pixel à la fois. Pour chaque pixel, on définit une région fixe prédéterminée qui est centrée sur ce pixel. Nous rappelons que dans la définition des régions R_1^{m,a_m,b_m} et R_2^{m,a_m,b_m} , nous avons posé comme contrainte que ces régions doivent être choisis de sorte qu'une région peut être obtenue de l'autre par une translation et un étalement spatiale (dans ce cas, on considère seulement la translation). La version Alba ne respecte pas ces contraintes, et les régions que Alba prend sont extraites telles qu'elles sont de l'image originale en prenant tout simplement $N/2$ pixels de chacune des cotés gauche et droite du pixel en question. Les images ne sont donc pas segmentées de façon à les décomposer en des régions cohérentes. La raison c'est que la segmentation nuit à la précision de la corrélation de phase unidimensionnelle (Hawi and Sawan 2011). Ainsi, lorsqu'on fait la corrélation de phase, le bruit qui résulte du fait que le domaine d'étude comprend des objets de disparités différentes, l'effet des occlusions, le bruit d'échantillonnage et autres paramètres qui décrivent la transformation d'une image à l'autre sont considérés comme bruit qui nuit à la détection de la bonne valeur de disparité par la corrélation de phase. Ce bruit dans l'espace de recherche de la corrélation de phase fait en sorte que la maximisation de cet espace de recherche conduirait à la détection d'un pic qui ne correspond pas forcément à la vraie disparité. Pour résoudre ce problème, Alba fait la détection de plusieurs pics dans le domaine de recherche. Le nombre de pics à détecter est typiquement entre $N/4$ et $N/3$ où N est la dimension de l'espace de recherche (nombre d'éléments). Pour chacun des pics détectés, on fait une étude de consistance. Ainsi, la disparité retenue est celle qui minimise l'erreur:

$$S(x,y,d) = S(x-1,y,d) - C(x-1-w,y,d) + C(x+w,y,d) \qquad (1.54)$$

Avec :

$$C(x,y,d) = \sum_{j=-w}^{w} |f(x+d,y+j) - g(x,y+j)| \qquad (1.55)$$

La fonction $C(x,y,d)$est connue sous le nom de SAD (Sum of Absolute Differences). Dans ces équations, (x,y) sont les coordonnées d'un point dans une image, d une valeur de disparité qu'on veut évaluer et w reflète la taille du domaine d'étude sur lequel ont veut effectuer la mesure SAD.

Enfin, un filtrage passe bas est effectué en faisant la convolution de l'image de profondeur obtenue avec un filtre passe bas. Figure 1.19 illustre la version ALBA.

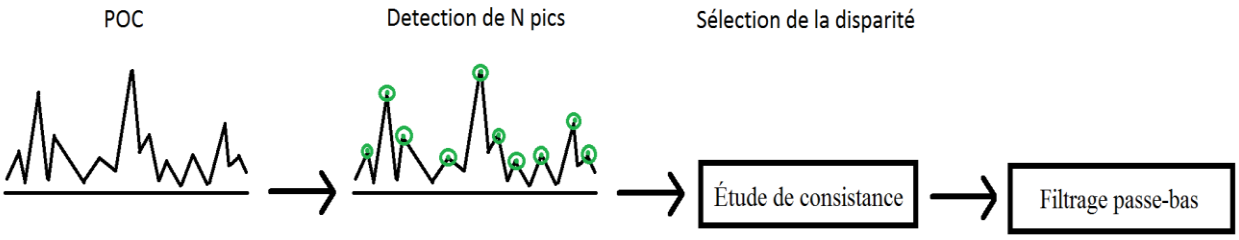


Figure 1.19 Illustration des étapes entreprises dans la version ALBA de la corrélation de phase

1.2.3 Système de stéréovision à base de scalogramme

Le système d'El-Etriby (El-Etriby, Al-Hamadi et al. 2007) est basé sur la représentation des signaux d'entrée sous forme de scalogramme. Un scalogramme est une représentation qui permet d'extraire d'un signal une information fréquentielle mais aussi spatiale en même temps. Il s'agit de faire la convolution du signal d'intérêt avec une banque de filtres qui contient des ondelettes ayant des fréquences différentes. En particulier, El-Etriby utilise des ondelettes de Gabor. Une ondelette de Gabor peut être décrite par:

$$G(x,\lambda) = \rho(x,\lambda)e^{-i\phi(x,\lambda)} \ , x = [-m\lambda/2, m\lambda/2] \tag{1.56}$$

L'amplitude $\rho(x,\lambda)$ est définie comme :

$$\rho(x,\lambda) = e^{-\left(\frac{x}{m\lambda\sigma}\right)^2} \tag{1.57}$$

La phase $\phi(x,\lambda)$ est exprimée selon :

$$\phi(x,\lambda) = 2\pi x/\lambda \tag{1.58}$$

Dans ces équations, λ est la longueur d'onde de l'ondelette, m est le nombre d'ondelettes qui peuvent être contenus dans une fenêtre et σ est un paramètre qui contrôle la variance de l'amplitude de l'ondelette. Si S_1 est un signal extrait d'une ligne épipolaire de la première image stéréoscopique et S_2 un signal extrait d'une ligne épipolaire de la deuxième image, le scalogramme $B_{S_i}(x,\lambda)$ d'un signal S_i, $i = [1,2]$, est obtenu en faisant la convolution de S_i avec les ondelettes de Gabor:

$$B_{S_i}(x,\lambda) = S_i * G(x,\lambda) \tag{1.59}$$

Il est intéressant de noter que le scalogramme prend un signal unidimensionnel et produit une représentation bidimensionnelle. La raison d'emploi de cette représentation est la prise en compte de l'étalement spatiale lors de la recherche de correspondance. Ainsi, la formulation du problème de la stéréovision passive qu'on vise de résoudre est celui décrit par l'équation (1.8). L'algorithme d'El-Etriby est adapté au cas où l'objet à reconstruire en 3D est étendu sur plusieurs niveaux de profondeur. Lorsqu'un objet est étendu sur plusieurs niveaux de profondeurs, son image dans une vue est étalée sur une région de largeur différente de celle de son image dans l'autre vue comme visualisé dans Figure 1.7. Le scalogramme permet de prendre en compte de cette différence. On procède par calculer $B_{S_1}(x,\lambda)$ et $B_{S_2}(x,\lambda)$, les scalogrammes des signaux S_1 et S_2. Chacune des composantes des scalogrammes comprend une amplitude $\rho_{S_i}(x,\lambda)$ et une phase $\phi_{S_i}(x,\lambda)$. La disparité peut être inférée à partir de l'information sur la phase contenue dans le scalogramme. La disparité d peut être obtenue d'après l'équation suivante:

$$d = \left|\phi_{S_1}(x_1,\lambda_1)\lambda_1 - \phi_{S_2}(x_2,\lambda_2)\lambda_2\right| \tag{1.60}$$

On note que x_1 et λ_1 sont respectivement l'indice d'une ondelette de Gabor et sa longueur d'onde dans le scalogramme de la première image qui est celle de la caméra de gauche dans cet exemple précis. Similairement x_2 et λ_2 sont définis sur le scalogramme de l'image de droite. Lorsque l'objet est oblique par rapport au système de caméras comme dans Figure 1.7, $\lambda_1 \neq \lambda_2$ et la recherche des valeurs de λ_1 et λ_2 optimaux repose sur une comparaison des amplitudes des deux scalogrammes. La valeur exacte de l'étalement spatiale peut être déduite en faisant la fraction de λ_1 par λ_2. À noter aussi que pour inférer la disparité au niveau d'un pixel donné, on emprunte une stratégie qui est pareille à celle utilisée par les pyramides d'images: On cherche une première valeur avec une précision médiocre, c'est-à-dire avec un λ_1 élevé, mais avec une grande résolution et on utilise cette approximation pour estimer avec plus de précision la disparité en utilisant un λ_1 plus petit, et ainsi de suite…

1.2.4 Discussion

Les résultats dans Figure 1.18 montrent que la segmentation permet d'avoir plus de cohérence aux frontières d'objets. Les objets sont bien délimités et c'est plus facile à faire leur distinction dans Figure 1.18 (d) (avec segmentation) que dans Figure 1.18 (c) (sans segmentation). Il faut faire attention que lorsqu'on travaille avec la corrélation de phase unidimensionnelle, la

segmentation a un effet nuisible sur le taux d'erreur sur les disparités (Hawi and Sawan 2011). La raison c'est qu'une segmentation brute qui prend en référence la couleur du pixel à lequel on veut calculer la disparité est plus susceptible au bruit d'échantillonnage (Birchfield and Tomasi 1998). Pour illustrer l'effet du bruit d'échantillonnage sur le problème de mise en correspondance, on considère que la couleur d'un objet de la scène varie selon:

$$b(t) = \frac{1}{(1 + e^{\frac{t}{10}})^2} \tag{1.61}$$

Le contenu de la ligne épipolaire de l'image de gauche est noté par $x(n)$ et celui de l'image de droite par $y(n)$. L'image de gauche va percevoir cette variation de couleur sur un nombre de pixels fini, sur des échantillons $n = t/T$, avec T la période d'échantillonnage. On considère que l'objet se situe à une distance uniforme par rapport au système de caméras de sorte que la transformation qui décrit le passage de l'image de l'objet dans la vue de gauche à celle de droite soit décrite par une simple translation. Ainsi, on pose dans cette expérience que $y(n) = x(n - \alpha)$, où α est un montant de translation qui n'est pas un entier. On pose que $x(n) = b(n)$ par simplification. L'erreur d'assignation $e(n)$ est définie par :

$$e(n) = |y(n) - x(n)| \tag{1.62}$$

Si α est un nombre entier, cette erreur est intuitivement nulle. Si par contre α est un nombre décimal, l'erreur d'assignation sera plus importante là où il y a un changement brusque de couleur comme illustré dans Figure 1.20.

Dans Figure 1.21, nous illustrons l'effet de la segmentation sur la corrélation de phase unidimensionnelle. Le défi avec la corrélation de phase unidimensionnelle est d'arriver à des images de profondeur cohérentes aux frontières d'objets et en même temps avoir de bons taux d'erreur.

La corrélation de phase bidimensionnelle prend en entrée des patrons qui contiennent beaucoup plus de pixels qui sont repartis sur une plus grande région. Ainsi, c'est plus probable d'avoir des pixels qui ont des couleurs similaires à celle de la valeur de référence et le bruit d'échantillonnage est moins marqué.

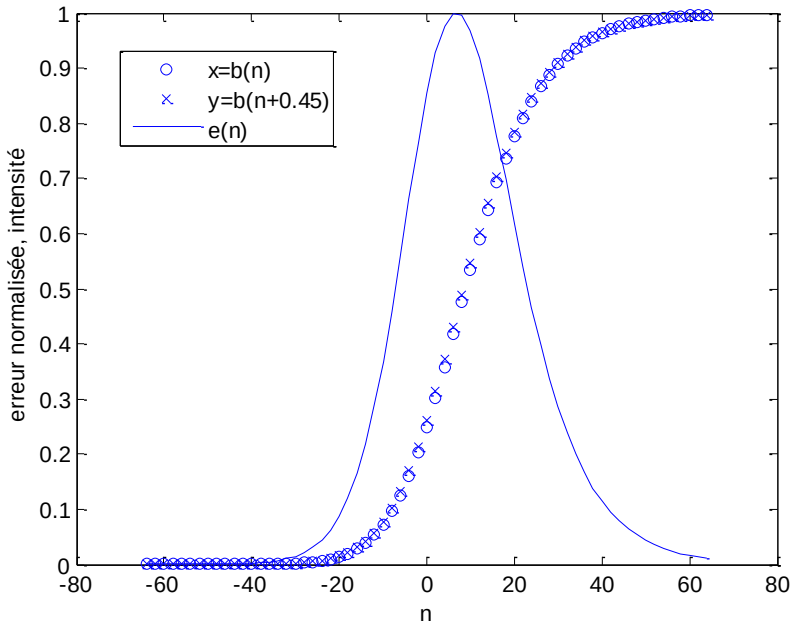
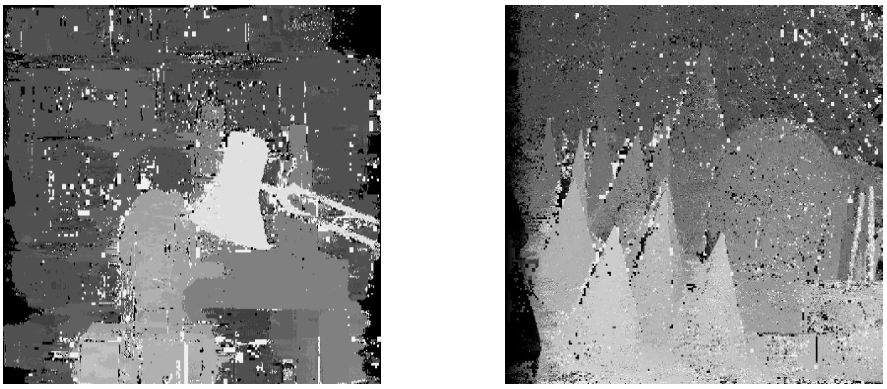


Figure 1.20 Illustration de l'erreur d'assignation de pixels de la vue de gauche $x(n)$ à ceux de la vue de droite $y(n)$. Il est remarquable que l'erreur augmente là où il y a un changement brusque de l'intensité de couleur. Le changement brusque de couleur se trouve souvent aux frontières d'objets.

Ceci dit, la corrélation de phase unidimensionnelle, bien que plus rapide que la corrélation de phase bidimensionnelle, est caractérisée par un taux d'erreur plus élevé que celui de la version bidimensionnelle et cette erreur augmente lorsqu'on tente de faire la segmentation pour avoir des images de profondeur plus cohérents. Les images de profondeur obtenues à partir du système de Alba (Alba and Arce-Santana 2009) sont floues comme ceux de Figure 1.18 (c) puisqu'on ne fait pas de segmentation.



(a) (b)

Figure 1.21 Images de disparités obtenues avec la corrélation de phase unidimensionnelle avec segmentation. Bien que les frontières d'objets soient plus cohérentes, le bruit sur les disparités est visuellement considérable. Des résultats plus détaillés se trouvent dans (Hawi and Sawan 2011).

1.3 Conclusion

Dans ce chapitre, nous avons vu des exemples de méthodes globales et méthodes locales qui peuvent être employés dans l'inférence de profondeur. Les méthodes globales sont à la base des meilleures méthodes dans la base de données de *Middlebury*. Nous avons présenté un système à base de HMM qui illustre la propagation d'erreur dans un graphe qui est régi par une seule propagation. Ce système est une introduction à la propagation de conviction qui fait la propagation dans tous les sens au niveau de chaque nœud mais aussi de façon itérative. Avec ce traitement intensif, de meilleures images de profondeur sont générés (Weinman, Tran et al. 2008), mais c'est pas facile de les implémenter en temps réel vu qu'ils requièrent des dizaines d'itérations sur chacun des nœud du graphe, une caractéristique qui a encouragé des chercheurs à trouver des simplifications pour réduire le temps d'exécution (Felzenszwalb and Huttenlocher 2006).

Les méthodes locales sont plus rapides et ne sont pas itératifs. La cross-corrélation et la corrélation de phase sont parmi les méthodes locales les plus rapides. En fait, pour chercher deux pixels correspondants, il suffit de faire la corrélation entre deux régions prédéfinies alors que d'autres méthodes locales comparent une région de la vue de gauche avec plusieurs régions de la vue de droite (Yoon and Kweon 2005) pour chercher le correspondant avec une stratégie du gagnant prend le tout. D'autre part, c'est difficile de maintenir avec la corrélation de phase

unidimensionnelle une bonne précision et à la fois détecter les frontières d'objets comme nous avons mentionné dans ce chapitre. Il est intéressant de pouvoir régler ce problème. Un patient qui perçoit son environnement via quelques centaines de phosphènes doit avoir accès à non seulement des estimés de disparité précis, mais aussi des estimés qui sont stables aux frontières d'objets, de sorte qu'il puisse utiliser son raisonnement mentale pour estimer la forme des objets en accumulant suffisamment de l'information de plusieurs cartes de phosphènes reconstruites sur un certain intervalle de temps. Dans le chapitre suivant, nous allons voire comment améliorer la corrélation de phase unidimensionnelle de sorte qu'on ait des résultats qui soient admis dans *Middlebury* en termes de précision et en même temps avoir des images de profondeur qui soient cohérents aux frontières d'objets.

CHAPITRE 2 SYSTÈMES DE STÉRÉOVISION PASSIVE À BASE DE PHASE DÉDIÉS AUX STIMULATEURS INTRA-CORTICAUX VISUELS

2.1 Présentation de l'article

Dans le premier chapitre, nous avons fait un survol de quelques méthodes locales et globales. Nous avons vu que la corrélation de phase unidimensionnelle éprouve une difficulté à maintenir à la fois une bonne précision et une bonne cohérence aux frontières d'objets. Ce problème n'est pas présent dans la corrélation de phase bidimensionnelle qui requiert plus de temps de calcul que la corrélation de phase unidimensionnelle. Dans cette section, on présente en plus de profondeur la corrélation de phase. On intègre un article soumit à la revue Computer Vision and Image Understanding en Juillet 2011. La contribution de cet article est tout d'abord de comprendre les points de faiblesse de la corrélation de phase, de présenter une méthode d'amélioration de la précision de la corrélation de phase unidimensionnelle et aussi de présenter une optimisation de la corrélation de phase bidimensionnelle. La formulation du problème de la stéréovision que nous considérons est celui de l'équation (1.53).

L'article commence par introduire l'implant intra cortical visuel, expliquer l'intérêt de l'emploi de la stéréovision et présenter une revue de littérature touchant différentes approches de résolution du problème de la mise en correspondance. Ensuite, on définit la corrélation de phase et on étudie sa sensibilité à l'amplitude des signaux d'entrée pour proposer une solution qui est l'extraction de caractéristiques. On présente ensuite EPOC, l'amélioration de la corrélation de phase unidimensionnelle. L'optimisation de la corrélation de phase, MMPOC, est présentée après. On fait une évaluation et comparaison des deux systèmes. On finit par décrire l'implémentation de l'EPOC dans le cadre de l'implant intra-cortical visuel. L'implémentation est détaillée dans ANNEXE 1. Nous rappelons que la mesure d'erreur se fait avec les équations présentées dans l'introduction, notamment l'équation (1.10).

2.2 Phase-Based Passive Stereovision Systems Dedicated to Cortical Visual Stimulators

Firas Hawi, and Mohamad Sawan, *Fellow, IEEE*

Polystim Neurotechnologies Laboratory, Departement of Electrical Engineering, Polytechnique Montreal.
firas.hawi@ polymtl.ca

Abstract-In this paper, we design, evaluate and compare two phase-based passive stereovision architectures. We present two approaches to implement phase-based correspondence search algorithms in real time for sparse stereovision applications. The first approach enhances the accuracy of the 1D phase correlation method using a graphical framework. The second approach optimizes the 2D phase correlation method at the cost of degradation in disparity estimation accuracy. We describe techniques to reduce the effects of occlusions on depth inference. Occlusions are present in a scene composed of objects that obstruct other objects in the background and cause erroneous disparity estimation. We report experimental results that encourage the use of the proposed systems in a 3D imaging device dedicated to cover vision for blinds through cortical visual stimulation. A prototype of the selected system is implemented in a FPGA.

Index Terms—Computer Vision, Signal Processing, Stereo, Real-Time Systems, 3D imaging

2.2.1 Introduction

Cortical visual stimulators are subject of many researches that aim to give blind people the ability to have a visual perception of their surrounding environment. Electrodes implanted in the visual cortex of the brain are charged of inducing phosphenes that are visual sensations by electrically

stimulating appropriate neurons [1]. Visual information is wirelessly communicated to the cortical stimulators by an external camera and dedicated processor. Few clinical validations reported demonstration and partial recovery of the visual field [2]. The visual perception is actually composed of dispersed groups of phosphenes, making depth inference by corresponding elements of the left and right eyes views a difficult task for the brain. One solution to that problem is to provide the cortical stimulators with depth information instead of a grayscale picture of the environment. In that case, phosphene intensities will reflect the distance between the patient and objects of the surrounding environment. Depth information may be provided by corresponding elements of the left and right views of a short baseline passive stereovision system that is composed of a pair of cameras mounted on glasses.

Methods of correspondence search can be classified in two categories: Global and local methods. Global methods offer high pixel level disparity estimation accuracy and are robust to occlusions and large disparity jumps. Belief propagation (BP) was used for disparity map inference on Markov Random Fields by optimizing an energy function. Scharstein *et al.* propose to estimate the energy function parameters using a large dataset of stereovision pair images with their disparity maps in a conditional random field framework [3]. Some strategies were developed to optimize the running time of BP [4]. The iterative nature of BP and the message propagation process it employs make it more suited to dense stereovision.

Local methods associate windows in the left and right views of a stereovision pair images. Many correlation-based correspondence search algorithms were implemented in real time [5]. Although they are fast enough for several applications, algorithms that correlate predefined windows for pixel matching show poor results where occlusions and high disparity jumps occur, which is generally the case near objects borders. To solve this issue, local based measures such as Sum of Absolute Differences (SAD) or Sum of Squared Differences (SSD) can be used. The main drawback with these approaches is that they are sensitive to the chosen window size. Some techniques like finding an appropriate window adaptively and choosing an optimal window from a set of predefined windows were suggested [6], [7]. Methods that apply a weighting based on color intensity values and spatial proximity to the pixel of interest proved to be robust in handling occlusions and large disparity jumps and less prone to the window size problem [8].

In our application, we are interested in stimulating a few tens to a few hundreds phosphenes, i.e. we want to develop a sparse stereovision system. For that reason, we construct our system based on local methods. In fact, it is possible to reduce computation time according to the number of phosphenes to stimulate.

The Phase Correlation method [9], also called Phase Only Correlation (POC), was applied in high-accuracy stereovision systems that work in occlusion free environments [10], and was implemented in real time on a PC [11]. Although a POC variant in [11] was proved to be more accurate than some other real time stereovision systems for some datasets, the reconstructed scenes were ambiguous near objects borders. In the other hand, POC is a local method that meets sparse stereovision application needs in terms of efficiency. More specifically, it is possible to make a tradeoff between the number of phosphenes to stimulate and the range of disparity values in a flexible manner when employed with Gaussian pyramids. Since accuracy is an important requirement in our application, we are interested in improving phase correlation displacement estimation accuracy.

In this work, we present two strategies to implement robust phase correlation techniques in a reasonable processing time for sparse stereovision applications. The first contribution is to show that the phase correlation can be put in a graphical framework to yield to better accuracies. The second contribution is to show that the running time of 2DPOC can be optimized algorithmically. The third contribution is to present a real time implementation of a POC variant on FPGA.

In section 2, we review the phase correlation theory and present feature extraction by similarity and proximity. Then, we describe a technique to enhance the accuracy of 1DPOC. The later method is easier to implement than the original 2D version of the phase only correlation, but less accurate. We also introduce a reverse strategy by optimizing the original 2DPOC method. The proposed technique makes a compromise between the number of disparities that can be inferred from the cross-phase spectrum and the accuracy of disparity estimation. In section 3, we present and analyze simulation results to retain one technique to be used in the cortical stimulators application. The implementation of the retained system is described and results are presented.

2.2.2 Methodology

In this section, we start by reviewing phase correlation and basic principles that will be used in the description of the main algorithms. Then, we present the enhanced version of the 1DPOC method and the optimized version of 2DPOC.

4.2.2.1 Basic Principles

We start by reviewing the Phase Only Correlation that was originally defined in the two-dimensional space [9] and describe a 1D version that is faster to execute. Then the sensitivity of phase determination will be analyzed to know what parts of a signal have the biggest contribution to the phase calculation. Finally, a feature extraction mechanism used to reduce the effect of occlusions and large disparity jumps is presented.

The Phase Only Correlation is a method that measures the translation between two images. We denote by (n_1, n_2) the spatial indexes that scan the pixels of an image or subimage such that $n_1 \in \{-N_1, -N_1+1, \dots, N_1\}$ scans the subimage vertically and $n_2 \in \{-N_2, -N_2+1, \dots, N_2\}$ scans the subimage horizontally. Similarly, (k_1, k_2) denote the frequency indexes of the discrete Fourier transform components that scan an image's spectrum. Given two images $f_1(n_1, n_2)$ and $f_2(n_1, n_2)$, and their respective Fourier transforms $F_1(k_1, k_2)$ and $F_2(k_1, k_2)$, such that:

$$\begin{aligned} F_i(k_1, k_2) &= \sum_{n_2=-N_2}^{N_2} \sum_{n_1=-N_1}^{N_1} f_i(n_1, n_2)\, W_{2N_1+1}^{k_1 n_1} W_{2N_2+1}^{k_2 n_2} \\ &= A_{F_i}(k_1, k_2) exp\left(j\theta_{F_i}(k_1, k_2)\right) \end{aligned} \tag{2.1}$$

where θ_{F_i} and A_{F_i} denote the phase and amplitude components of F_i and $W_N^{kn} = \exp(-j2\pi kn/N)$, the cross phase spectrum is expressed as:

$$\begin{aligned} \tilde{C}(k_1, k_2) &= \frac{F_1(k_1, k_2) F_2(k_1, k_2)^*}{|F_1(k_1, k_2) F_2(k_1, k_2)^*|} \\ &= \exp\left(j\left(\theta_{F_1}(k_1, k_2) - \theta_{F_2}(k_1, k_2)\right)\right) \end{aligned} \tag{2.2}$$

where the asterisk (*) denotes the complex conjugate. The Phase Only Correlation function performs the 2D Inverse Fourier Transform of $C(k_1, k_2)$ and yields to a search space $\tilde{c}$:.

$$\tilde{c}(n_1, n_2) = IFFT\left(\tilde{C}(k_1, k_2)\right) \tag{2.3}$$

A peak can be observed at $\tilde{c}(\Delta_1, \Delta_2)$ if $f_1(n_1, n_2)$ is obtained from $f_2(n_1, n_2)$ by a simple translation such that:

$$f_1(n_1, n_2) = f_2(n_1 - \Delta_1, n_2 - \Delta_2) \tag{2.4}$$

Looking for the maximum in the search space yields to knowing the displacement amount (Δ_1, Δ_2).

Since the original version of the Phase Only Correlation allows one to know both vertical and horizontal displacements, calculating the epipolar constraint is not necessary. When image rectification is possible, a reduction in computation resources and time can be achieved by using a 1D version of the Phase Only Correlation at the cost of degradation in pixel disparity estimation accuracy. In order to avoid the computation of a 2DFFT, we set $k_1 = 0$ in (2.1) to obtain:

$$F_i(k_1, k_2)|_{k_1=0} = \sum_{n_2=-N_2}^{n_2=N_2} \sum_{n_1=-N_1}^{n_1=N_1} f_i(n_1, n_2) W_{2N_2+1}^{k_2 n_2} \tag{2.5}$$

Eq. (2.5) can be computed by first integrating $f_i(n_1, n_2)$ in the vertical direction then performing a 1DFFT. This leads to define a one-dimensional version of the cross phase spectrum:

$$\tilde{C}(k_2) = \frac{F_1(k_1, k_2)|_{k_1=0} F_2(k_1, k_2)^*|_{k_1=0}}{\left| F_1(k_1, k_2)|_{k_1=0} F_2(k_1, k_2)^*|_{k_1=0} \right|} \tag{2.6}$$

The search space can then be calculated using a one-dimensional inverse Fourier Transform:

$$\tilde{c}(n_2) = IFFT\left(\tilde{C}(k_2)\right) \tag{2.7}$$

Experiments show that POC exhibits drawbacks when it comes to estimate disparity values near objects borders in a scene composed of multiple objects [11]. It is noticed that regions with higher intensity levels spread their disparity values to neighbouring regions with lower intensity.

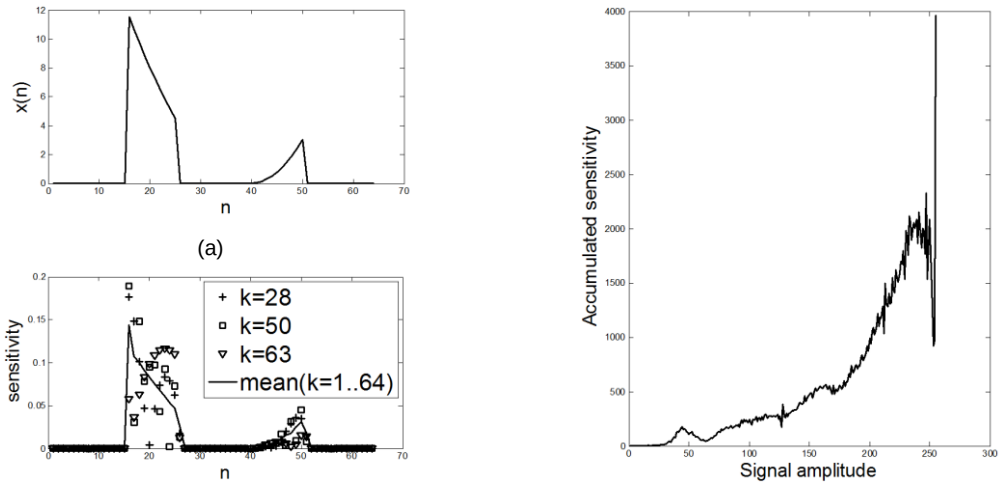


Figure 2.1 Experiment that shows a test signal (a) and the sensitivity $S_n^{\vartheta(n,k)}(n,k)$ analysis on the input samples for 3 frequencies (b) along with the mean sensitivity evaluated over all frequencies. In (c), we show the result of accumulating the mean sensitivities of all possible samples of width 64 taken from the *Teddy* image scanlines. The input signals $x(n)$ in (c) were normalized to have a maximum value of 255. The mean sensitivity is calculated over all frequencies.

Let's suppose we want to use the epipolar constraints and use the POC function to deduce the disparity at a specific location on the reference image. The phase $\theta(k)$ of a frequency component of the spectrum of a signal $x(n)$ of length N defined over the scan line containing the pixel of interest can be calculated using (2.8):

$$\theta(k) = arctan\big(\beta(k)\big) + c \tag{2.8}$$

$$\beta(k) = \frac{\sum_{n=1}^{N} x(n)\sin(2\pi kn/N)}{\sum_{n=1}^{N} x(n)\cos(2\pi kn/N)} \tag{2.9}$$

In (2.8), c is a constant which value depends on the sign of the real part of the Fourier component, it is omitted in the sensitivity calculation. Because the $arctan(.)$ function is continuous and strictly monotonous rising, the sensitivity analysis can be restricted on the $\beta(k)$ terms for simplicity. We do that by noticing that $\beta(k)$ can be written as a linear combination of terms we denote by $\vartheta(n,k)$:

$$\beta(k) = \sum_{n=1}^{N} \vartheta(n,k) \tag{2.10}$$

$$\vartheta(n,k) = \frac{x(n)\sin(2\pi kn/N)}{\sum_{l=1}^{N} x(l)\cos(2\pi kl/N)} \tag{2.11}$$

It can be seen in the example of Fig. 1, where the left image of the *Teddy* image pair was used [12], [13], that the terms composing $\beta(k)$ are sensitive to the amplitude of the input signal using the following formula for sensitivity calculation:

$$S_n^{\vartheta(n,k)}(n,k) = \left|\frac{\partial\vartheta(n,k)}{\partial n}\right|_{n,k} \Big/ \sum\nolimits_n \left(\left|\frac{\partial\vartheta(n,k)}{\partial n}\right|_{n,k}\right) \tag{2.12}$$

Thus, we can assume that low level intensity values have a weak contribution to the phase determination. The disparity values that are calculated based on phase difference at objects having low color intensity levels will eventually be affected by the disparity of neighboring objects with higher color intensity. For that reason, we apply feature extraction before performing phase correlation.

Assuming that pixels belonging to the same object are more likely to have similar disparities when they are close enough to each other, we wish to deform the input signal such that the phase value at each frequency is determined mostly by pixels that have disparity values that are similar to the disparity of the pixel of interest. Inspired by the Gestalt principles of similarity and proximity [8], [14], color based grouping is performed on the area surrounding a referred pixel to account for its pixels membership to a specific object, then proximity weighting is applied to favour nearby pixels and reduce boundary effects in Fourier space.

Color based grouping is achieved by first calculating the color distance between a reference pixel p_α in the image of reference and all pixels in a window $g_i(n_1,n_2)|_{i=1,2}$ such that the window defined over the reference image $g_1(n_1,n_2)$ is centered on p_α. We work in the RGB color space so that the color distance is defined by:

$$D_{\alpha,g_i} = \sum_{c=\{R,G,B\}} |{p_\alpha}^c - g_i^c(n_1,n_2)| = \sum_{c=\{R,G,B\}} |g_1^c(0,0) - g_i^c(n_1,n_2)| \quad (2.13)$$

where the c superscript denotes the color channel.

Similarity is computed using a reference value γ:

$$W_{\alpha,g_i} = [\gamma - min(D_{\alpha,g_i},\gamma)]/\gamma \quad (2.14)$$

Proximity weights $S(n_1,n_2)$ can be calculated directly based on the squared Euclidian distance between the pixels of a window and its center:

$$S(n_1,n_2) = [(N_1^2+N_2^2)-(n_1^2+n_2^2)]/(N_1^2+N_2^2) \quad (2.15)$$

It is possible to use a windowing technique employed to reduce boundary effects in Fourier space to account for the distance between p_α and a neighboring pixel.

Using the color and spatial weights, a pattern can be constructed for each of the left and right views:

$$P_{\alpha,i}(n_1,n_2) = S(n_1,n_2)W_{\alpha,g_i} \quad (2.16)$$

Grouping based on color similarity and spatial proximity is widely used in the literature and allows for more coherent depth maps to be constructed. With phase correlation, however, applying feature extraction is not enough as we shall see in the results section. It actually yields to poor accuracies. However, due to the sensitivity of the phase determination on the intensity level of the input signals, we consider applying this kind of feature extraction important in our application. In the following section, we describe an algorithm that allows for higher accuracies to be achieved with 1DPOC.

4.2.2.2 The enhanced 1DPOC Method

For real-time operating correspondence search algorithms, the 1DPOC method is a substitute of choice for the original 2DPOC. This is due to the fact that it uses 1D Fourier transforms instead of its more time and resource consuming 2D version. The cost that comes with this simplification is a decrease in disparity determination accuracy. For sparse stereovision systems, this is problematic. As the number of disparity estimates decreases, the noise on the discontinuous disparity map plays a more considerable role in making the scene ambiguous. A patient that sees through a map of 100 phosphenes will not be able to rely on the neighborhood of the noised data to better estimate the profile of the environment.

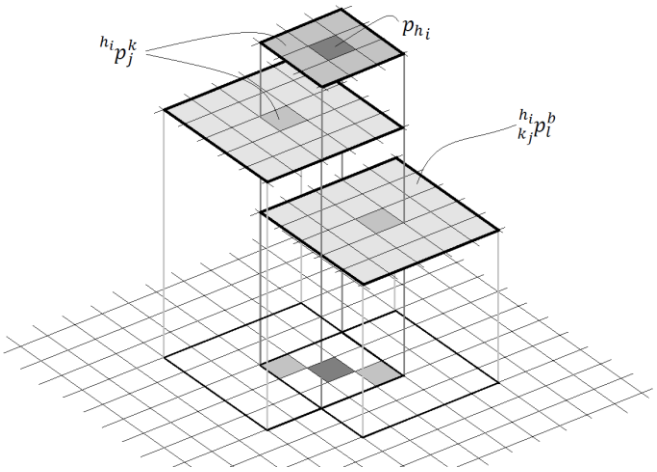


Figure 2.2 Visualization of sets of pixels $\{ {}^{h_i}_{k_j}p_l^b\}$ and $\{{}^{h_i}p_j^k\}$ that are grouped into *b* and *k* clusters

The EPOC (enhanced 1DPOC) is based on manipulating the neighborhood of pixels corresponding to the phosphenes to stimulate. In addition to the pixels of interest, we define two kinds of clusters: the *b-clusters* and the *k-clusters*. We use the notations ${}^{h_i}_{t_j}\xi^s$ and ${}^{h_i}_{t_j}N_u^s$ to denote the clusters and the nodes associated to them respectively. In these notations, h_i refers to the phosphene number i to stimulate, s denotes the type of the cluster and can take the values $\{k, b\}$ for *k-cluster* and *b-cluster*, u refers to the number of the pixel p associated to the current node and t_j indicate the type of the parent node and its subscript j. For every pixel of reference p_{h_i} that is associated to a phosphene h_i to stimulate, we construct a *k-cluster* ${}^{h_i}\xi^k$. We denote by ${}^{h_i}N_j^k$ the nodes that belong to ${}^{h_i}\xi^k$ and by ${}^{h_i}p_j^k$ the pixels associated to them, such that $p_{h_i}\epsilon\{{}^{h_i}p_j^k\}, p_{h_i} =$

${}^{h_i}p_1^k$ and $j\epsilon[1,J]$. For every node belonging to ${}^{h_i}\xi^k$, we construct a state array ${}^{h_i}S_{j,m}^k$ of length M and a *b-cluster* ${}_{k_j}^{h_i}\xi^b = \{{}_{k_j}^{h_i}N_l^b\}, l\epsilon[1,L]$. For every node ${}_{k_j}^{h_i}N_l^b$, there is a pixel ${}_{k_j}^{h_i}p_l^b$ associated to it and a state array ${}_{k_j}^{h_i}S_{l,m}^b$.

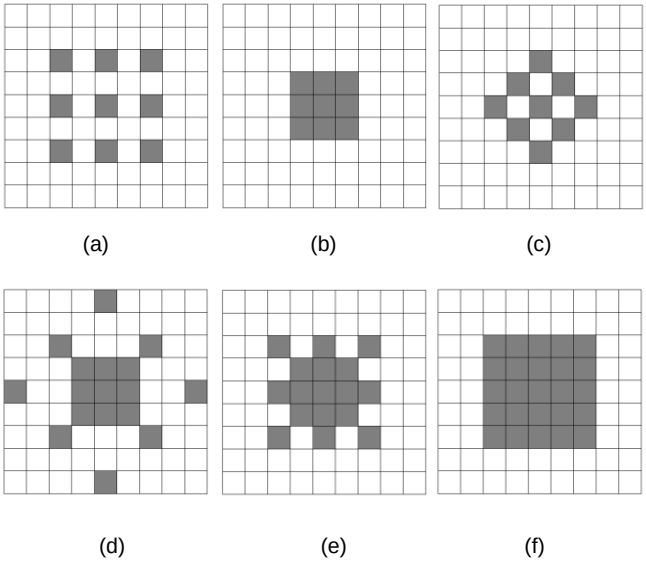


Figure 2.3 Possible patterns that can be used to describe $\{{}_{k_j}^{h_i}p_l^b\}$. Pattern1 is shown in (a), pattern 2 in (b), pattern 3 in (c), pattern 4 in (d), pattern 5 in (e) and pattern 6 in (f)

The set of pixels $\{{}^{h_i}p_j^k\}$ is chosen to be centered on p_{h_i}. An example is illustrated in Fig. 2.2 where $\{{}^{h_i}p_j^k\}$ and $\left\{{}_{k_j}^{h_i}p_l^b\right\}$ form square shaped sets of pixels centered on p_{h_i} and ${}^{h_i}p_j^k$ respectively. There are many ways to define the shape taken by the group of pixel $\left\{{}_{k_j}^{h_i}p_l^b\right\}$. Fig. 3 illustrates a few cases that will be used in system evaluation later in this paper.

Message propagation and disparity estimation are described by the following steps:

Step1: For each of the two views, construct L patterns using (2.16), one for each pixel belonging to a *b-cluster*. The constructed patterns can be described by the following expression:

$$pattern(l) = P_{{}_{k_j}^{h_i}p_l^b,i}(n_1,n_2), l = 1..L \tag{2.17}$$

Step2: In (2.5), replace $f_i(n_1, n_2)$ by the patterns calculated in step 1 and compute a search space ${}^{h_i}_{k_j}\tilde{c}^b_l(n_2)$ for every pair of patterns using (2.6) and (2.7) to determine the state arrays values:

$$ {}^{h_i}_{k_j}S^b_{l,m} = {}^{h_i}_{k_j}\tilde{c}^b_l(m) \tag{2.18}$$

Step3: Deduce the state array of the current *k-cluster* node by summing up the state arrays of the associated *b-cluster* nodes:

$$ {}^{h_i}S^k_{j,m} = \sum_{l=1}^{L} {}^{h_i}_{k_j}S^b_{l,m} \tag{2.19}$$

Step4: Find the disparity value ${}^{h_i}d^k_j$ associated to ${}^{h_i}N^k_j$ from the argument of the maximum value of ${}^{h_i}S^k_{j,m}$:

$$ m_{max} = \underset{1 \le m \le M}{\operatorname{argmax}} \; {}^{h_i}S^k_{j,m} \tag{2.20}$$

Step5: Repeat steps 1-4 for every node ${}^{h_i}N^k_j$.

Step6: For every disparity value ${}^{h_i}d^k_j$ calculated in step 4, compute the energy function given by:

$$ {}^{h_i}E^T_j = w_t \, {}^{h_i}E^t_j + w_u \, {}^{h_i}E^u_j + w_v \, {}^{h_i}E^v_j \tag{2.21}$$

where, assuming that $g_1(n_1, n_2)$ is defined over the left view:

$$ {}^{h_i}E^t_j = \left| {}^{h_i}d^k_j - {}^{h_i}d^k_1 \right| $$

$$ {}^{h_i}E^u_j = \left| g_2(n_{1\alpha}, n_{2\alpha} - {}^{h_i}d^k_j) - g_1(n_{1\alpha}, n_{2\alpha}) \right| $$

$$ {}^{h_i}E^v_j = |g_1(n_{1\alpha}, n_{2\alpha}) - g_1(0,0)| $$

$$ \alpha = {}^{h_i}p^k_j $$

Step7: The disparity value ${}^{h_i}d$ associated to p_{h_i} is the one that minimize the energy function in (2.21).

$$^{h_i}d = {}^{h_i}d_j^k \Big/ \left\{ j = \underset{1 \le j \le J}{\operatorname{argmin}} \; {}^{h_i}E_j^T \right\} \tag{2.22}$$

The procedure described in steps 1-7 needs to be performed for each phosphene h_i.

Since the EPOC method requires to manipulate the disparity at the neighborhood of the pixels of interest, it turns out that computation burden is increased for sparse stereovision. In dense stereovision systems, it is possible to simplify the implementation by noticing that a computed search space can be used to enhance the accuracy of disparity estimation at other neighboring pixels.

4.2.2.3 The Multiple Modes 2DPOC Method

In this section, we explore a different strategy to implement the phase only correlation method. Instead of enhancing the robustness of a fast correlation technique, we try to optimize a robust technique to meet execution time constraints. The proposed algorithm aims to improve the execution time of the 2DPOC method when multiple disparity values are required to be calculated by the application. More specifically, we want the cross phase spectrum defined in (2.2) to contain enough information to infer V disparity values instead of only one. The multiple modes 2DPOC (MM2DPOC or simply MMPOC) method allows to do that in exchange for a deterioration in disparity estimation accuracy.

In order to optimize the 2DPOC method, we first pose the hypothesis that the main information in an image spectrum is contained in its low frequencies.

Consider a set of V mode pairs $\{(\varphi_\nu, \rho_\nu)\}$ where $\nu \in [1, V]$, $\varphi_\nu \in [0,1]$ and $\rho_\nu \in [0,1]$, and a set of V pairs of images $\{f_1^\nu(n_1, n_2)\,, f_2^\nu(n_1, n_2)\}$. The goal is to calculate the displacement amount between the two images of each pair. Instead of performing the 2DPOC on each image pair individually, we create a two-dimensional superposition array x_i for each of the first ($i = 1$) and second ($i = 2$) views. The two superposition arrays are calculated according to:

$$x_i(n_1, n_2)|_{i=1,2} = \sum_{\nu=1}^{V} f_i^\nu(n_1, n_2) e^{-j2\pi n_1 \varphi_\nu} e^{-j2\pi n_2 \rho_\nu} \tag{2.23}$$

Once $x_1(n_1, n_2)$ and $x_2(n_1, n_2)$ are calculated, their Fourier transforms $X_1(n_1, n_2)$ and $X_2(n_1, n_2)$ are deduced using (2.1). The cross phase spectrum is then calculated using (2.2):

$$\tilde{C}^{sup}(k_1, k_2) = \frac{X_1(k_1, k_2)X_2(k_1, k_2)^*}{|X_1(k_1, k_2)X_2(k_1, k_2)^*|} \tag{2.24}$$

The spectrum of the superposition array X_i can be seen as the superposition of shifted image spectrums:

$$X_i(k_1, k_2) = \sum_{v=1}^{V} F_i^v\left(k_1 - c_{\varphi_v}, k_2 - c_{\rho_v}\right) \tag{2.25}$$

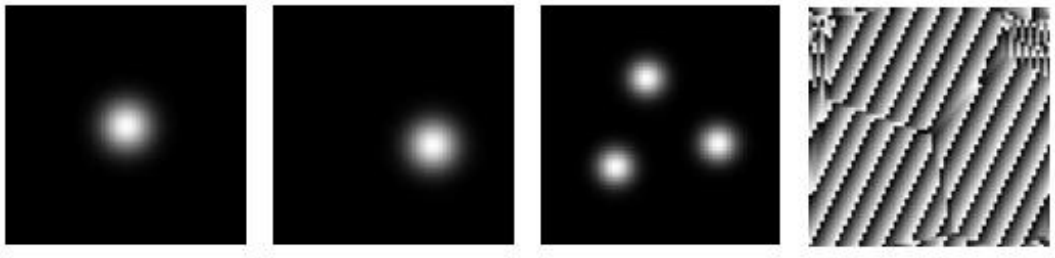
where the shifted image spectrums centers are defined as:

$$\begin{aligned} c_{\varphi_v} &= \varphi_v(2N_1 + 1) \\ c_{\rho_v} &= \rho_v(2N_2 + 1) \end{aligned} \tag{2.26}$$

Because we made the hypothesis that the main information in an image's spectrum is contained in its low frequencies, we expect that the amplitude and phase values of the superposition array spectrum at a specific frequency are almost equal to the amplitude and phase values of the image spectrum which center is the nearest to the frequency of interest.

$$X_i(k_1, k_2) \approx F_i^v\left(k_1 - c_{\varphi_v}, k_2 - c_{\rho_v}\right)\Big|_{(k_1,k_2)\in ne(c_{\varphi_v}, c_{\rho_v})} \tag{2.27}$$

It follows that the phase values of the cross phase spectrum that are useful to calculate the displacement amount between two specific images $\{f_1^v(n_1, n_2), f_2^v(n_1, n_2)\}$ can be read from $\tilde{C}^{sup}$ at frequencies that are nearest to the center $(c_{\varphi_v}, c_{\rho_v})$ associated to the images of interest, with additive noise. An example is shown in Fig. 4.



(a) (b) (c) (d)

Figure 2.4 Experiment that illustrates the information contained in the MMPOC method cross-phase spectrum. Given a Reference image (a) and an image that is shifted from the original (b) by an amount of (5 , 10), we use 3 modes to generate the phase information 3 times in different emplacements of the cross phase spectrum. The amplitude of $X_1(k_1, k_2)$ is shown in (c). In (d), we illustrate the phase of the superposition cross-phase spectrum $\widetilde{C}^{sup}$.

At this level, the phase information that is necessary to deduce the disparity of V pairs of images is contained in the superposition cross phase spectrum $\widetilde{C}^{sup}$. To find those disparity values, we proceed by extracting each of the V regions of interest that contain the necessary phase information using windows that are centered on the shifted centers of the original images' spectrum $(c_{\varphi_v}, c_{\rho_v})$. We choose rectangular windows of dimensions (N_3, N_4) for simplicity so that the regions of interest R^v are deduced using the following equation:

$$\begin{aligned} R^v(k_1, k_2) &= R^{v'}\left(k_1 + c_{\varphi_v}, k_2 + c_{\rho_v}\right) \\ R^{v'}(k_1, k_2) &= \tilde{C}^{sup}(k_1, k_2) Rect\left(k_1 - c_{\varphi_v}, k_2 - c_{\rho_v}\right) \end{aligned} \tag{2.28}$$

The disparity values associated to each of the regions of interest are deduced independently using (2.3).

We apply the MMPOC method using a 2D hanning window to calculate the proximity weights for feature extraction and use an energy minimization mechanism similar to the one described in step 6 of the EPOC method description. We also apply Gaussian filtering to the input image pairs to reduce interference.

2.2.3 Results and implementation

The proposed algorithms are evaluated using the Middlebury dataset [12], [13]. Error rates are reported in Table 1 and Table 2. We compare our results with three other phase based stereovision systems. The first is a 1DPOC variant that was implemented in real time on a PC with low pass post-filtering [11], the second is based on phase difference [15] and the third is a phase based algorithm that aims to improve depth map inference of a scene composed of slanted objects [16]. Fig. 7 depicts disparity maps of the proposed algorithms. After system comparison and selection we make further comparison with existing methods and describe an implementation on FPGA.

Tableau 2.1 EPOC results and comparison

Method		Tsukuba			Venus			Teddy			Cones			Average
		Nonoc	All	Disc	Nonoc	All	Disc	Nonoc	All	Disc	Nonoc	All	Disc	Error(%)
Alba[11]		7.86	9.78	29.1	6.06	7.65	45.2	37.0	43.3	50.1	22.5	31.0	42.3	**27.6**
Proposed (EPOC)	(S1)	5.49	7.34	15.5	9.13	10.5	28.0	15.2	23.6	28.0	14.6	23.7	24.3	**17.1**
	(S2)	4.84	6.62	14.8	9.71	11.0	27.3	16.5	24.8	30.0	15.5	24.5	25.7	**17.6**
(S5)		14.6	16.0	26.3	29.4	30.4	44.5	36.9	43.0	50.4	38.9	45.4	49.0	**35.4**

Nonoc: Non occluded regions; All: All regions; Disc: Disclosed regions.

Tableau 2.2 MMPOC results and comparison

Method		Tsukuba			Venus			Teddy			Cones			Average
		Nonoc	All	Disc	Nonoc	All	Disc	Nonoc	All	Disc	Nonoc	All	Disc	Error(%)
Slanted[16]		4.26	6.53	15.4	6.71	8.16	26.4	14.5	23.1	25.5	10.8	20.5	21.2	**15.3**
Etriby[15]		4.89	7.11	16.3	8.34	9.76	26.0	20.0	28.0	29.0	19.8	28.5	27.5	**18.8**
Proposed (MMPOC)	(S0)	2.88	4.80	10.5	6.55	7.82	17.4	14.4	22.1	27.9	15.2	22.7	24.5	**14.7**
	(S3)	4.28	6.27	14.2	10.7	12.1	25.6	18.4	26.6	32.9	18.4	26.6	28.6	**18.7**
	(S4)	5.78	7.79	16.6	12.9	14.4	31.7	22.4	30.3	36.7	24.6	32.6	34.5	**22.5**

Nonoc: Non occluded regions; All: All regions; Disc: Disclosed regions.

4.2.3.1 EPOC Results

System parameter values used in the simulations are mentioned in Table 3. The EPOC algorithm is simulated with two different values of the window height N_1. S1 and S2 denote the two configurations of the algorithm. We set $N_1 = 2$ for S1 and $N_1 = 4$ for S2. Both configurations use pattern 6 illustrated in Fig. 3. Image pyramids of depth $d_{pyr} = 3$ were used. In Table 1, EPOC results are compared against another existing 1DPOC based system [11].

Tableau 2.3 EPOC parameters values

Parameter	Value	Parameter	Value
N_2	31	w_t	0.1
J	9	w_u	2
γ	30	w_v	2
L	25	N_1	S1: 2; S2: 4

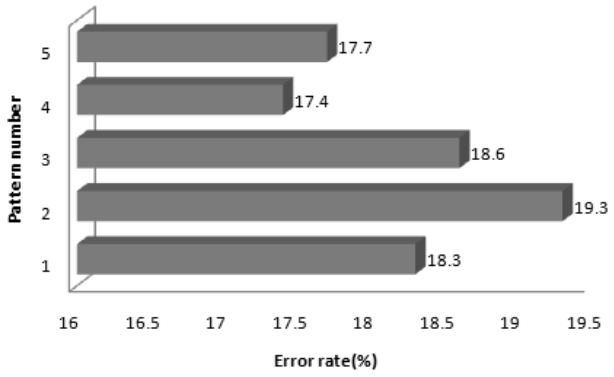


Figure 2.5 Error rates obtained with different patterns with $N_1 = 2$.

Fig. 5 shows the mean error rate obtained for the EPOC system using the other pattern configurations shown in Fig. 3. It can be seen that system accuracy is improved when the patterns are extended to pixels that are further away from the centers and when the number of the retained pixels is bigger. Table 1 provides results for a simple POC system with color similarity and spatial proximity weightings. This system is denoted by S5 ($N_1 = 2$). We conclude that the accuracy of the EPOC algorithm is mainly due to the message propagation strategy it employs. Using Gestalt principles, however, contributes to the coherency of the reconstructed disparity maps (Fig. 7).

4.2.3.2 MMPOC Evaluation

In order to evaluate the MMPOC algorithm, two configurations containing different numbers of modes are introduced. The first configuration is denoted by S3, it considers 2 modes taking the values $(\varphi_1, \rho_1) = (0\,, 0)$ and $(\varphi_2, \rho_2) = (0.5\,, 0.5)$. The second configuration is denoted by S4, it contains 3 modes taking the values $(\varphi_1, \rho_1) = (0\,, 0)$, $(\varphi_2, \rho_2) = (0.67\,, 0.33)$ and $(\varphi_3, \rho_3) = (0.33\,, 0.67)$.

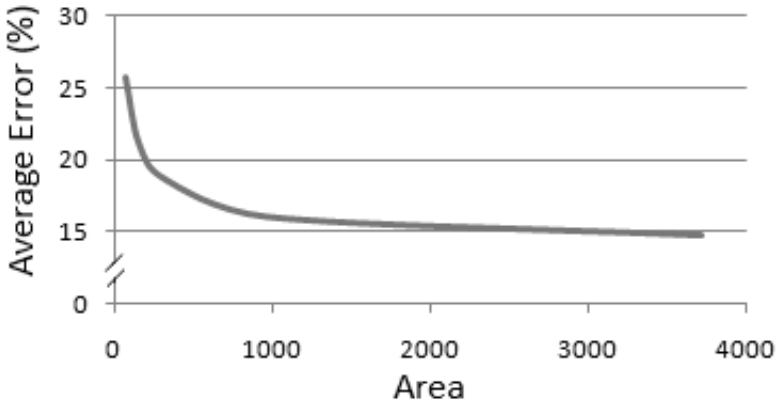


Figure 2.6 Error rate obtained with different spectral band dimensions. Selected dimensions are mentioned in Table 4. The area refers to the number of pixels involved in the spectral band.

TABLEAU 2.4 MMPOC SIMULATION RESULTS WITH 1 MODE

N_3	N_4	Area	Av. Error (%)
3	21	63	25.7
7	21	147	21.1
15	21	315	18.6
31	31	961	16.0
61	61	3721	14.7

(N_3, N_4) are the dimensions of the spectral band. Area = $N_3 \times N_4$.

The dimensions of the rectangular windows used to extract phase data before applying the IFFT must be small enough to limit the effect of interferences between shifted spectrums but also wide enough to extract much relevant information from the superposition spectrum. Fig. 6 illustrates the average error when using rectangular windows of different areas and only one mode. More detailed information about the experiment is shown in Table 4. We note that the graph shape may

vary when using different spectral window dimensions for the same area values, but we can still have an idea of the role of the retained spectral band in the system accuracy.

Taking into account the presence of interferences between shifted spectrums, we use different window dimensions for the two configurations, and choose windows of dimensions (21,21) for the S3 configuration and (15,15) for the S4 configuration.

We notice that when using a band of dimensions (61,61), we get the best average error among the considered configurations, so we denote that configuration by S0 and we show more detailed evaluation information in Table 1. It is noticed that while increasing the number of modes used in the MMPOC algorithm, system accuracy degrades as it becomes more prone to the effects of interferences between shifted spectrums. However, the performance of the system is enhanced at the same time.

Since the 2DPOC method is a more advanced method than the 1DPOC method, we chose to compare its accuracy against other available phase based systems. These methods, described in [15] and [16], were not designed to be run in real-time, but focus on improving disparity estimation accuracy with slanted objects. We notice that the MMPOC setup S0 yield better results, and that in configuration S3, the result is still comparable. In configuration S0, 6 Fourier transforms are performed for each pair of disparity values. In configuration S3, only 4 are needed.

4.2.3.3 System selection

The MMPOC method yields to smoother disparity maps (Fig. 7) but requires the computation of 2D Fourier transforms. In the other hand, EPOC can have a flexible architecture that allows us to make a trade-off between disparity estimation accuracy and performance easily, by selecting an appropriate pattern that describes the distribution of $\{ {}^{h_i}_{k_j}p_l^b \}$. In terms of evaluation, it can be noticed that when simulated with patterns 1,3,4,5 and 6, the EPOC method yields to a smaller average error than the MMPOC method that uses only two modes. For those reasons, we choose to implement the EPOC algorithm for the cortical visual stimulators application. Moreover, we will test the stimulation of a set of 10x10 electrodes in real time using system configuration S2. Although the S1 configuration is more accurate and easier to implement, we aim to implement a configuration that is more robust to epipolar misalignments and camera noise that may cause

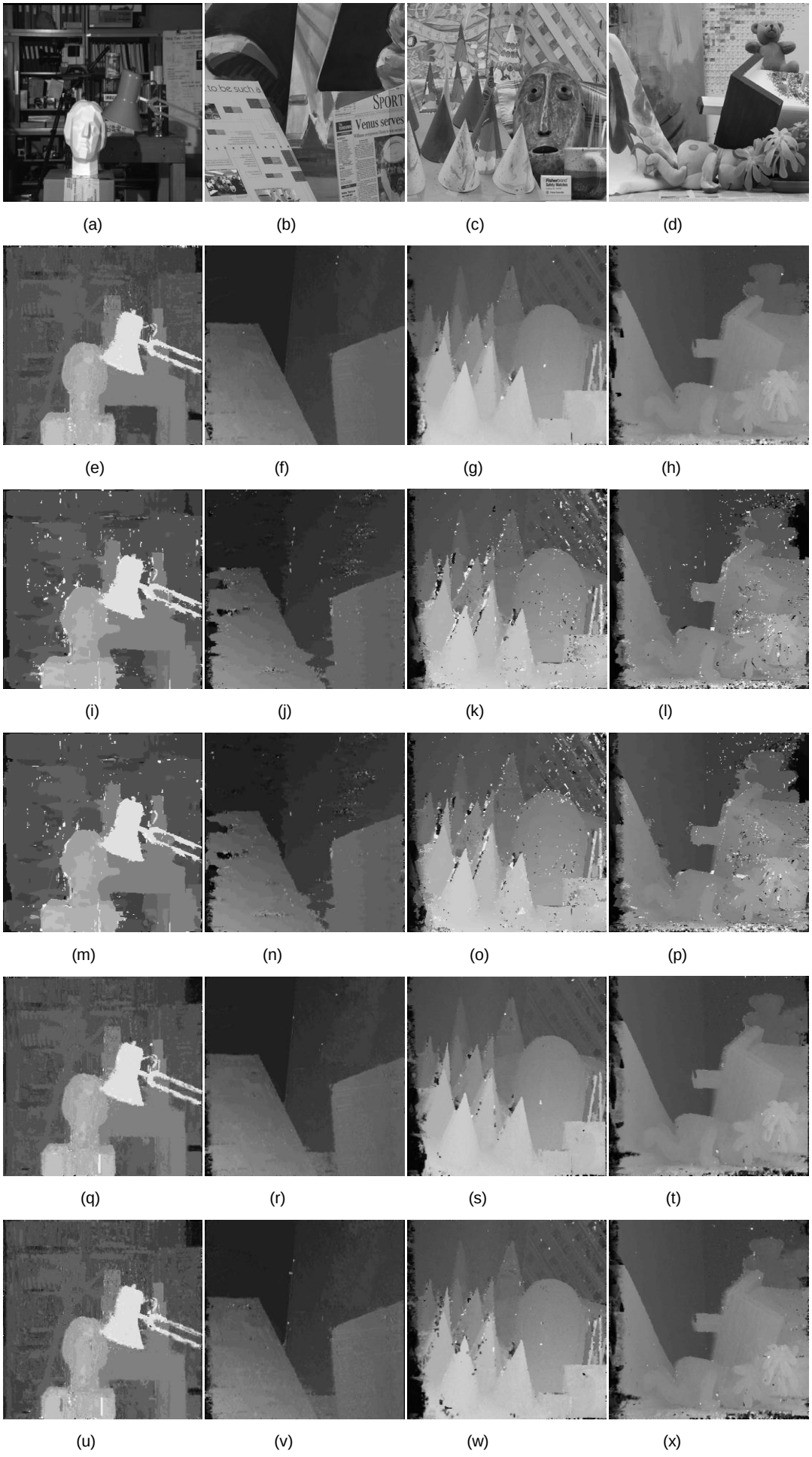


Figure 2.7 Reference images used in the simulations (a-d) and disparity maps for the S0 (e-h), S1 (i-l), S2 (m-p), S3 (q-t) and S4 (u-x) configurations.

erroneous disparity estimations. Actually, we want our system to operate with compact cameras that are portable on glasses. The image quality of compact picture capturing devices like webcams are likely to be lower than the quality of images provided by the cameras used to generate the Middlebury dataset. In Table 5, we make a comparison between the proposed (S2) configuration and main methods dedicated to real-time and near real-time applications.

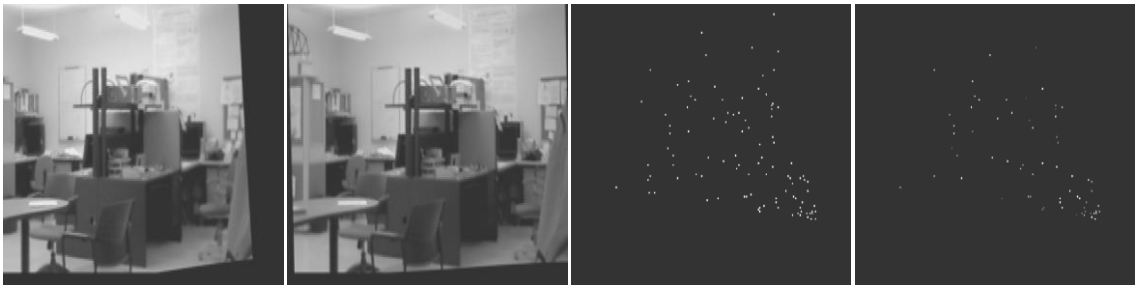
Tableau 2.5 System comparison for the tsukuba scene

Method	nonocc	all	Disc
Real-time GPU [17]	2.05	4.22	10.6
Real-time Var [18]	3.33	5.48	16.8
⋮			
Proposed (S2)	**4.84**	**6.62**	**14.8**
⋮			
ALBA[11]	7.86	9.78	29.1
SAD [19]	12.1	13.4	28.2
LWPC [20]	-	14.16	38.98

Nonoc: Non occluded regions; All: All regions; Disc: Disclosed regions.

4.2.3.4 Implementation

In this section, we describe the implementation of the EPOC algorithm. We rectify a pair of stereovision images that were captured in an environment that contains specular reflections using two webcams as seen in Fig. 8. We want to stimulate a set of 10x10 electrodes. Inspired by the work done in [1], we generate a map of 100 phosphenes. The main architecture of the system is described in Fig. 9.



(a) (b) (c) (d)

Figure 2.8 Implementation results. The reference image is the right view shown in (a), the left view is shown in (b). The generated 100 phosphene map is shown in (c). In (d) we show the obtained disparity map.

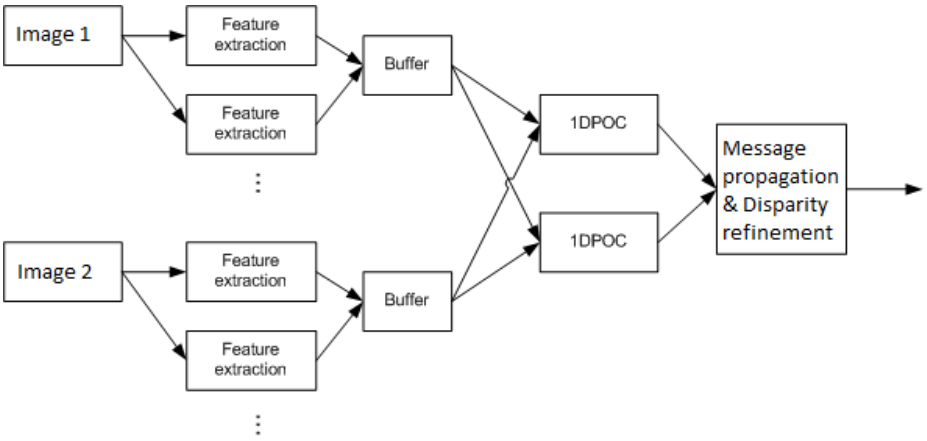


Figure 2.9 Main building blocks of the implemented system.

The design targets a Virtex 6 FPGA from Xilinx. There are 14 feature extraction blocks that operate in parallel to perform color and spatial weighting. At each one of these blocks, the 2D signals are transformed to 1D vectors prior to performing 1DFFT. The proximity weights are supplied to these blocks by an external generator. Two 1DPOC blocks are implemented to enhance the execution speed of the algorithm. The message propagation and disparity refinement block uses patterns of 5x5 pixels to define the distribution of the $\{{}^{h_i}_{k_j}p_l^b\}$ pixels. Given that the number of $k-cluster$ pixels J is equal to 9 and that the pixels belonging to a $k-cluster$ are grouped into a square shaped pattern, we manage to use the 1D search spaces calculated in step 2

of the disparity refinement process of the EPOC algorithm to determine the state arrays ${}^{h_i}S^k_{j,m}$ of multiple $k-cluster$ nodes. Thus, for each phosphene to stimulate, we need to compute the search space for each one of pixels forming a 7x7 pattern. The design was made using System Generator and has the ability to provide disparity maps (100 phosphenes) for less than 5ms with a clock cycle period of 6.3ns.

With the same number of phase-correlations performed, it is possible to apply the EPOC method to dense stereovision and generate a disparity map of about 170×170 pixels. This is because for each phosphene to stimulate, 49 POCs are required by the S2 configuration. And so, given that a calculated search space can be used to enhance the accuracy of all neighboring disparities, and since the most costly bloc in the implementation is the POC bloc, a depth map of $100 \times 49 \times 6 \approx 170 \times 170$ pixels can be generated at 30fps. Table 6 shows specifications of the EPOC implementation and presents other real-time stereo implementations specifications. It can be seen that our method can achieve high disparity range with a relatively good accuracy (Table 5) in real time.

Tableau 2.6 Comparison of system specifications

Method	Platform	Image size	Range*	Rate
Realtime GPU [17]	3.0Ghz PC with a Radeon XL1800 GPU	320×240	**16**	**43fps**
Realtime Var [18]	2.83GHz PC, Intel CPU	-	**22**	**<3.5fps**
ALBA[11]	Intel Core 2 Duo, 2.4 Ghz	256×256	**48**	**110fps**
SAD [19]	Stratix I-IV	750×400	**60**	**60fps**
LWPC [20]	Xilinx Virtex2000E FPGA	256×360	**20**	**30fps**
This Work	Xilinx Virtex 6 FPGA	Sparse(100)	**63**	**200fps**

** Range of disparity values*

2.2.4 Conclusion

In the present work, we designed and evaluated two phase based correspondence search algorithms. In the first algorithm, EPOC, we enhanced the accuracy of the 1DPOC, a method that was executed in real time on a PC. In the second algorithm, we optimized the 2DPOC algorithm by making the cross phase spectrum richer in information to be able to infer many disparity values from one computation. After performing system evaluation and comparison, we implemented the EPOC algorithm on FPGA as a prototype to validate its performance in cortical

visual stimulator applications. Sparse stereovision applications require to be operated by a local method for efficiency. Local methods are mostly characterized by low disparity estimation accuracies relatively to global methods. In EPOC, phase correlation was improved by combining it with a message propagation approach. The fact that we used only one iteration in message propagation allowed us to keep to some extends the benefits of local methods that come with phase correlation. A compromise between performance, accuracy and disparity range could be achieved to fulfill the needs of our application. A disparity range of 63 could be achieved at 200fps for the sparse stereovision system of our application and more can be achieved with Gaussian pyramids.

2.2.5 Acknowledgement

The authors would like to acknowledge support from NSERC and the Canada Research Chair on Smart Medical Devices.

2.2.6 References

[1] L. X. Buffoni, et al., "An image processing system dedicated to cortical visual stimulators," in CCECE 2003 Canadian Conference on Electrical and Computer Engineering: Toward a Caring and Humane Technology, May 4, 2003 - May 7, 2003, Montreal, Canada, 2003, pp. 1497-1500.

[2] E. M. Schmidt, et al., "Feasibility of a visual prosthesis for the blind based on intracortical micro stimulation of the visual cortex," Brain, vol. 119, pp. 507-522, April 1, 1996.

[3] D. Scharstein and C. Pal, "Learning conditional random fields for stereo," in 2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR'07, June 17, 2007 - June 22, 2007, Minneapolis, MN, United states, 2007.

[4] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient belief propagation for early vision," International Journal of Computer Vision, vol. 70, pp. 41-54, 2006.

[5] H. Hirschmuller, et al., "Real-time correlation-based stereo vision with reduced border errors," International Journal of Computer Vision, vol. 47, pp. 229-246, 2002.

[6] O. Veksler, "Stereo correspondence with compact windows via minimum ratio cycle," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, pp. 1654-60, 2002.

[7] A. F. Bobick and S. S. Intille, "Large occlusion stereo," International Journal of Computer Vision, vol. 33, pp. 181-200, 1999.

[8] K.-J. Yoon and I.-S. Kweon, "Locally adaptive support-weight approach for visual correspondence search," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005, June 20, 2005 - June 25, 2005*, San Diego, CA, United states, 2005, pp. 924-931.

[9] C. D. Kuglin and D. C. Hines, "The phase correlation image alignment method," in Proceedings of the 1975 International Conference on Cybernetics and Society, 23-25 Sept. 1975, New York, NY, USA, 1975, pp. 163-5.

[10] K. Takita, et al., "High-accuracy subpixel image registration based on phase-only correlation," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol. E86-A, pp. 1925-34, 2003.

[11] A. Alba and E. Arce-Santana, "Phase-Correlation Guided Search for Realtime Stereo Vision," in Combinatorial Image Analysis. 13th International Workshop, IWCIA 2009, 24-27 Nov. 2009, Berlin, Germany, 2009, pp. 212-23.

[12] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," International Journal of Computer Vision, vol. 47, pp. 7-42, 2002.

[13] D. Scharstein and R. Szeliski. (2001), Middlebury stereo vision page [website]. Available: http://vision.middlebury.edu/stereo

[14] L. Nalpantidis and A. Gasteratos, "Biologically and psychophysically inspired adaptive support weights algorithm for stereo correspondence," Robotics and Autonomous Systems, vol. 58, pp. 457-464, 2010.

[15] S. El-Etriby, et al., "Dense depth map reconstruction by phase difference-based algorithm under influence of perspective distortion," Ul Ordona 21, Warsaw, 01-237, Poland, 2006, pp. 349-361.

[16] S. El-Etriby, et al., "Dense stereo correspondence with slanted surface using phase-based algorithm," in 2007 IEEE International Symposium on Industrial Electronics, ISIE 2007, June 4, 2007 - June 7, 2007, Caixanova - Vigo, Spain, 2007, pp. 1807-1813.

[17] L. Wang, et al., "High-quality real-time stereo using adaptive cost aggregation and dynamic programming," in 3rd International Symposium on 3D Data Processing, Visualization, and Transmission, 3DPVT 2006, June 14, 2006 - June 16, 2006, Chapel Hill, NC, United states, 2007, pp. 798-805.

[18] S. Kosov, et al., "Accurate Real-Time Disparity Estimation with Variational Methods," in Advances in Visual Computing. 5th International Symposium, ISVC 2009, 30 Nov.-2 Dec. 2009, Berlin, Germany, 2009, pp. 796-807.

[19] K. Ambrosch and W. Kubinger, "Accurate hardware-based stereo vision," Computer Vision and Image Understanding, vol. 114, pp. 1303-16, 2010.

[20] A. Darabiha, et al., "Reconfigurable hardware implementation of a phase-correlation stereo algorithm," Machine Vision and Applications, vol. 17, pp. 116-32, 2006.

CHAPITRE 3 DISCUSSION SUR EPOC

Ce chapitre explique l'algorithme EPOC en plus de profondeur. EPOC est mis dans un contexte graphique et est comparé avec la version Alba dans un contexte Bayesien. Ce chapitre commence par une formulation du problème de la stéréovision qui est adaptée à la corrélation de phase pour aboutir à une formulation mathématique de l'algorithme EPOC, suivie par une description graphique de l'algorithme.

3.1 Formulation

La formulation sur laquelle l'algorithme EPOC repose est l'équation (1.53). Contrairement à l'algorithme d'Alba, on tente de segmenter les images stéréoscopiques en des régions. La formulation s'écrit toujours sous la forme:

$$\sum_{m=1}^{M} R_1^{m,a_m,b_m}(t) = \sum_{m=1}^{M} \alpha^m R_2^{m,a_m,b_m}(t-\beta^m) \qquad (3.1)$$

Cette fois, les régions R_i^{m,a_m,b_m} ne sont pas tirées directement des images stéréoscopiques, mais expriment une distribution de probabilité. Cette distribution exprime la probabilité qu'un pixel $p_i^{m,t}$ appartienne au même objet à lequel appartient un pixel de référence $p_1^{m,0}$. Dans cette notation, $i = \{1,2\}$ est l'indice de l'image à laquelle appartient le pixel $p_i^{m,t}$, m est l'indice de la région R_i^{m,a_m,b_m} à laquelle appartient le pixel $p_i^{m,t}$ et t est l'indice du pixel à l'intérieur de cette région. Notons que l'image à laquelle appartient le pixel de référence est toujours l'image 1 ($i = 1$). Dans notre cas, $t \in [-31,31]$ et le pixel de référence possède un $t = 0$. En fait, dans le chapitre 2, nous avons fixé la taille des fenêtres d'étude à 63. Ceci suggère que $|b_m - a_m| = 63$.

Par $obj(p_i^{m,t})$, on note l'objet ou portion d'objet qui contient le pixel $p_i^{m,t}$. Ainsi, avec EPOC, les régions R_i^{m,a_m,b_m} sont représentées par la probabilité qu'un pixel qui se trouve dans la première ou la deuxième image stéréoscopique soit l'image du même objet à lequel appartient un pixel de référence qui est défini exclusivement dans la première image :

$$R_i^{m,a_m,b_m}(t) = P(p_i^{m,t} \in obj(p_1^{m,0}) | couleur(p_i^{m,t}), couleur(p_1^{m,0}), t) \qquad (3.2)$$

Dans cette définition des régions, $couleur\left(p_i^{m,t}\right)$ réfère à la couleur du pixel $p_i^{m,t}$. Cette probabilité est conditionnée sur la couleur du pixel courant $p_i^{m,t}$ et celle du pixel de référence $p_1^{m,0}$, mais aussi sur la position du pixel courant t en vertu de l'équation (2.16). Nous rappelons que l'équation (2.16) décrit la procédure de segmentation d'un signal en régions et compte sur la position relative du pixel par rapport au centre et de la couleur du pixel.

D'autre part, la corrélation de phase elle-même est sensible à l'amplitude des vecteurs d'entrée. C.à.d., s'il existe deux régions dans la fenêtre d'étude qui ont deux disparités différentes, la valeur de disparité inférée à partir de la corrélation de phase sera éventuellement celle de la région qui a l'intensité la plus élevée. Cette propriété est démontrée expérimentalement dans Figure 2.1 avec l'étude de sensibilité de la phase. La raison de montrer cette sensibilité sous forme graphique est que c'est plus facile de la voir qu'avec les équations. Ceci dit, il est favorable d'exprimer l'espace de recherche (2.3) de la corrélation de phase sous forme de distribution de probabilité qui soit conditionné sur l'amplitude des régions. Notons que l'espace de recherche est normalisé de sorte que la somme de tous ses éléments vaut l'unité du fait que son spectre est normalisé dans (2.2), et donc c'est favorable d'exprimer l'espace de recherche de la corrélation de phase $\tilde{c}(n_2)$ comme distribution de probabilité.

$$\tilde{c}(n_2) = P(disp\left(p_1^{m,0}\right)|R_1^{m,a_m,b_m}(t), R_2^{m,a_m,b_m}(t)) \tag{3.3}$$

Maintenant, d'après la loi de Bayes:

$$\begin{aligned} &P\left(disp\left(p_1^{m,0}\right)\middle|R_1^{m,a_m,b_m}(t), R_2^{m,a_m,b_m}(t)\right) P\left(R_1^{m,a_m,b_m}(t), R_2^{m,a_m,b_m}(t)\right) \\ &\quad = P\left(R_1^{m,a_m,b_m}(t), R_2^{m,a_m,b_m}(t)\middle|disp\left(p_1^{m,0}\right)\right) P(disp\left(p_1^{m,0}\right)) \end{aligned} \tag{3.4}$$

L'inconnue qu'on souhaite chercher est $P(disp\left(p_1^{m,0}\right))$, la distribution de probabilité sur toutes les valeurs de disparités possibles pour inférer la disparité au pixel qui est le centre de la région m. On peut écrire :

$$\begin{aligned} &P(disp\left(p_1^{m,0}\right)) \\ &= \frac{P\left(disp\left(p_1^{m,0}\right)\middle|R_1^{m,a_m,b_m}(t), R_2^{m,a_m,b_m}(t)\right) P\left(R_1^{m,a_m,b_m}(t), R_2^{m,a_m,b_m}(t)\right)}{P\left(R_1^{m,a_m,b_m}(t), R_2^{m,a_m,b_m}(t)\middle|disp\left(p_1^{m,0}\right)\right)} \end{aligned} \tag{3.5}$$

Dans cette équation, $P\left(disp\left(p_1^{m,0}\right)\middle|R_1^{m,a_m,b_m}(t),R_2^{m,a_m,b_m}(t)\right)$ est déterminée par la fonction corrélation de phase unidimensionnelle qui est faite sur des régions segmentées, $P\left(R_1^{m,a_m,b_m}(t),R_2^{m,a_m,b_m}(t)\middle|disp\left(p_1^{m,0}\right)\right)$ est la mesure de consistance faite par la couche de type k de l'algorithme EPOC et $P\left(R_1^{m,a_m,b_m}(t),R_2^{m,a_m,b_m}(t)\right)$ est la probabilité que la segmentation faite dans R_1^{m,a_m,b_m} soit conforme à celle faite dans R_2^{m,a_m,b_m}. Grâce au bruit d'échantillonnage, illustré dans Figure 1.20, le pixel de référence qui est choisi dans la première image peut ne pas avoir la meilleure couleur de référence pour segmenter la deuxième image. Pour pallier à ce problème, on fait une marginalisation de $P\left(disp\left(p_1^{m,0}\right)\middle|R_1^{m,a_m,b_m}(t),R_2^{m,a_m,b_m}(t)\right)$ sur les régions qui se situent près des régions d'indice m. Cette marginalisation utilise l'hypothèse que les pixels voisins ont de bonnes chances d'appartenir à une même région.

$$P\left(disp\left(p_1^{m,0}\right)\right) = \frac{\sum_{k=m-v}^{m+v} P\left(disp\left(p_1^{m,0}\right)\middle|R_1^{k,a_k,b_k}(t),R_2^{k,a_k,b_k}(t),p_1^{k,0}\in obj\left(p_1^{m,0}\right)\right)P\left(R_1^{k,a_k,b_k}(t),R_2^{k,a_k,b_k}(t)\middle|p_1^{k,0}\in obj\left(p_1^{m,0}\right)\right)P\left(p_1^{k,0}\in obj\left(p_1^{m,0}\right)\right)}{P\left(R_1^{m,a_m,b_m}(t),R_2^{m,a_m,b_m}(t)\middle|disp\left(p_1^{m,0}\right)\right)} \quad (3.6)$$

En vertu de l'hypothèse qui dit que des pixels voisins pourraient appartenir à un même objet, $P(p_1^{k,0}\in obj(p_1^{m,0}))$, qui est la probabilité que le nouveau pixel de référence $p_1^{k,0}$ appartienne au même objet à lequel le pixel de reference originale $p_1^{m,0}$ appartient, vaut $1/K$ aux pixels voisins et 0 pour les pixels lointains, K une constante de normalisation tel que $\sum_{k=1}^{K} 1/K = 1$. $P(p_1^{k,0}\in obj(p_1^{m,0}))$ suit alors le modèle de Potts suivant:

$$P\left(p_1^{k,0}\in obj\left(p_1^{m,0}\right)\right) = \begin{cases}\frac{1}{K}\ si\ k\in[m-v,m+v] \\ 0 \qquad sinon\end{cases}, K=2v+1 \quad (3.7)$$

Ainsi, $P\left(p_1^{k,0}\in obj\left(p_1^{m,0}\right)\right)$ dépend seulement de la position relative du nouvel pixel de référence par rapport l'originale bien qu'on pouvait ajouter un terme qui est dépendant de la similarité de leurs couleurs. Cette marginalisation dans le numérateur est faite par la couche b de l'algorithme EPOC et se fait en sommant les espaces de recherche de 1DPOC voisins. Par simplification nous nous sommes contentés de faire cette marginalisation en 1D ($k\in[m-v,m+v]$) bien que avec EPOC, on la fait en 2D. La variable v défini l'étendu des pixels intervenants dans la couche b, et l'utilisation d'un modèle de Potts suggère que les pixels

d'indices v sont définis sur un segment (1D) ou rectangle (2D), bien que dans le chapitre 2 nous avons présentés des choix plus sophistiqués (Figure 2.3).

Le terme $P\left(R_1^{k,a_k,b_k}(t), R_2^{k,a_k,b_k}(t) | p_1^{k,0} \in obj(\, p_1^{m,0})\right)$ est maintenant moins sensible au bruit d'échantillonnage (après marginalisation) parce qu'on est en train de compter sur plusieurs pixels de référence au lieu d'un seul.

Remarquons que pour le moment nous n'avons pas changé le dénominateur. Le dénominateur $P\left(R_1^{m,a_m,b_m}(t), R_2^{m,a_m,b_m}(t) \middle| disp\left(p_1^{m,0}\right)\right)$ est pris en compte dans la couche k d'EPOC. Ce terme vise à vérifier que les deux régions peuvent être obtenues une de l'autre en utilisant le montant de translation $disp\left(p_1^{m,0}\right)$ pour évaluer la fiabilité de l'estimé $disp\left(p_1^{m,0}\right)$. La couche b, celle qui prend en compte de la marginalisation au numérateur de l'équation (3.6) est aussi susceptible à l'erreur, une autre marginalisation, au niveau de la couche k cette fois-ci, permet d'étendre l'étude de consistance $P\left(R_1^{m,a_m,b_m}(t), R_2^{m,a_m,b_m}(t) \middle| disp\left(p_1^{m,0}\right)\right)$ sur les pixels voisins aussi. La formulation finale de l'algorithme EPOC est donnée par :

$$P\left(disp\left(p_1^{m,0}\right)\right) =$$

$$\sum_{u=m-y}^{m+y}\left[\left[\sum_{k=u-v}^{u+v} P\left(disp\left(p_1^{u,0}\right) | R_1^{k,a_k,b_k}(t), R_2^{k,a_k,b_k}(t), p_1^{k,0} \in obj(\, p_1^{u,0}), p_1^{u,0} \in obj(\, p_1^{m,0})\right) P\left(R_1^{k,a_k,b_k}(t), R_2^{k,a_k,b_k}(t) | p_1^{k,0} \in obj(\, p_1^{u,0}), p_1^{u,0}\right.\right.\right. \tag{3.8}$$

$$\left.\left.\left. \in obj(\, p_1^{m,0})\right) P\left(p_1^{k,0} \in obj(\, p_1^{u,0}) | p_1^{u,0} \in obj(\, p_1^{m,0})\right) P(p_1^{u,0} \in obj(\, p_1^{m,0}))\right] / P\left(R_1^{u,a_u,b_u}(t), R_2^{u,a_u,b_u}(t) \middle| disp(p_1^{u,0})\right)\right]$$

Ici, y définit l'étendu des pixels de la couche k et $P(p_1^{u,0} \in obj(\, p_1^{m,0}))$ suit un modèle de Potts d'étendu $2y + 1$.

3.2 Approche graphique de l'algorithme EPOC

Dans cette section, on fait une description graphique de l'algorithme EPOC. Figure 3.1 illustre un graphe factorisé qui décrit l'architecture du système faisant intervenir les ensembles de nœuds de type *b* (couche 2) et ceux de type *k* (couche 3). Ici, on utilise le patron visualisé dans Figure 3.2 pour décrire l'organisation des nœuds de type b.

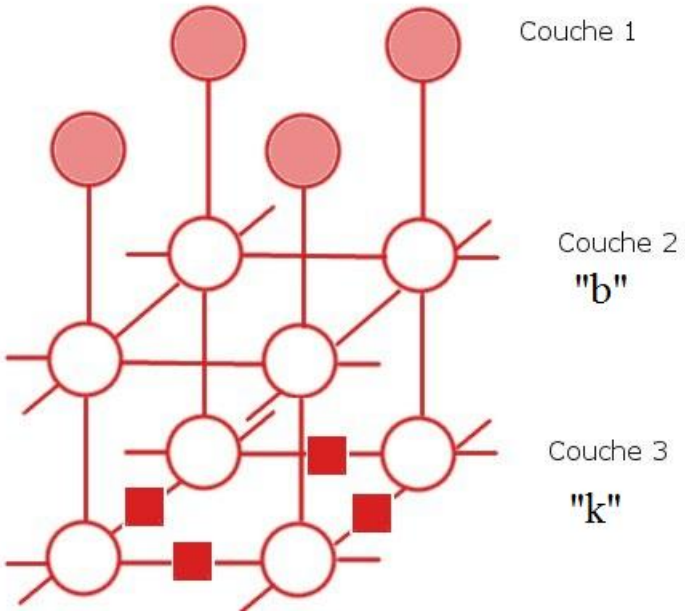


Figure 3.1-Architecture graphique de l'algorithme EPOC

Dans la partie précédente, nous avons expliqué que les nœuds de la couche b s'échangent des messages. La taille des messages est le nombre de disparités possibles à inférer. Cette étape ressemble à la propagation de conviction qui fait la propagation de messages qui contiennent aussi la probabilité d'avoir chacune des valeurs de disparités possibles. Une première différence est que la propagation de conviction est itérative. Une seule itération de BP consomme $O(nk^2)$ instructions dans une application de stereovision dense où n est le nombre totale de pixels et k le nombre de disparités possibles. Une seule itération de la couche b consomme $O(n.k.log(k))$ dans une application de stéréovision dense (la transformée de Fourier consomme $O(k.\log(k))$). Dans une application de stéréovision dispersée comme la notre où nous sommes intéressés à inférer la profondeur au niveau de seulement h pixels, $h < n$, une itération de BP consommera toujours $O(nk^2)$ instructions alors que la couche b consommera seulement $O(h.k.log(k))$. D'où l'avantage d'EPOC dans une application de stéréovision dispersée. Remarquons que les messages sont passés de la même façon qu'avec BP: en faisant sommer les messages des nœuds voisins. Ainsi, tout comme BP, les nœuds de cette couche sont représentés par un graphe non directionnel qui –dans le cas d'utilisation du patron illustré dans Figure 3.2 pour la couche b- relie chaque nœud à ces 4 plus proches voisins. Les entrées du système sont les images segmentées et sont représentées par des cercles pleins dans Figure 3.1. La couche k admet comme entrée les nœuds de la couche b. Au niveau de chaque nœud de type k, on ne retient qu'une seule disparité de la

couche b, celle qui maximise l'espace de recherche $\tilde{c}(n_2)$. De façon équivalente, on peut dire que les vecteurs d'état au niveau de la couche k sont d'une taille égale au nombre de disparités possible, mais la distribution de probabilité est nulle partout sauf à la valeur de disparité qui maximise l'espace de recherche $\tilde{c}(n_2)$ au niveau de la couche b. Chaque nœud de type k possède une valeur de disparité qui lui est passée d'un nœud de type b de façon exclusive. Ainsi, dans Figure 3.1, les nœuds de type k sont reliés aux nœuds de type b un à un. Le calcul de $P\left(R_1^{u,a_u,b_u}(t), R_2^{u,a_u,b_u}(t)\middle|disp(p_1^{u,0})\right)$ par les nœuds de type k se fait à travers des fonctions d'erreurs adaptées au problème, des fonctions qu'on appelle des fonctions d'énergie, d'où l'emploi des nœuds carrés (nœuds facteurs) qui expriment cette relation complexe entre les nœuds ronds de la couche k qui expriment les variables d'états. La détermination des fonctions d'énergie est inspirée de l'ICM (Iterated Conditional Modes) (Bishop 2006) et va être décrite en plus de profondeur sous peu.

Dans chacune des couches 2 et 3 du graphe de Figure 3.1, la distribution conjointe peut être écrite comme un produit de fonctions potentielles $\psi_C(\mathbf{x}_C)$ définies sur les cliques maximales du graphe :

$$p(x) = \frac{1}{Z}\prod_C \psi_C\,(\mathbf{x}_C) \tag{3.9}$$

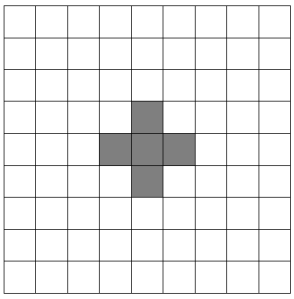


Figure 3.2-Patron sous forme de croix utilisé pour définir les ensembles de nœuds de type b (*b-clusters)*

Où la fonction de partition est donnée par:

$$Z = \sum_x \prod_C \psi_C\,(\mathbf{x}_C) \tag{3.10}$$

On assume que les fonctions potentielles $\psi_C(\mathbf{x}_C)$ sont strictement positifs, donc on les écrit sous la forme d'exponentielles négatives de fonctions d'énergie $E(\mathbf{x}_C)$:

$$\psi_C(\mathbf{x}_C) = exp\{-E(\mathbf{x}_C)\} \tag{3.11}$$

On aura donc:

$$p(x) = \frac{1}{Z}\prod_C exp\{-E(\mathbf{x}_C)\} \tag{3.12}$$

Maximiser la probabilité $p(x)$ revient à minimiser $E(\mathbf{x}_C)$. Nous allons montrer comment déterminer $E(\mathbf{x}_C)$ dans la description de la couche 3, mais tout d'abord décrivons les couches 1 et 2.

3.2.1 Couche 1

Les nœuds de cette couche sont des cercles pleins et représentent les conditions initiales du graphe. Les vecteurs d'état présents au niveau des nœuds de cette couche sont déterminés à partir de l'espace de recherche calculé par la fonction de corrélation de phase. Si on note l'espace de recherche par $c(k_2)$ et le vecteur d'état par f, ce dernier peut être déduit du premier par la relation suivante :

$$f = -c(k_2) + 1 \tag{3.13}$$

Cette nouvelle façon de définir le vecteur d'état f nous exige de faire une minimisation de la fonction d'énergie au niveau de la couche 2 plutôt que de la maximiser. Son but est de garder la consistance avec la convention de l'énergie positive dans (6.3). On rappelle que dans la description précédente de l'algorithme EPOC, on avait définit les vecteurs d'état comme étant les espaces de recherche, on avait alors à faire une maximisation de la fonction d'énergie.

3.2.2 Couche 2

Les nœuds de cette couche s'échangent des vecteurs d'états. À un nœud donné, le vecteur d'état est mis à jour en lui ajoutant les vecteurs d'états des nœuds voisins. Le message envoyé d'un nœud p à un nœud voisin q est décrit simplement par:

$$m_{pq}^1(f_q) = f_p \tag{3.14}$$

Les nœuds de la couche 2 du graphe de l'EPOC envoient leur information sous forme d'une impulsion bruitée aux nœuds voisins. La réponse finale - obtenue après accumulation - est donc résultante d'un mécanisme de vote. Si l'objet qu'on veut localiser est sous forme d'un plan parallèle aux cameras, la réponse idéale de chaque nœud sera la même et sera constituée d'un pic correspondant à la disparité reflétant la distance uniforme à cet objet. Sinon, les espaces de recherche $c^i(k_2)$ calculés aux nœuds voisins présenteront des pics répartis à gauche et à droite du pic détecté au nœud en question. Pour que la répartition de l'erreur ne favorise pas une valeur erronée, il est important que les patrons décrivant l'ensemble de nœuds b soit symétrique par rapport aux axes horizontale et verticale.

Avant de terminer cette partie, on signale que lorsque le montant de translation n'est pas une valeur entière, la réponse de la corrélation de phase n'est plus une impulsion de Dirac (Takita, Aoki et al. 2003). Supposons que $g(n_1, n_2)$ est obtenue de $f(n_1, n_2)$ en faisant une translation de $\delta_1$en direction de $n_1$et δ_2 en direction de n_2, i.e. :

$$g(n_1, n_2) = f(n_1 - \delta_1, n_2 - \delta_2) \tag{3.15}$$

La fonction de corrélation de phase bidimensionnelle s'exprime comme suit :

$$r(n_1, n_2) \cong \frac{\alpha}{N_1 N_2} \frac{\sin\{\pi(n_1 + \delta_1)\}}{\sin\left\{\frac{\pi}{N_1}(n_1 + \delta_1)\right\}} \frac{\sin\{\pi(n_2 + \delta_2)\}}{\sin\left\{\frac{\pi}{N_2}(n_2 + \delta_2)\right\}} \tag{3.16}$$

Où α est une constante dont la valeur maximale est l'unité.

Dans Figure 3.3, la courbe de la fonction de corrélation de phase est visualisée pour le cas $\alpha \neq 1$. On remarque que cette fonction comprend un maximum rencontré au point de coordonnées

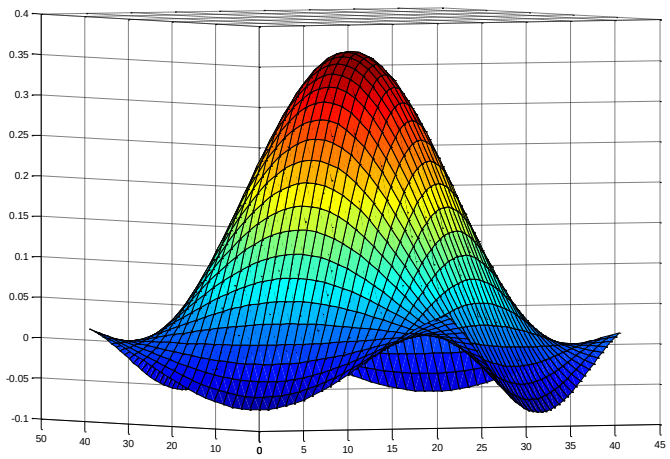


Figure 3.3-Visualisation de la fonction de corrélation de phase

(δ_1, δ_2) qui représentent la translation en pixels.

3.2.3 Couche 3

Dans cette couche, les fonctions potentielles $\psi_C\ (\mathbf{x}_C)$ représentées par des nœuds carrés dans le graphe de Figure 3.1 sont définies par :

$$\psi_C\ (\mathbf{x}_C) = \begin{cases} \mathbf{x}_C(i), si\ \mathbf{x}_C(i) = argmax(\mathbf{x}_C(i)) \\ 0, \quad sinon \end{cases} \tag{3.17}$$

La fonction d'énergie $E(x_C)$ peut être décomposée en une somme de plusieurs fonctions d'énergie qui sont multipliées par des poids respectifs. Si on note par y les nœuds contenant l'état du système à priori et par x les autres nœuds du graphe, de sorte que $\mathbf{x} = \{x, y\}$, on a:

$$E(x, y) = -\sum_i^M p_i E_i(x, y) \tag{3.18}$$

Ici, M est le nombre de fonctions d'énergie total. Les vecteurs d'état de $\mathbf{x}$ représentent des disparités. On choisit parmi la paire d'images stéréoscopiques une image de référence sur laquelle on définit le graphe. Avant de présenter les termes $E_i(x, y)$, on pose deux hypothèses essentielles à la construction de la fonction d'énergie :

-Hypothèse de surface lisse: Assume que la disparité des pixels voisins qui appartiennent à un même objet varie de façon lisse, selon une fonction continue partiellement monotone. Cette hypothèse aide à éliminer le bruit sur les disparités des pixels d'un objet si on sait que deux pixels appartiennent à un même objet.

-Hypothèse de l'uniformité de couleur: les couleurs qui constituent un objet sont fortement corrélées par régions, ce qui veut dire que s'il y a deux pixels qui ont des couleurs similaires, ils appartiendront au même objet.

Ainsi, on pose les fonctions d'énergie suivantes:

1-Fonction de mise à terre:

Cette fonction empêche le graphe de diverger de la solution initiale lorsque le nombre d'itérations est élevé, elle est donnée par :

$$E_1(\mathbf{x}) = \sum_i |x_i - y_j| \tag{3.19}$$

Où i représente l'indice de nœud.

2-Fonction de lissage:

Cette fonction d'énergie donne un coût élevé à la disparité évaluée si elle ne se situe pas dans un plan défini par ses voisins :

$$E_2(\mathbf{x}) = -\sum_{\{\mathrm{j,k,l,m}\}/\{p_j,p_k,p_l,p_m\}\in voi(p_i)} 4x_i - (x_j + x_k + x_l + x_m) \tag{3.20}$$

Ici, les indices de disparités $\{\mathrm{j,k,l,m}\}$ sont choisis de sorte que les pixels $\{p_j, p_k, p_l, p_m\}$ soient voisins du pixel p_i.

En fait, si on définit $j$comme étant le voisin à gauche de i, k celui de droite, l celui d'en haut et m celui d'en bas, on peut écrire $E_2(\mathbf{x})$ sous la forme :

$$E_2(\mathbf{x}) = \sum_{\{\mathrm{j,k,l,m}\}/\{p_j,p_k,p_l,p_m\}\in voi(p_i)} \{(x_j - x_i) - (x_i - x_k)\} + \{(x_l - x_i) - (x_i - x_m)\} \tag{3.21}$$

Soit

$$E_2(\mathbf{x}) = \sum_{\{\mathrm{j,k,l,m}\}/\{p_j,p_k,p_l,p_m\}\in voi(p_i)} \{pente_{gauche} - pente_{droite}\} + \{pente_{haut} - pente_{bas}\} \tag{3.22}$$

Donc cette fonction de coût favorise le cas où cinq pixels voisins $\{p_i, p_j, p_k, p_l, p_m\}$ appartiennent à un plan d'orientation quelconque.

3-Fonction de consistance :

Si une disparité d est attribuée à un certain pixel p, les couleurs des pixels correspondants doivent être similaires. Cette fonction attribue un coût proportionnel à la différence de couleur entre deux pixels correspondants dans les deux vues.

$$E_2(\mathbf{x}) = \sum_i \sum_{\mathrm{c}} |I_c^{gauche}(i - x_i) - I_c^{droite}(i)| \tag{3.23}$$

Où c désigne le canal (il existe un seul canal pour une image à fond de gris et trois canaux pour une image RGB). Ici, I_c^{gauche} et I_c^{droite} désignent l'intensité de couleur au canal c de l'image de gauche et de droite respectivement.

4-Fonction de priorité :

Vu que dans une scène qui contient des objets qui cachent d'autres, on a plus d'information sur les objets de l'avant plan que sur ceux de l'arrière plan à cause des régions en occlusion, on s'attend à ce que certains pixels reflétant des objets de l'arrière plan soient confondus à ceux de l'avant plan, surtout près des frontières d'objets. On pose donc une fonction d'énergie qui donne la priorité aux plus petites valeurs de disparités (qui sont réservés aux objets plus loin). Cette fonction d'énergie doit être active si la nouvelle valeur de disparité potentielle est différente de la disparité en cour d'un certain montant ξ. En prenant l'image de gauche comme référence :

$$E_4(\mathbf{x}) = \sum_{i,j} p_4 w(x_i, y_i) \tag{3.24}$$

$$w(x_i, y_i) = \begin{cases} |x_i| & x_i - y_i < \xi \\ 0 & autrement \end{cases} \tag{3.25}$$

Ce n'est pas nécessaire d'employer toutes les fonctions d'énergie en même temps. Certaines fonctions contribuent à l'aspect visuel de l'image de profondeur (lissage) alors que d'autres diminuent l'erreur d'estimation de disparités. Le choix final des fonctions d'énergies à employer dépend des besoins de l'application.

3.3 Comparaison de EPOC et ALBA par l'évidence

Dans cette section, on essaie de comparer le principal modèle retenu dans ce mémoire avec une autre de la même classe. On emprunte une approche Bayesienne pour faire cette comparaison. On dispose d'un ensemble de données d'entraînement D et d'un système $\mathcal{M}_i$ régit par un ensemble de paramètres w. Par entraînement, les paramètres w sont déterminés en fonction des données d'apprentissage. Lorsqu'on stimule le système entraîné, sa réponse est aussi dépendante des données d'entraînements que le modèle et la dimensionnalité du système le permettent. La régression polynomiale est un bon exemple de cette dépendance. En fait, la réponse optimale d'un système de régression polynomiale est obtenue en choisissant le bon nombre d'échantillons, la bonne dimensionnalité et le bon terme de régulation. À défaut de quoi, le polynôme généré

risque d'être sur-adapté ou sous-adapté aux données d'apprentissage. C'est le cas lorsqu'on essaie d'approximer des données par un polynôme d'ordre très différent de celui des données d'apprentissage, ou que les données d'entraînements ne sont pas suffisantes. La capacité d'un modèle à représenter les données d'entraînements est une propriété intéressante pour classifier ou valoriser un système donné et le comparer avec d'autres.

L'évidence est un outil qui permet de mesurer la capacité du modèle $\mathcal{M}_i$ à représenter les données d'entraînements. À la lumière de la discussion faite dans (Bishop 2006), on définit l'évidence par l'équation suivante

$$p(D|\mathcal{M}_i) = \int p(D|w, \mathcal{M}_i)p(w|\mathcal{M}_i)dw \tag{3.26}$$

C'est la probabilité de retrouver les données d'entraînements dans un modèle entrainé par ces données. On considère que les paramètres w à priori sont réparties selon une distribution uniforme $(w|\mathcal{M}_i) = \frac{1}{\Delta w_{priori}}$. Après entraînement, un modèle complexe doit conduire à une distribution à posteriori plus concentrée autour de la configuration des paramètres la plus probable w_{MAX} sur une plage $\Delta w_{posterieur}$ plus réduite par rapport aux modèles simples. En faisant des approximations, simplifications et le logarithme, on obtient pour un modèle ayant M paramètres:

$$\ln p(D) \approx \ln p(D|w_{MAX}) + M \ln(\frac{\Delta w_{posterieur}}{\Delta w_{priori}}) \tag{3.27}$$

Ici, le premier terme $\ln p(D|w_{MAP})$ représente la capacité du modèle dans sa configuration w_{MAX} la plus probable à représenter les données d'entraînements. Ce terme dépend implicitement de la dimensionnalité. Lorsque la complexité du modèle augmente, ce terme diminue. Le deuxième terme $M \ln(\frac{\Delta w_{posterieur}}{\Delta w_{priori}})$ diminue avec la complexité du système. Plus le modèle devient compliqué, $\Delta w_{posterieur}$ devient de plus en plus petit par rapport à Δw_{priori}. Le logarithme de la fraction $\frac{\Delta w_{posterieur}}{\Delta w_{priori}}$ devient de plus en plus négatif et le terme $M \ln(\frac{\Delta w_{posterieur}}{\Delta w_{priori}})$ devient pénalisant. Afin de maximiser l'évidence, il faut faire un compromis entre ces deux termes.

La version ALBA (Alba and Arce-Santana 2009) de la corrélation de phase est constituée tout d'abord d'un calcul de la fonction POC unidimensionnelle de deux signaux. Au lieu de chercher

la valeur de disparité qui maximise l'espace de recherche, on fait une détection de pics sur l'espace de recherche $\tilde{c}(k_2)$. On retient de cet espace de longueur 64 unités environ 16 pics. Le choix final de la valeur de disparité est obtenu en faisant une étude de consistance par conformité de couleur entre les deux vues avec les 16 candidats possibles. Par contraste, dans la couche *k* de l'EPOC, on dispose de 9 pics qui sont issues de 9 espaces de recherche différents avant de procéder avec un processus de minimisation d'énergie.

Les deux systèmes sont de même dimensionnalité ou de même nombre de paramètres qui est de 64. Le premier terme de l'évidence $\ln p(D|w_{MAP})$ est plus grand pour le système EPOC parce que le nombre de données d'entraînement est plus important. En fait, on dispose d'un vecteur d'entraînement par nœud de type *b*. Lorsqu'on dispose d'un nombre de vecteurs d'entraînement plus important, on limite le comportement aléatoire dans un système de grand ordre. Dans le cas de la régression polynomiale, on réduit l'effet de sur-adaptation en ajoutant des échantillons d'apprentissage aux endroits peu couverts par un ensemble de données d'entraînement. Dans le cas du EPOC, l'information supplémentaire apportée par chaque vecteur d'apprentissage provient de la référence utilisée dans l'extraction de caractéristiques. En fait, la détermination de la phase est sensible à l'amplitude du signal. Les données d'entraînement sont différentes dans le fait que la valeur de la phase qui y est extraite repose sur des régions différentes du signal. Dans le cas de ALBA, on dispose d'un seul vecteur d'entraînement.

Le deuxième terme $M \ln(\frac{\Delta w_{posterieur}}{\Delta w_{priori}})$ est moins pénalisant pour ALBA. En fait, lorsqu'on extrait 16 pics au lieu d'un seul, on est en train d'augmenter $\Delta w_{posterieur}$. Si on travaille avec 9 nœuds de type *k* dans EPOC, ce deuxième terme est plus pénalisant parce $\Delta w_{posterieur}$ vaut au plus 9. En fait, les pics détectés au niveau des 9 nœuds de type *k* ne sont pas fortement strictement exclusifs. Il est à noter qu'en fin de compte, les deux systèmes auront $w_{posterieur} = 1$ puisque seule une valeur de disparité est extraite, mais la discussion est faite sur la partie corrélation de phase des deux systèmes, excluant le processus de minimisation d'énergie et du test de consistance de couleur.

3.4 Discussion

Dans la comparaison entre ALBA et EPOC, nous avons vu que le point de force de EPOC vient du fait qu'il intègre un plus grand nombre de vecteurs d'entraînements. À noter que EPOC utilisent des images segmentées, et nous rappelons que la segmentation nuit à la précision de POC (35.4% d'erreur d'après Tableau *2.1*) bien que ça conduit à des images plus cohérentes aux frontières (Figure 1.21). Grâce à l'augmentation des vecteurs d'entraînement, la segmentation qui était un point de faiblesse de POC est devenue un point de force. En fait, la segmentation contribue à la diversification des vecteurs d'entraînements, on s'attend donc à ce que l'entropie du système au niveau de la couche b (i.e. le montant d'information contenue dans les vecteurs d'entraînements voisins) augmente. La disparité sélectionnée a atteint 17.1 % d'erreur, une amélioration très remarquable par rapport au système de précision 35.4% qui applique la segmentation avec POC, mais aussi une amélioration toujours énorme par rapport au système d'Alba qui a une précision de 27.6% tout en ajoutant aux images reconstruites une cohérence aux frontières qui est difficile à faire sans l'application de la segmentation. La stratégie employée par Alba est de ne pas faire de la segmentation pour ne pas avoir à traiter le bruit d'échantillonnage.

Par rapport à la propagation de conviction, L'algorithme EPOC ne comprend qu'une seule itération, et même cette itération ($O(n.k.log(k))$) est d'un ordre inférieur à celui de BP ($O(nk^2)$). Dans le calcule d'ordre, on n'a pas pris en compte de la présence de la racine carrée et de la division. Dans notre design sur FPGA, ces derniers ne contribuent pas significativement à la latence puisque celui qui prend le plus de temps ce sont les transformées de Fourrier. Les divisions et racines carrées sont faites en parallèle.

3.5 Conclusion

Ce chapitre était dédié à la présentation de l'algorithme EPOC en profondeur. Nous avons formulé de façon probabiliste EPOC en utilisant la relation de Bayes et autres astuces utilisées dans le calcul de probabilité tels que la marginalisation. La couche b correspond à une marginalisation du numérateur de (3.5) et la couche k correspond à une autre marginalisation plus globale. EPOC a été comparé au système d'Alba en utilisant l'évidence. Le point de force d'EPOC vient du fait qu'il prend en compte de plusieurs espaces de recherche.

Jusqu'à ce point, l'inférence de profondeur a été basée sur une démarche algorithmique, mais c'est intéressant de comprendre comment le cerveau pourrait inférer la profondeur. Cette compréhension nous permettrait de mettre POC dans un contexte cérébrale ce qui est favorable puisque EPOC est destiné à être utilisé dans un implant cérébrale dont le rôle est de remplacer la fonction du cerveau qui est l'inférence de profondeur. Nous ne visons pas de prédire l'architecture exacte du cerveau, une tâche laissée aux scientifiques de ce domaine, mais ce que nous tentons dans le chapitre suivant, c'est d'expliquer comment une architecture très abondante dans le cerveau possède la capacité d'inférer la profondeur.

CHAPITRE 4 CORRÉLATION DE PHASE ET MÉMOIRE ASSOCIATIVE

L'inférence de la profondeur de champ requiert de faire l'association entre la vue de gauche et celle de droite d'un système visuel binoculaire. C'est intéressant de pouvoir associer les deux vues en utilisant une mémoire associative. Notamment, parce que les mémoires associatives sont abondantes dans le cerveau humain. C'est l'exemple des Neurones miroirs (Oztop, Kawato et al. 2006). L'inférence de profondeur en utilisant une mémoire associative inciterait d'éventuels recherches futures sur le cerveau vers cette direction.

Dans ce chapitre, on explique comment le montant de translation entre deux signaux peut être inféré en utilisant une mémoire associative. Pour y arriver, on va utiliser la version quantique des réseaux de Hopfield en se basant sur la corrélation de phase. Ainsi, on aurait accordé à la corrélation de phase un rôle dans la construction d'un système mimétique.

4.1 Mémoire associative de Hopfield

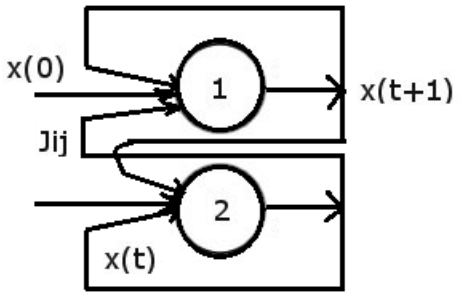


Figure 4.1-Réseau de Hopfield

Créées par John Hopfield, ces réseaux servent à adresser la mémoire par son contenu. Ils sont un exemple de mémoire associative. Ils garantissent la convergence vers un minimum local mais ne garantissent pas la convergence vers la bonne solution. Les poids du réseau sont contenus dans la matrice J, appelée le Hebbien, dont les éléments peuvent être déduits par :

$$J_{hj} = \frac{1}{N}\sum_{k=1}^{P} v_h^k v_j^k \tag{4.1}$$

Où N est la longueur des vecteurs d'entrée, P est le nombre de patrons de classes retenues par le réseau, v_h est un vecteur d'entrée. La version originale des réseaux de Hopfield (Figure 4.1)

comprend des vecteurs $\vec{x}$, dont les éléments prennent des valeurs dans $\{-1, +1\}$. Les itérations sont décrites par :

$$\vec{x}(t+1) = signe(\, J\vec{x}(t)\,) \tag{4.2}$$

La dérivation d'une version quantique de la mémoire de Hopfield est expliquée dans Annexe 2. La différence majeure par rapport à la mémoire associative de Hopfield originale, c'est que l'espace d'étude est l'espace complexe. Les vecteurs d'entraînements ne sont plus binaires, mais sont des nombres complexes. Un exemple de création de vecteurs d'entraînement complexes à partir de données réels est donnée par les équations suivantes:

$$\varphi_j^k = 2\pi \left(1 + exp\left(\frac{\bar{v}^k - v_j^k}{\sigma(v^k)} \right) \right)^{-1} \tag{4.3}$$

$$\psi_j^k \leftrightarrow e^{i\varphi_j^k} \tag{4.4}$$

Où $\bar{v}^k$ est la moyenne d'un vecteur d'entrée v^k et $\sigma(v^k)$ sa variance. Le vecteur d'entraînement est ψ_j^k, la phase est φ_j^k. Cet exemple est tiré d'une simulation d'une version quantique de la mémoire associative de Hopfield sur un ordinateur dans une application dédiée à la reconnaissance d'empreintes digitales (Perus, Bischof et al. 2004). Cet exemple montre qu'on peut travailler seulement avec la phase bien que la version quantique de la mémoire associative de Hopfield peut fonctionner avec un ψ_j^k d'amplitude non unitaire tel qu'expliqué dans Annexe 2.

Le Hebbien J_{hj} est donc remplacé par une matrice G qu'on appelle l'Hologramme:

$$G_{hj} = \sum_{k=1}^{P} \psi_h^k \left(\psi_j^k\right)^* = \sum_{k=1}^{P} e^{i(\varphi_h^k - i\varphi_j^k)} = \sum_{k=1}^{P} e^{i\varphi_h^k} e^{-i\varphi_j^k} \tag{4.5}$$

On peut déduire le résultat d'une itération à partir de :

$$\Psi_{\mathrm{h}}^{\mathrm{sortie}} = \sum_{\mathrm{j}=1}^{\mathrm{N}} G_{hj} \Psi_{\mathrm{h}}^{\mathrm{entrée}} \tag{4.6}$$

4.2 Architecture Sérielle

On veut calculer le montant de translation entre deux signaux x_1 et x_2. Dans l'exemple de Perus (Perus, Bischof et al. 2004) du réseau de Hopfield quantique, les vecteurs d'entrée sont transformés du plan réel au plan complexe par l'intermédiaire d'une assignation directe, élément par élément, en utilisant l'équation (4.3). Ici, on se propose de passer au domaine complexe en faisant la transformée de Fourier des vecteurs d'entrée bruts $\bar{v}^k$ et en normalisant:

$$\psi_j^k = \frac{FFT\{\bar{v}^k\}}{|FFT\{\bar{v}^k\}|} = e^{i\varphi_j^k} \tag{4.7}$$

Les vecteurs d'entrée bruts sont constitués des deux signaux x_1 et x_2 :

$$\bar{v}^k = x_k \tag{4.8}$$

Chaque vecteur d'entrée est de taille N. L'hologramme G est calculé en utilisant un seul vecteur d'entrée à la fois:

$$G_{hj} = \sum_{k=1}^{P} \psi_h^k \left(\psi_j^k\right)^* \Big|_{P=2} = \sum_{k=1}^{2} e^{i\varphi_h^k} e^{-i\varphi_j^k} = \sum_{k=1}^{2} e^{i(\varphi_h^k - \varphi_j^k)} \tag{4.9}$$

Si on applique une entrée unitaire au système entraîné, tel que :

$$\Psi_{\mathrm{h}}^{\mathrm{entrée}} = 1 = e^{i0} \tag{4.10}$$

L'information sur la différence de phase entre x_1 et x_2 sera contenue dans le spectre de sortie

$$\Psi_{\mathrm{h}}^{\mathrm{sortie}} = \sum_{\mathrm{j}=1}^{\mathrm{N}} G_{hj} \Psi_{\mathrm{h}}^{\mathrm{entrée}} = \sum_{\mathrm{j}=1}^{\mathrm{N}} G_{hj} \tag{4.11}$$

Le montant de translation entre x_1 et x_2 peut être déduit en faisant tout d'abord la transformée de Fourier inverse et ensuite une recherche de l'argument qui maximise l'espace de recherche:

$$c = IFFT\left\{\Psi_{\mathrm{h}}^{\mathrm{sortie}}\right\} \tag{4.12}$$

$$disp = argmax\{c\} + cst \tag{4.13}$$

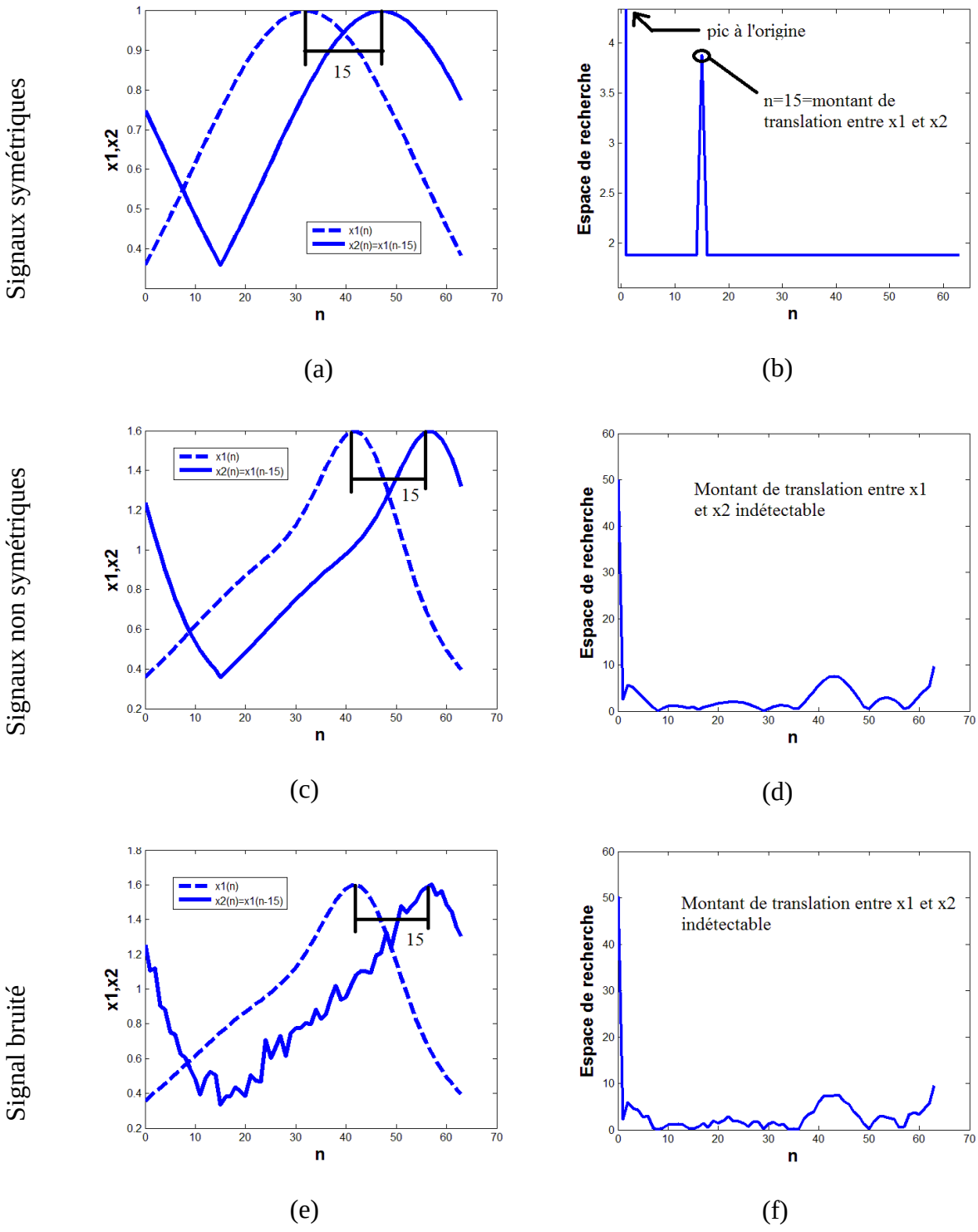


Figure 4.2 Expérimentation avec l'architecture sérielle de Hopfield. Dans les graphiques (a) et (b), on calcule le montant de translation entre deux signaux symétriques. Dans les graphiques (c) et (d), on utilise des signaux non symétriques. Dans (e) et (f), on calcul le montant de translation entre un signal et son instance translatée qui est bruitée.

Où cst est une constante qui dépend de l'ordre d'apparition des fréquences dans le spectre. Figure 4.2 montre un exemple. Dans cette figure, la première colonne illustre des paires de signaux, chaque paire de signaux est composée d'un signal et de son instance translatée. La deuxième colonne illustre la sortie de l'architecture sérielle proposée et qui permet d'inférer le montant de translation entre les signaux de chaque paire de signaux de la première colonne.

4.3 Architecture Parallèle

Dans l'architecture parallèle, le passage au domaine complexe se fait en concaténant deux vecteurs qui correspondent aux spectres normalisés des signaux $x_1$et x_2

$$\psi_j = \left[\frac{FFT\{\bar{v}^1\}}{|FFT\{\bar{v}^1\}|}, \frac{FFT\{\bar{v}^2\}}{|FFT\{\bar{v}^2\}|}\right] = \left[e^{i\varphi_j^1}, e^{i\varphi_j^2}\right] = e^{i\varphi_j}\ , \bar{v}^k = x_k \tag{4.14}$$

L'hologramme G devient dans ce cas une matrice dont les dimensions sont deux fois plus grandes que dans le cas de l'architecture sérielle. Toutefois, l'entraînement du réseau se fait en une seule itération.

$$G_{hj} = \sum_{k=1}^{P} \psi_h^k \left(\psi_j^k\right)^* \Big|_{P=1} = e^{i\varphi_h}\, e^{-i\varphi_j} \tag{4.15}$$

On stimule le système avec un vecteur $\Psi_{\mathrm{h}}^{\text{entrée}}$. Celui-ci est un signal binaire défini par:

$$\Psi_{\mathrm{h}}^{\text{entrée}} = \begin{cases} 0, \text{si h} \lesseqgtr \mathrm{N} \\ 1, \text{autrement} \end{cases} \tag{4.16}$$

Le choix de la forme du signal stimulant le réseau détermine si on fait la mesure de la translation par rapport au signal de gauche ou celui de droite. Le résultat $\Psi_{\mathrm{h}}^{\text{sortie}}$ sera nul pour les valeurs de h qui annulent $\Psi_{\mathrm{h}}^{\text{entrée}}$. Pour les valeurs de h où $\Psi_{\mathrm{h}}^{\text{entrée}}$ est unitaire, les valeurs de $\Psi_{\mathrm{h}}^{\text{sortie}}$ constituent les composantes spectrales de l'espace de recherche utile à la détermination du montant de translation. Ce dernier est déterminé en faisant la transformée inverse de la sortie du réseau et en faisant une recherche de la valeur maximale sur l'espace de recherche comme dans le cas de l'architecture sérielle (équations (4.12) et (4.13)). Les résultats obtenus avec cette architecture sont illustrés dans Figure 4.3.

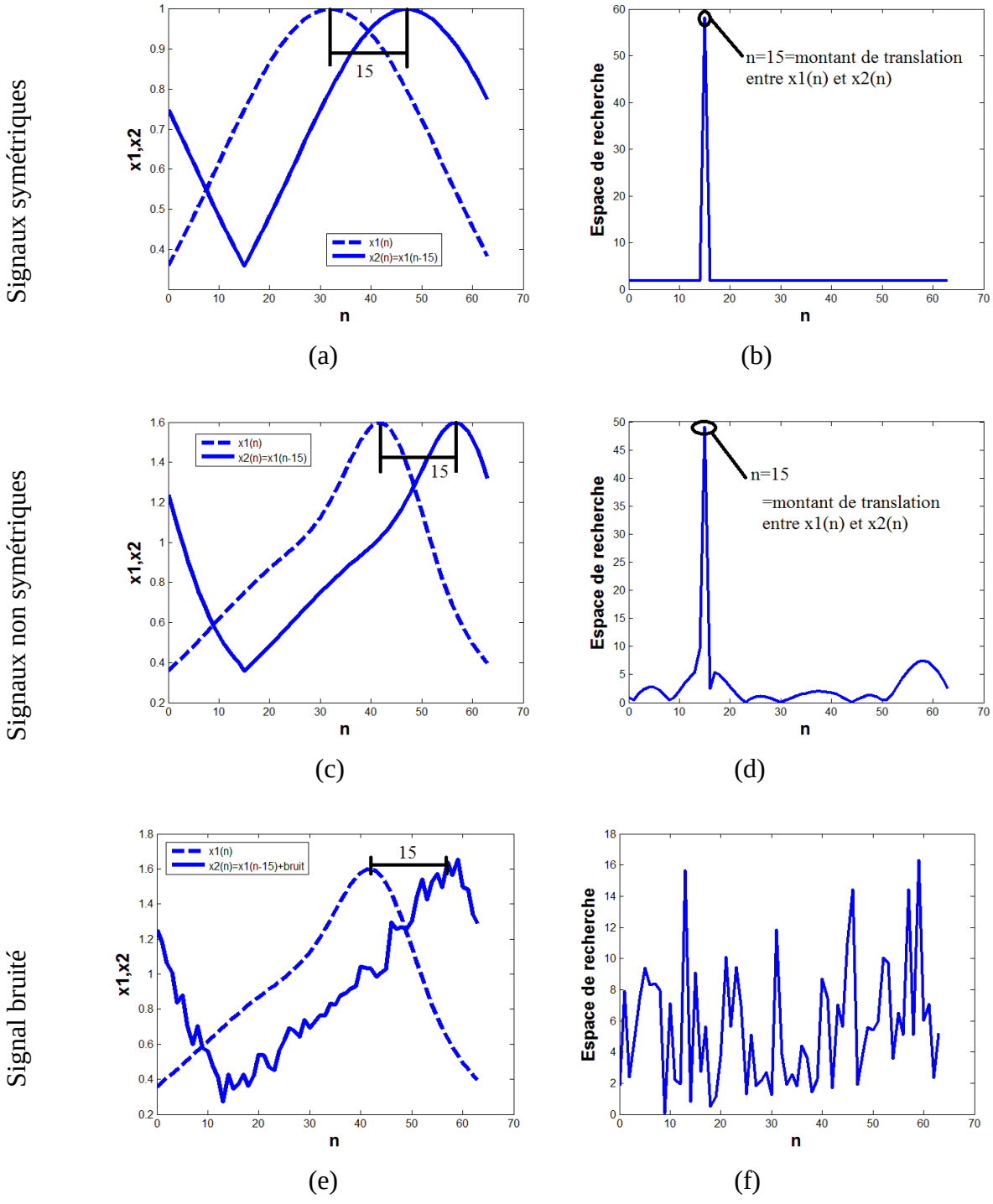


Figure 4.3 Expérimentation avec l'architecture parallèle de Hopfield. Dans les graphiques (a) et (b), on calcule le montant de translation entre deux signaux symétriques. Dans (c) et (d), on utilise des signaux non symétriques. Dans (e) et (f), on calcul le montant de translation entre un signal et son instance translatée qui est bruitée.

4.4 Explication

Considérons les signaux x_1 et x_2, tel que:

$$x_2 = x_1(n-d) \tag{4.17}$$

On note leurs transformées de Fourier par X_1 et X_2, ainsi:

$$X_i(f) = A_i(f)e^{j\theta_f^i} \tag{4.18}$$

$$X_2(f) = X_1(f)e^{-j2\pi fd} \tag{4.19}$$

On définit les matrices suivantes:

$$\check{G}_{11}(k,l) = \frac{X_1(k)}{|X_1(k)|}\frac{X_1^*(l)}{|X_1(l)|} = e^{j\theta_k^1}e^{-j\theta_l^1} \tag{4.20}$$

$$\check{G}_{12}(k,l) = \frac{X_1(k)}{|X_1(k)|}\frac{X_2^*(l)}{|X_2(l)|} = e^{j\theta_k^1}e^{-j\theta_l^2} \tag{4.21}$$

$$\check{G}_{21}(k,l) = \frac{X_2(k)}{|X_2(k)|}\frac{X_1^*(l)}{|X_1(l)|} = e^{j\theta_k^2}e^{-j\theta_l^1} \tag{4.22}$$

$$\check{G}_{22}(k,l) = \frac{X_2(k)}{|X_2(k)|}\frac{X_2^*(l)}{|X_2(l)|} = e^{j\theta_k^2}e^{-j\theta_l^2} \tag{4.23}$$

Dans l'architecture sérielle, l'hologramme peut être écrit sous la forme :

$$G = \check{G}_{11} + \check{G}_{22} \tag{4.24}$$

$$G(k,l) = e^{j\theta_k^1}e^{-j\theta_l^1} + e^{j\theta_k^2}e^{-j\theta_l^2} \tag{4.25}$$

$$G(k,l) = e^{j\theta_k^1}e^{-j\theta_l^1} + e^{j\theta_k^1}e^{-j\theta_l^1}e^{-j2\pi kd/N}e^{-j2\pi ld/N} \tag{4.26}$$

Si x_1 est symétrique par rapport à l'axe d'origine de sorte que $x_1(n) = x_1(-n)$, ses composantes spectrales sont réelles, ainsi :

$$G(k,l) = 1 + e^{-j2\pi kd/N}e^{-j2\pi ld/N} \tag{4.27}$$

$$G(k,l) = 1 + e^{-j2\pi(k+l)d/N} \tag{4.28}$$

Ainsi, la sortie est donnée par :

$$\Psi_{\mathrm{k}}^{\mathrm{sortie}} = \sum_{\mathrm{l}=1}^{\mathrm{N}} 1 + e^{-j2\pi(k+l)d/N} \tag{4.29}$$

$$\Psi_{\mathrm{k}}^{\mathrm{sortie}} = N + \left(\sum_{\mathrm{l}=1}^{\mathrm{N}} e^{-j2\pi(l)d/N} \right) e^{-j2\pi kd/N} \tag{4.30}$$

$$\Psi_{\mathrm{k}}^{\mathrm{sortie}} = N + \left(2 \sum_{\mathrm{l}=1}^{\mathrm{N}/2} \cos\left(-2\pi dl/N\right) \right) e^{-j2\pi dk/N} \tag{4.31}$$

On obtient,

$$\Psi_{\mathrm{k}}^{\mathrm{sortie}} = N + \ e^{-j2\pi kd/N} \tag{4.32}$$

La transformée de Fourier inverse d'une constante est une impulsion de Dirac à l'origine. On prévoit donc observer un pic à l'origine de l'espace de recherche. La transformée de Fourier inverse d'une exponentielle complexe est un pic translaté d'un montant qui correspond à sa phase. Dans ce cas, $e^{-j2\pi kd/N}$ correspond à l'information pertinente à la détermination du montant de translation recherché.

Dans l'architecture parallèle, l'hologramme peut être écrit comme suit:

$$G = \begin{bmatrix} \check{G}_{11} & \check{G}_{12} \\ \check{G}_{21} & \check{G}_{22} \end{bmatrix} \tag{4.33}$$

Si on construit le vecteur d'entrée $\Psi_{\mathrm{h}}^{\mathrm{entrée}}$ de sorte que:

$$\Psi_{\mathrm{h}}^{\mathrm{entrée}} = \begin{cases} 0, \text{si h} \leq \mathrm{N}/2 \\ 1, \text{autrement} \end{cases} \tag{4.34}$$

Le vecteur de sortie sera calculé selon:

$$\Psi_{\mathrm{k}}^{\mathrm{sortie}} = \sum_{\mathrm{l}=1}^{2\mathrm{N}} G_{kl} \Psi_{\mathrm{k}}^{\mathrm{entrée}} = \begin{cases} \Psi_{\mathrm{k}}^{\mathrm{sortie1}} = 0 & , si \ \mathrm{h} \leq \mathrm{N}/2 \\ \Psi_{\mathrm{k}}^{\mathrm{sortie2}} = \sum_{\mathrm{j}=1}^{\mathrm{N}} \left[\check{G}_{21} + \check{G}_{22} \right]_{kl}, si \ \mathrm{h} > \mathrm{N}/2 \end{cases} \tag{4.35}$$

$$\Psi_{\mathrm{k}}^{\mathrm{sortie2}} = \sum_{\mathrm{l}=1}^{\mathrm{N}} e^{j\theta_k^2} e^{-j\theta_l^1} + e^{j\theta_k^2} e^{-j\theta_l^2} \tag{4.36}$$

$$\Psi_{\mathrm{k}}^{\mathrm{sortie2}} = \sum_{\mathrm{l}=1}^{\mathrm{N}} e^{-j2\pi dk/N} e^{j\theta_k^1} e^{-j\theta_l^1} + e^{-j2\pi dk/N} e^{j\theta_k^1} e^{-j\theta_l^1} e^{j2\pi dl/N} \tag{4.37}$$

$$\Psi_{\mathrm{k}}^{\mathrm{sortie2}} = e^{j\theta_k^1} e^{-j2\pi dk/N} \sum_{\mathrm{l}=1}^{\mathrm{N}} e^{-j\theta_l^1} \left(1 + e^{j2\pi dl/N}\right) \tag{4.38}$$

$$\Psi_{\mathrm{k}}^{\mathrm{sortie2}} = e^{j\theta_k^1} e^{-j2\pi dk/N} \chi \tag{4.39}$$

Où χ est un nombre complexe. On voit que la phase du vecteur de sortie de l'architecture parallèle est dépendante à un certain niveau de la phase du signal originale.

Si le signal x_1 est symetrique pair, on a :

$$\Psi_{\mathrm{k}}^{\mathrm{sortie2}} = \mathrm{cst} \times e^{-j2\pi dk/N} \tag{4.40}$$

Contrairement au cas de l'architecture sérielle traité, on peut observer une impulsion qui correspond à la translation entre les signaux d'entrée directement, sans impulsion supplémentaire à l'origine.

4.5 Discussion

Les neurones miroirs impliqués dans l'apprentissage (Arbib 2002) sont destinés à être entraînés sur de longues périodes de temps et ne sont pas des modèles qui peuvent être employés pour expliquer la manière par laquelle le cerveau associe deux images qui sont issues des yeux pour inférer la profondeur en temps réel.

Les réseaux de Hopfield sont une forme de mémoire associative présente dans la littérature. Ils pourraient représenter des architectures de neurones qui ont été évolué à partir des neurones miroirs, ou de neurones plus primaires, pour jouer le rôle de mémoire associative au niveau du cortex visuel.

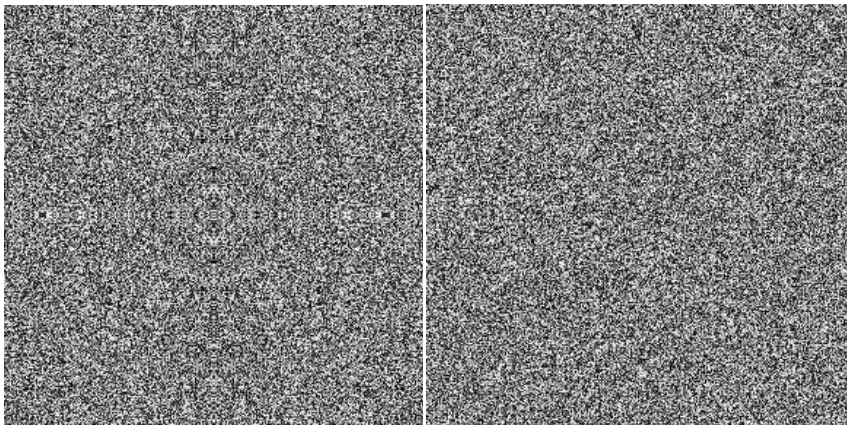
Inspirés de la corrélation de phase, on crée deux architectures basées sur un réseau de Hopfield quantique pour inférer la profondeur de champs. La première architecture est sérielle. Selon celle-ci, le réseau est entraîné itérativement. On compte deux itérations pour calculer l'hologramme du réseau: une itération par portion d'image issue d'un œil. On a trouvé que cette architecture permet de savoir le montant de translation entre deux signaux, étape nécessaire pour inférer la profondeur, avec une information supplémentaire mais prévisible qui doit être éliminée.

La deuxième architecture est parallèle. Elle nécessite plus de neurones que l'architecture sérielle mais requiert une seule itération. Les deux vues qui sont issues des yeux sont alors traitées en même temps et le réseau obtenu est plus rapide. En plus, sous des conditions optimales, le réseau semble être capable de fournir l'information sur le montant de translation entre deux signaux sans

supplément d'information trompeuse. Vu le nombre de neurones présents dans le cerveau qui est estimé à plus de 100 milliards, c'est convenable de donner la préférence à l'architecture parallèle.

Toutefois, pour avoir les meilleurs résultats, les deux architectures explorées semblent donner leur réponse optimale si les signaux d'entrée sont des instances translatées d'un signal symétrique. La question qui se pose c'est : Est-ce que la symétrie des signaux d'entrée constitue un caractère restrictif qui empêche l'explication de l'inférence de profondeur par la mémoire associative proposée?

D'après des expérimentations psychologiques, le cerveau humain exhibe une réponse plus marquée aux patrons symétriques que les patrons non symétriques (Norcia, Candy et al. 2002). Plus encore, lorsqu'on a présenté à des individus une image non symétrique (Figure 4.4), on a remarqué que les signaux détectés au niveau des régions cibles du cerveau humain sont multiphasés en fonction du patron. Ceci entre en corrélation avec nos observations surtout dans l'architecture parallèle du réseau de Hopfield quantique. En fait, dans ce dernier, lorsque les signaux d'entrée ne sont pas symétriques, la réponse devient entachée d'un déphasage supplémentaire, et le montant de déphasage est directement dépendant du signal d'entrée du système. Dans le cas du cerveau, ce déphasage néfaste à l'assimilation d'une scène non symétrique peut être constitué de la phase du spectre de l'image observée.



(a) (b)

Figure 4.4 Exemple de patron symétrique (a) et de patron non symétrique (b). Il fut montré que des parties bien déterminées du cerveau humain sont activées lorsqu'on leur présente un patron symétrique. Les patrons non symétriques engendrent le déclenchement de plusieurs parties du cerveau de façon non synchronisée (Norcia, Candy et al. 2002).

Dans la partie V1 du cortex visuel, il existe des neurones qui sont activés lorsque l'image contient des segments dirigés vers une direction donnée. Ces neurones jouent un rôle dans la détection de lignes parallèles. La détection des lignes parallèles est une des lois de gestalt. Ainsi, la détection de la symétrie par le cerveau, en détectant les lignes parallèles, peut être une opération préalable à l'inférence de profondeur par une mémoire associative similaire à celle qu'on a présenté.

Enfin, notons que si on multiplie (élément par élément) l'hologramme dans l'architecture parallèle (5.27) par la matrice suivante :

$$T = \begin{bmatrix} M0_N & M0_N \\ ID_N & M0_N \end{bmatrix} \tag{4.41}$$

Où $M0_N$ et ID_N sont respectivement la matrice nulle et la matrice identité de dimensions $N \times N$, le système parallèle sera équivalent à la corrélation de phase. Le système ne sera plus dépendant de la symétrie des signaux d'entrée. Toutefois, l'hologramme de l'architecture parallèle originale, permet de conserver les propriétés d'une mémoire associative.

4.6 Conclusion

Dans ce chapitre, nous avons exploré deux architectures basées sur un réseau de Hopfield quantique. La première architecture, l'architecture sérielle nécessite un entraînement en deux itérations. La deuxième, l'architecture parallèle, requiert une seule itération mais est plus dispendieuse en nombre de nœuds. Les deux architectures peuvent fournir le montant de translation entre un signal symétrique et son instance translaté. Dans le cas de signaux non symétriques, l'architecture parallèle semble capable d'inférer le montant de translation mais avec un bruit qui dépend de la phase des signaux d'entrée, alors que l'architecture sérielle est incapable. On fait une comparaison entre nos résultats et ceux obtenus par des expérimentations psychologiques, et on trouve des points communs. D'après Norcia (Norcia, Candy et al. 2002), le cerveau exhibe une réponse déterminée lorsqu'on expose à un individu un patron symétrique alors que cette réponse devient multiphasée lorsqu'on lui expose un patron non symétrique. Ceci entre en corrélation avec ce qu'on observe avec les architectures sérielle et parallèle qu'on a développé. On en déduit que l'inférence de profondeur en utilisant des neurones dont la configuration est telle que tous les neurones sont reliés ensemble est possible pour le cerveau.

CONCLUSION

Dans ce mémoire, nous nous sommes intéressés à la construction d'un système d'imagerie 3D. On a choisi de développer une méthode stéréoscopique dispersée. Contrairement aux systèmes de stéréovision dense, la stéréoscopie dispersée ne nécessite pas le calcul de disparités sur toute l'image, mais seulement au voisinage des pixels d'intérêt. On a fait ce choix parce que le nombre de phosphènes à stimuler peut varier de quelques centaines à quelques milliers. Parmi les systèmes de stéréoscopie dispersée, il y a les méthodes de corrélation. En particulier, la méthode de corrélation de phase était intéressante surtout parce que son application dans des environnements qui ne contiennent pas d'occlusions a été encourageante.

La contribution majeure de ce mémoire est l'algorithme EPOC. Dans ce dernier, la corrélation de phase est appliquée sur des pixels qui se trouvent au voisinage du pixel de référence au niveau duquel on veut calculer la disparité. Les espaces de recherche des corrélations de phase sont ensuite propagés pour corriger les espaces de recherche voisins. La propagation de messages entre nœuds est un aspect tiré des graphes Markoviens. Ceci dit, EPOC doit hériter certaines faiblesses de ces méthodes. Ces faiblesses sont constituées du fait qu'un espace de recherche doit être calculé au niveau de chaque pixel appartenant au voisinage du pixel d'intérêt et non pas seulement au niveau du phosphène à stimuler. Toutefois, le fait qu'on utilise une seule itération préserve la propriété de localité héritée de la corrélation de phase. Par exemple, quel que soit le nombre de corrélations de phase à utiliser pour raffiner la disparité, la résolution peut être toujours de 63-64 disparités. À noter qu'en stéréoscopie dense, si on veut appliquer EPOC sur deux images de N pixels chacun, il faut faire exactement N fois la corrélation de phase au total. Avec une implémentation sur FPGA, ceci suggère que le temps d'exécution peut être similaire au système POC simple avec segmentation de couleur dont l'erreur moyenne est d'environ 35%.

Une autre contribution est l'algorithme MMPOC. Cet algorithme montre que la corrélation de phase bidimensionnelle peut être optimisée algorithmiquement au coût d'une augmentation de l'erreur moyenne d'estimation de disparité.

Dans un autre développement, on tire profit de la phase pour calculer le montant de translation entre deux signaux en utilisant une version quantique de la mémoire associative de Hopfield. On construit deux architectures qui expliquent comment une mémoire associative peut inférer la

profondeur. On remarque que les points de faiblesse des mémoires associatives dans cette tâche ressemblent à celles du cerveau humain.

BIBLIOGRAPHIE

Alba, A. and E. Arce-Santana (2009). Phase-Correlation Guided Search for Realtime Stereo Vision. Combinatorial Image Analysis. 13th International Workshop, IWCIA 2009, 24-27 Nov. 2009, Berlin, Germany, Springer Verlag.

Arbib, M. (2002). The Mirror system, imitation, and the evolution of language. C. Nehaniv, & K. Dautenhahn (Eds), Imitation in animals and artifacts. MA, MIT Press**:** 229-280.

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

Bishop, C. (2006). Linear Models for regression. Pattern recognition and machine learning. New York, springer.

Bishop, C. M. (2006). Graphical Models. Pattern recognition and machine learning. NY, Springer**:** 362-422.

Brindley, G. S. and W. S. Lewin (1968). The sensations Produced by electrical stimulation of the Visual Cortex. Journal of Physiology. **196:** 479-493.

Buffoni, L. X., J. Coulombe, et al. (2003). An image processing system dedicated to cortical visual stimulators. CCECE 2003 Canadian Conference on Electrical and Computer Engineering: Toward a Caring and Humane Technology, May 4, 2003 - May 7, 2003, Montreal, Canada, Institute of Electrical and Electronics Engineers Inc.

Buffoni, L. X., J.Coulombe, et al. (2003). An image processing system dedicated to cortical visual stimulators, Montréal.

Cappé, O., E. Moulines, et al. (2005). Inference in Hidden Markov Models. New York, Springer-Verlag.

Delia, D. A. (2007). Reconstruction 3D de scènes dynamiques d'un capteur d'images dédié à un stimulateur visuel intracortical. Génie Électrique. Montréal, École Polytechnique de Montréal. **Maîtrise ès sciences appliquées** 176.

El-Etriby, S., A. K. Al-Hamadi, et al. (2007). Dense stereo correspondence with slanted surface using phase-based algorithm. 2007 IEEE International Symposium on Industrial Electronics, ISIE 2007, June 4, 2007 - June 7, 2007, Caixanova - Vigo, Spain, Institute of Electrical and Electronics Engineers Inc.

Felzenszwalb, P. F. and D. P. Huttenlocher (2006). "Efficient belief propagation for early vision." International Journal of Computer Vision **70**(Compendex): 41-54.

Feynman, R. P. and A. R. Hibbs (1965). Quantum Mechanics and Path Integrals. New York, McGraw-Hill.

Gervais, J.-F. (2004). Échange bidirectionnel de données avec un implant électronique alimenté par lien inductif,214 p, Mémoire de maitrise, École Polytechnique de Montréal.

Hawi, F. and M. Sawan (2011). "Phase-Based Passive stereovision systems dedicated to cortical visual stimualtors." Submitted to CVIU, July 2011.

Hirschmuller, H., P. R. Innocent, et al. (2002). Real-time correlation based stereo vision with reduced border error. International Journal of Computer Vision**:** 47(41-43): 229-246.

Kim, W., J. Park, et al. (2009). Stereo matching using population-based MCMC. International Journal of Computer Vision**:** 195-209.

Kuglin, C. D. and D. C. Hines (1975). The phase correlation image alignment method. Proceedings of the 1975 International Conference on Cybernetics and Society, 23-25 Sept. 1975, New York, NY, USA, IEEE.

Morikawa, M., A. Katsumata, et al. (1999). An Image processor Implementing Algorithms using Characteristics of Phase Spectrum of Two-Dimensional Fourier Transformation. IEEE international symposium on industrial electronics.

Nalpantidis, L. and A. Gasteratos (2010). "Biologically and psychophysically inspired adaptive support weights algorithm for stereo correspondence." Robotics and Autonomous Systems **58**(Compendex): 457-464.

Niitsuma, H. and T. Maruyama (2010). Sum of absolute difference implementations for image processing on FPGAs. 20th International Conference on Field Programmable Logic and Applications, FPL 2010, August 31, 2010 - September 2, 2010, Milano, Italy, IEEE Computer Society.

Norcia, A. M., T. R. Candy, et al. (2002). "Temporal dynamics of the human response to symmetry." Journal of Vision **2**(2).

Oztop, E., M. Kawato, et al. (2006). "Mirror neurons and imitation: A computationally guided review." Neural Networks **19**(Copyright 2006, The Institution of Engineering and Technology): 254-271.

Perus, M., H. Bischof, et al. (2004). "Quantum-implementable selective reconstruction of high-resolution images." Applied Optics **43**(Compendex): 6134-6138.

Perus, M. and S. K. Dey (2000). "Quantum systems can realize content-addressable associative memory." Applied Mathematics Letters **13**(Compendex): 31-36.

ROY, M. (1999). Conception et réalisation d'un prototype de la partie implantable d'un stimulateur visuel cortical, 216p, Mémoire de maitrise, École Polytechnique de Montréal.

Scharstein, D. and C. Pal (2007). Learning conditional random field for stereo. IEEE Conference on Computer Vision and Pattern Recognition.

Scharstein, D. and C. Pal (2007). Learning conditional random fields for stereo. 2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR'07, June 17, 2007 - June 22, 2007, Minneapolis, MN, United states, Inst. of Elec. and Elec. Eng. Computer Society.

Scharstein, D. and R. Szeliski. (2001). "Middlebury stereo vision page." from http://vision.middlebury.edu/stereo.

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

Schneider, G. E. (1967). "Contrasting visuomotor functions of tectum and cortex in the golden hamster." Psychologische Forschung **31**: 52-62.

Takita, K., T. Aoki, et al. (2003). "High-accuracy subpixel image registration based on phase-only correlation." IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **E86-A**(Copyright 2004, IEE): 1925-1934.

Trugenberger, C. A. (2002). "Quantum optimization for combinatorial searches." New Journal of Physics **4**(Copyright 2003, IEE).

Ungerleider, L. G. and M. Mishkin (1982). Two cortical visual systems. Analysis of Visual Behavior. Cambridge, MA, MIT press**:** 549–586.

Viterbi, A. (1967). "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm." Information Theory, IEEE Transactions on **13**(2): 260-269.

Wang, L., M. Liao, et al. (2007). High-quality real-time stereo using adaptive cost aggregation and dynamic programming. 3rd International Symposium on 3D Data Processing, Visualization, and Transmission, 3DPVT 2006, June 14, 2006 - June 16, 2006, Chapel Hill, NC, United states, Inst. of Elec. and Elec. Eng. Computer Society.

Weinman, J. J., L. Tran, et al. (2008). Efficiently learning random fields for stereo vision with sparse message passing. Computer Vision. 10th European Conference on Computer Vision, ECCV 2008, 12-18 Oct. 2008, Berlin, Germany, Springer-Verlag.

Yoon, K.-J. and I.-S. Kweon (2005). Locally adaptive support-weight approach for visual correspondence search. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005, June 20, 2005 - June 25, 2005, San Diego, CA, United states, Institute of Electrical and Electronics Engineers Computer Society.

ANNEXE1- Implémentation du système EPOC

Cette partie est dédiée à la description détaillée de l'implémentation du système EPOC présenté dans chapitre 4. L'implémentation est faite à l'aide de System Generator de Xilinx. EPOC est constitué de trois phases essentielles. La première phase est constituée du bloc de lecture de données et de segmentation, la deuxième est constituée de deux blocs de corrélation de phase et la troisième est le bloc de superposition des espaces de recherche et de minimisation d'énergie.

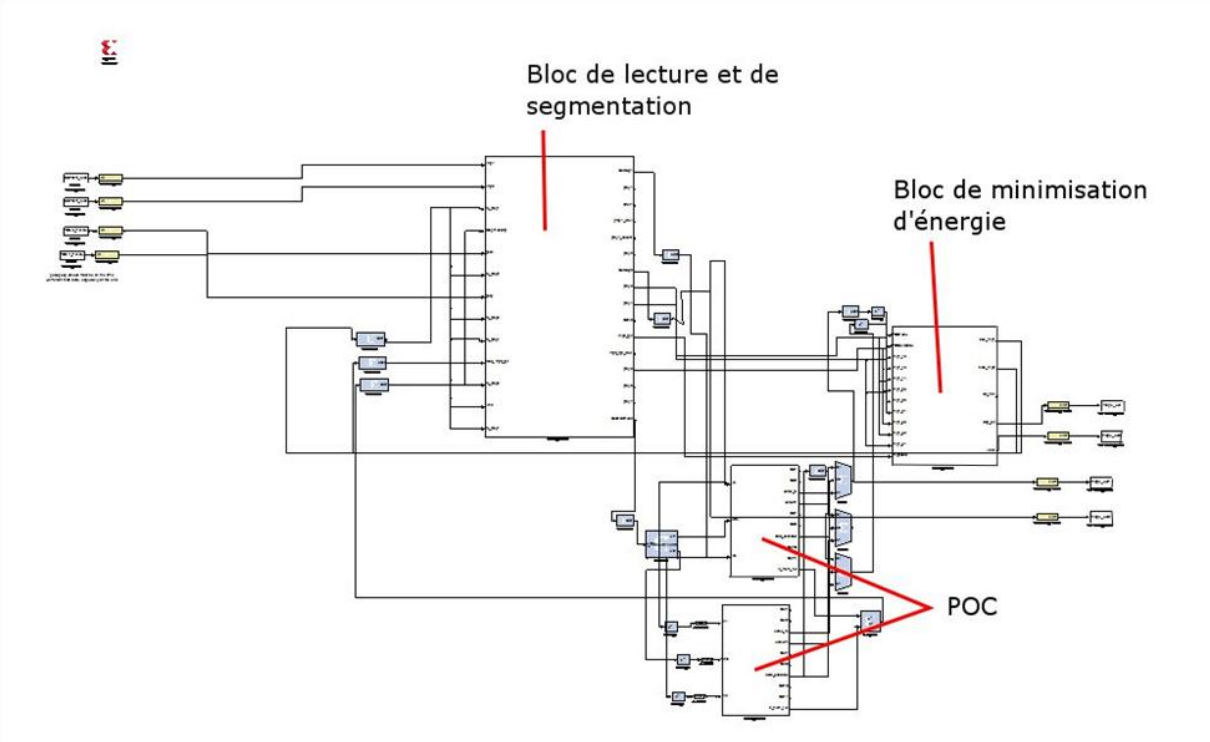


Figure 4.5-Principaux blocs de EPOC.

Le bloc de lecture et de segmentation est présenté en plus de détails dans Figure 4.6. Le module de lecture d'images lit les données essentielles issues des deux cameras. La configuration retenue requiert le calcul de $7 \times 7 = 49$ espaces de recherche. Pour calculer chacun de ces espaces, on a besoin d'une paire de signaux de 63 éléments chacun. Ceci dit, la taille des deux fenêtres par phosphène à stimuler retournée par ce bloc est de $(7 + 8{,}7 + 63) = (15{,}70)$. Plus tard, on effectue la transformée de Fourier de 64 points sur les paires de signaux de 63 éléments.

Les fenêtres de taille $(15,70)$ définies sur les deux images stéréoscopiques sont envoyées à 7 blocs de segmentation. Ces blocs (Figure 4.7) reçoivent chacun deux fenêtres de $(2N_1 + 1, 2N_2 + 1) = (9,63)$ qui sont tirés des deux fenêtres de dimensions $(15,70)$. Ils performent la segmentation de couleur sur chacune des deux fenêtres, mais aussi une segmentation spatiale en utilisant des poids spatiaux générés par une fonction externe qui alimente les 7 blocs de segmentation. Les fenêtres de taille $(2N_1 + 1, 2N_2 + 1) = (9,63)$ sont intégrés verticalement au niveau des blocs de segmentation pour aboutir à des vecteurs unidimensionnels de taille 63 éléments. Au niveau des blocs de segmentation, on sauvegarde des signaux utiles au calcul des fonctions d'énergie pour être utilisés plus tard.

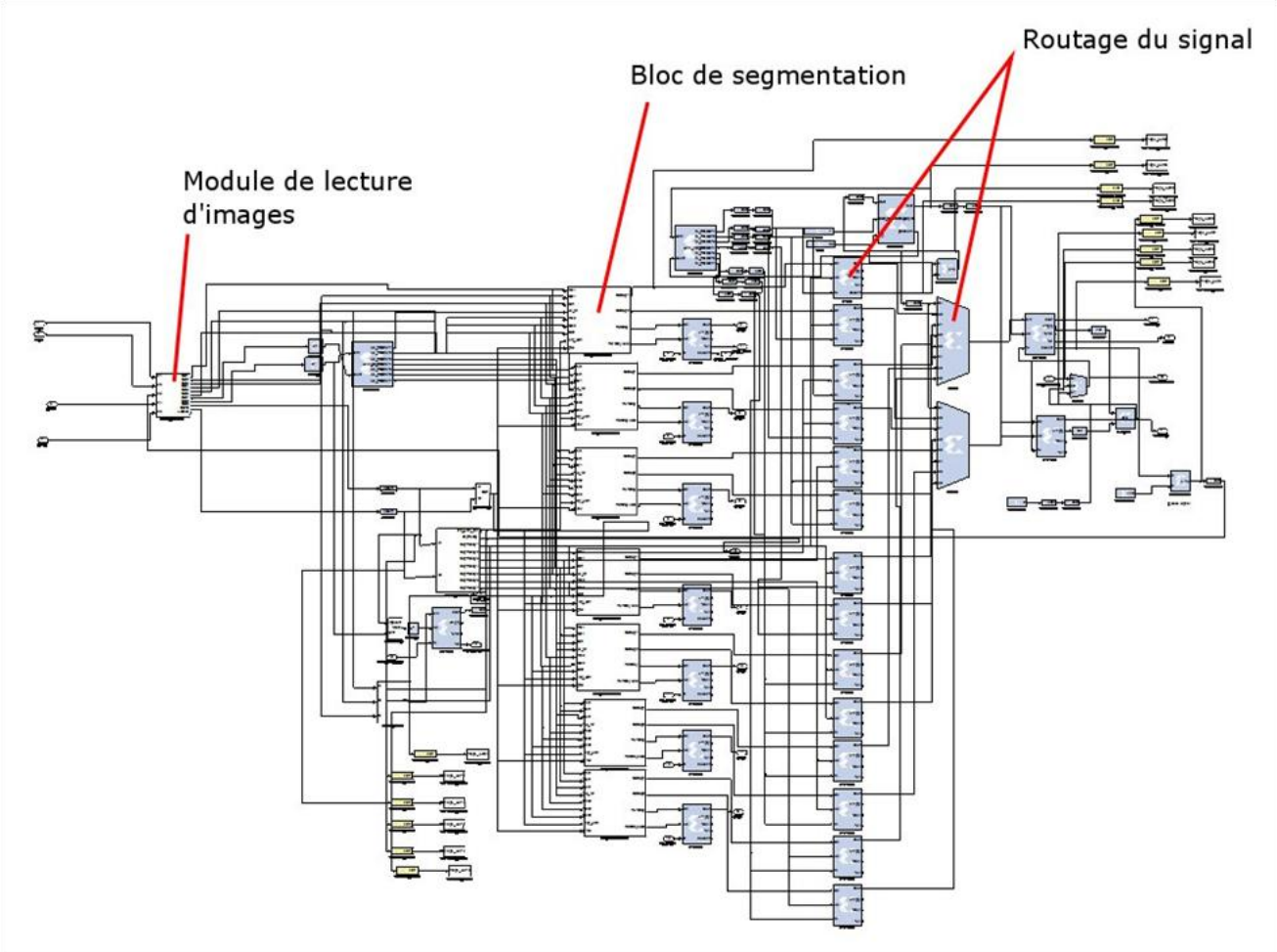


Figure 4.6 Bloc de lecture et de segmentation

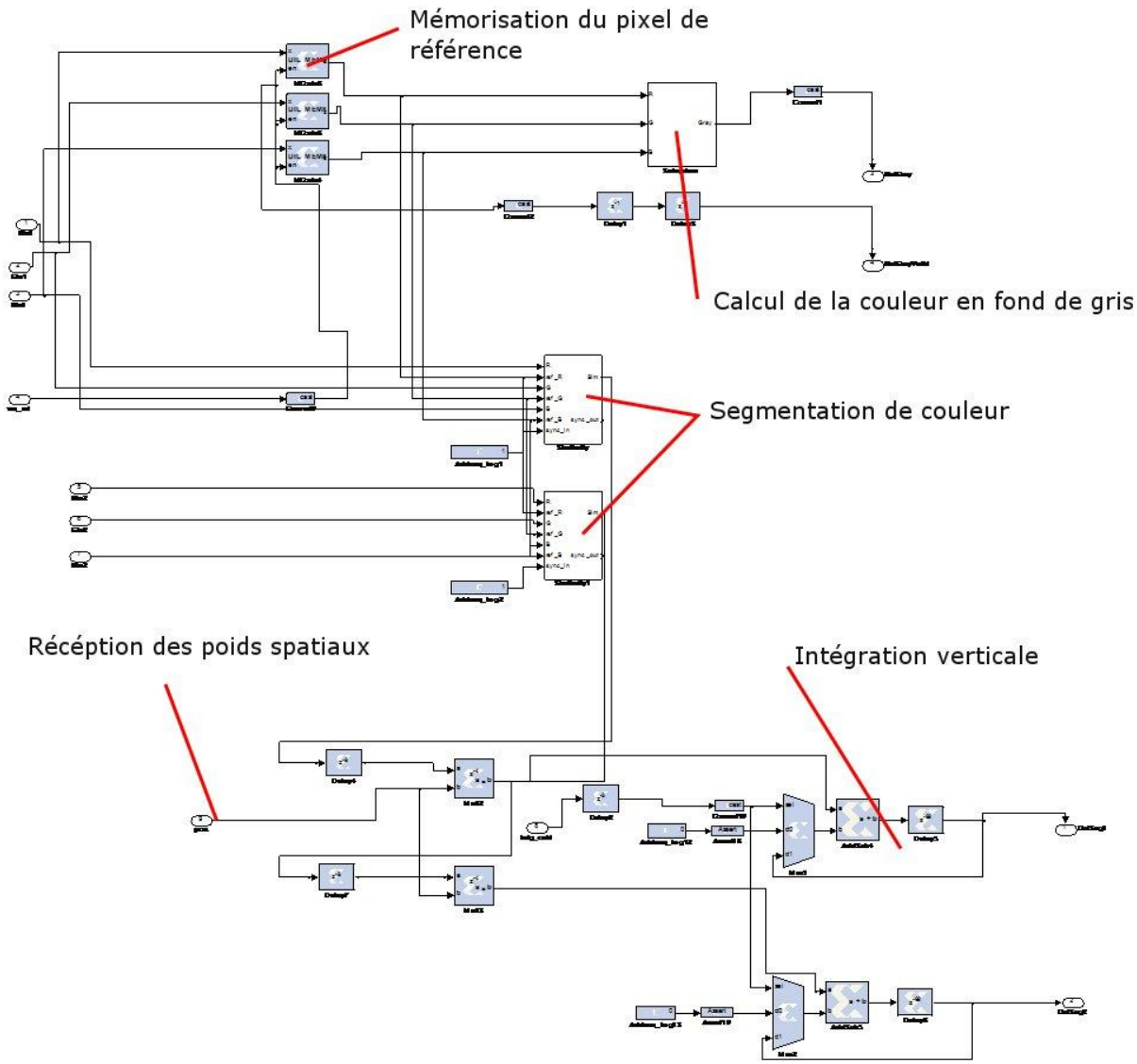


Figure 4.7 Bloc de segmentation

Les vecteurs unidimensionnels sont mémorisés dans deux FIFOs avant d'appliquer la fonction POC. Un routage spécial fait en sorte que la sortie de chaque bloc est mémorisée dans des FIFOs de petites tailles en parallèle. Le contenu de ces FIFO est ensuite routé aux FIFOs de plus grandes tailles. Ainsi, quelque soit le nombre de blocs de segmentation, la sortie du bloc de lecture et de segmentation est toujours composée essentiellement de deux signaux.

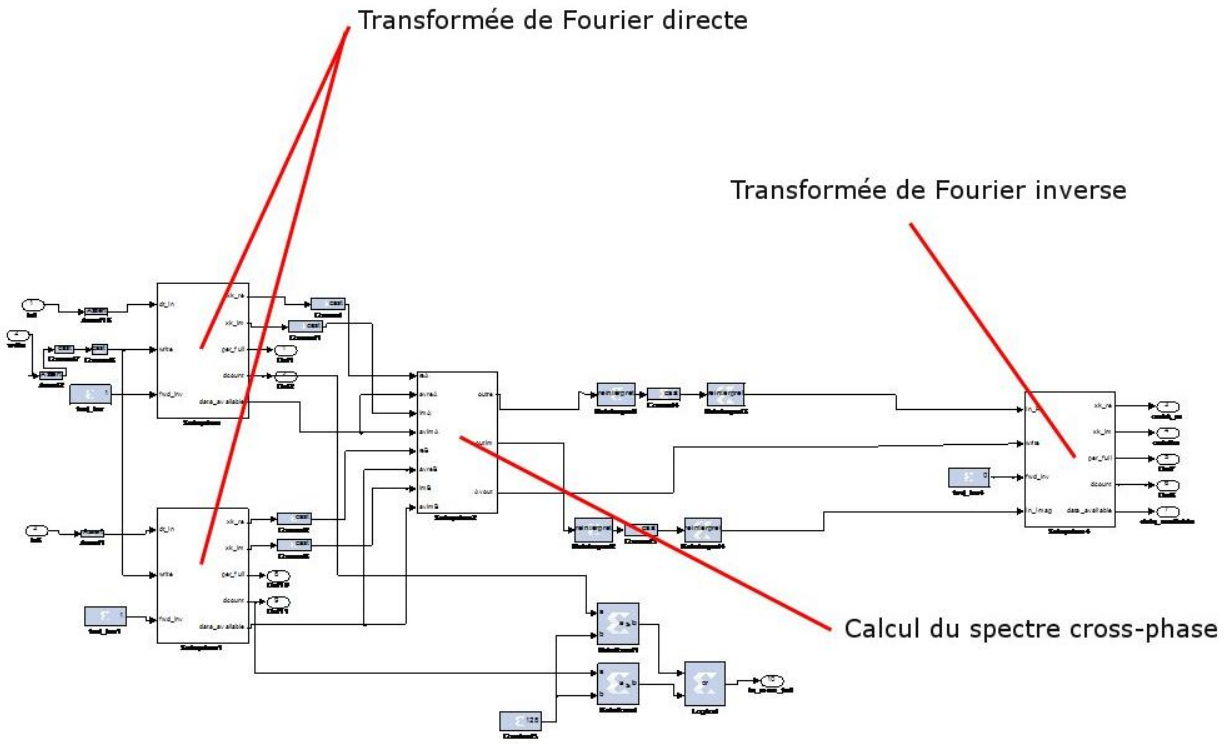


Figure 4.8 Bloc de corrélation de phase

L'implémentation d'EPOC comprend deux blocs de corrélation de phase pour plus de rapidité. On utilise des machines de Fourier à base 4. Le nombre de cycles nécessaires au calcul du spectre est environ 3 fois la longueur des signaux d'entrée. Un bloc de corrélation de phase est illustré dans Figure 4.8. Le spectre cross-phase comprend normalement une multiplication complexe suivie d'une division par la norme pour normaliser le spectre. Cependant, la multiplication de deux nombres donne un nombre de résolution supérieure de sorte que la division du résultat n'est plus pratique. Ainsi, on aura à réduire la résolution de la multiplication et le résultat de la division devient occasionnellement invalide. Pour pallier à ce problème, on normalise les spectres des deux signaux d'entrée avant de faire la multiplication. Le calcul du spectre cross-phase est illustré dans Figure 4.9.

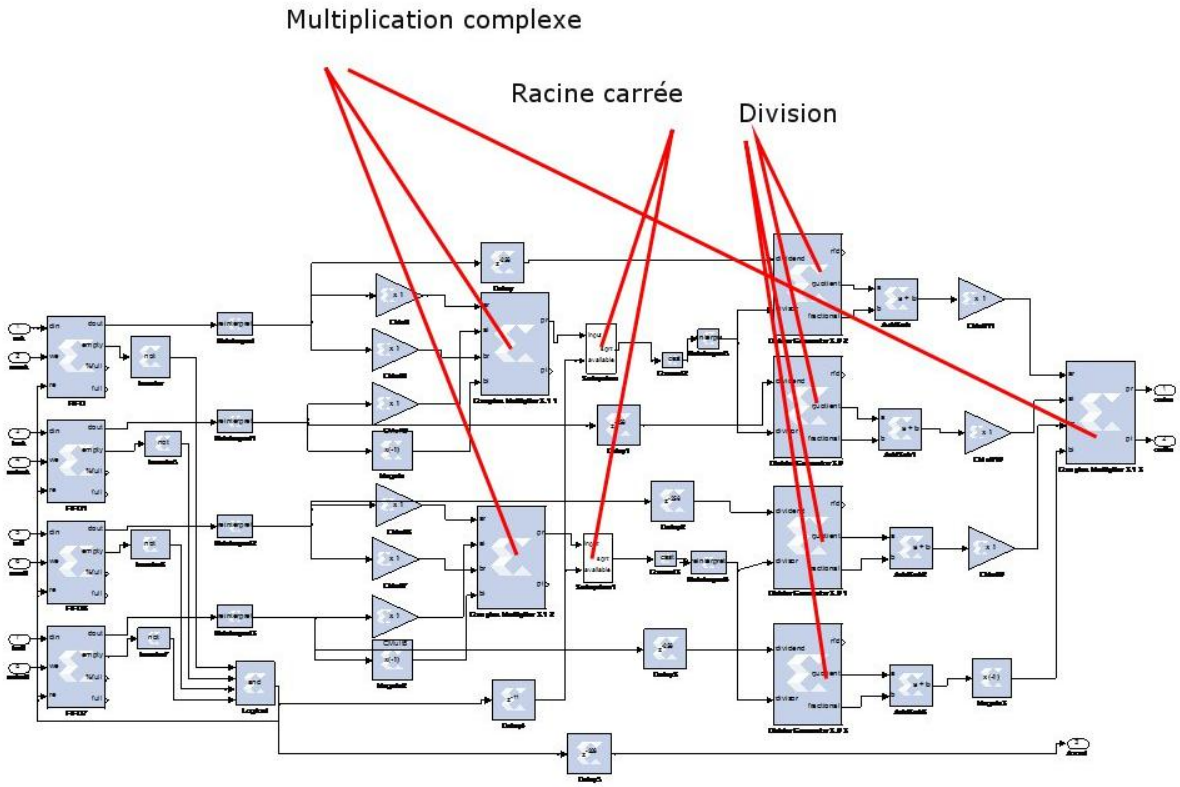


Figure 4.9 Calcul du spectre cross-phase.

La sortie des blocs de corrélation de phase alimente le bloc de superposition des espaces de recherche et de minimisation d'énergie (BSME). Celui-ci est illustré dans Figure 4.10. Puisqu'il existe 9 nœuds de type *k*, il existe 9 blocs qui font la superposition des espaces de recherche. Un signal reçu par le BSME peut être routé à plusieurs de ces blocs à la fois. L'argument du maximum est calculé au niveau de ces blocs. Cette dernière information est routée à 9 blocs de calcul d'énergie en parallèle. Enfin, un bloc cherche la disparité associée à l'énergie minimale.

Les résultats sont mentionnés dans Figure 4.11. Le nombre de cycles d'horloges recquis par l'application est moins que 750 000. Nous fournissons un graphique qui illustre les valeurs de disparités générés dans Figure 4.11. Les valeurs négatives réfèrent à des valeurs de disparités supérieures à 32 pixels. Ils peuvent référer à des singularités aussi.Un signal d'activation permet d'identifier les valeurs de disparités valides. Ce signal est illustré dans Figure 4.12.

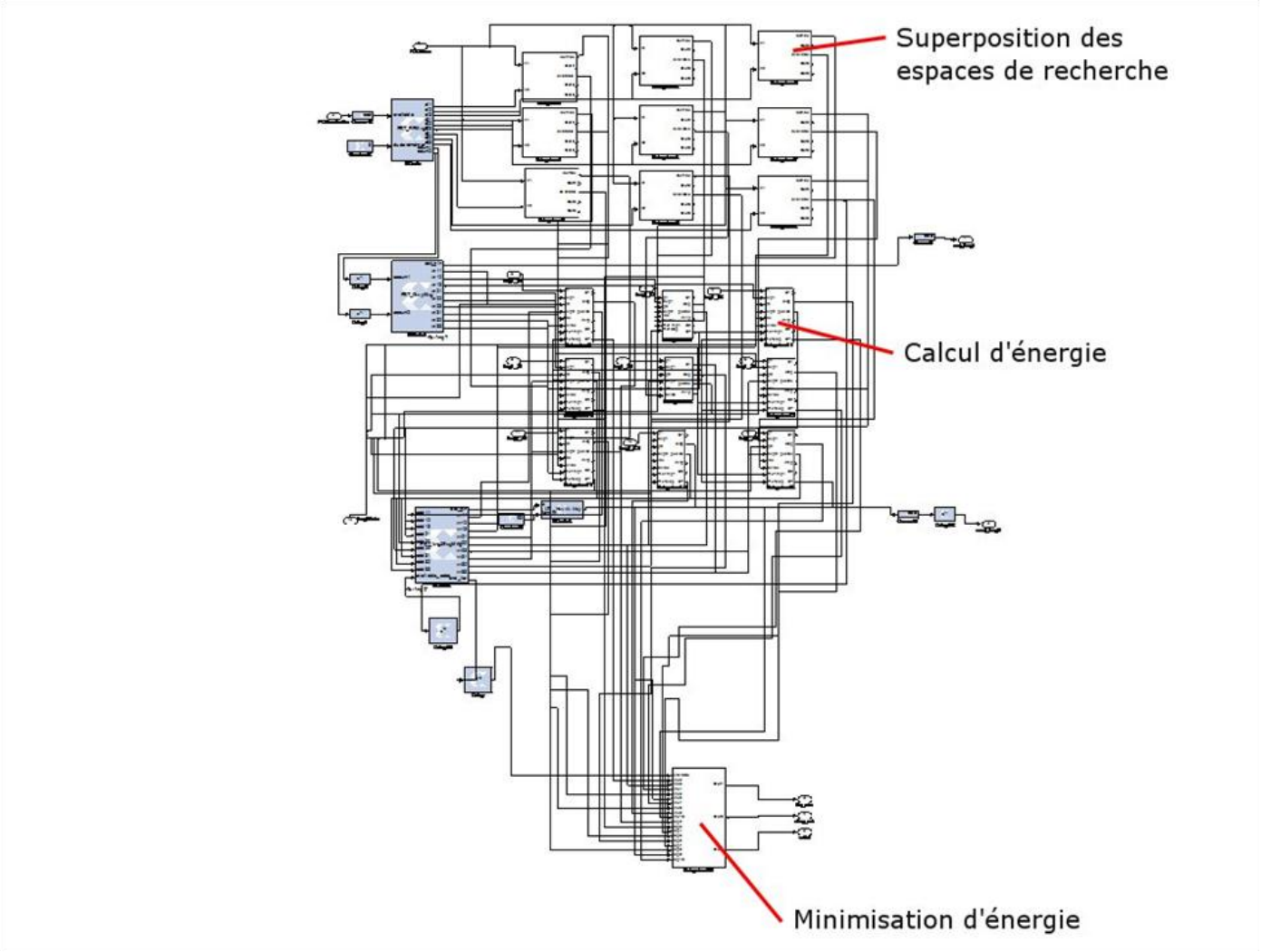


Figure 4.10 Bloc de superposition des espaces de recherche et de minimisation d’énergie

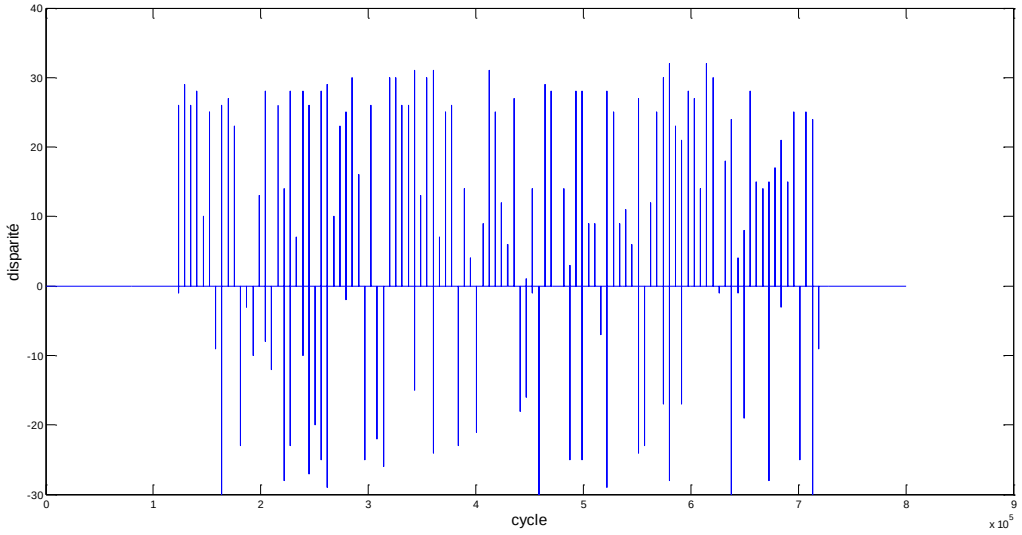


Figure 4.11 Génération des phosphènes par le système proposé.

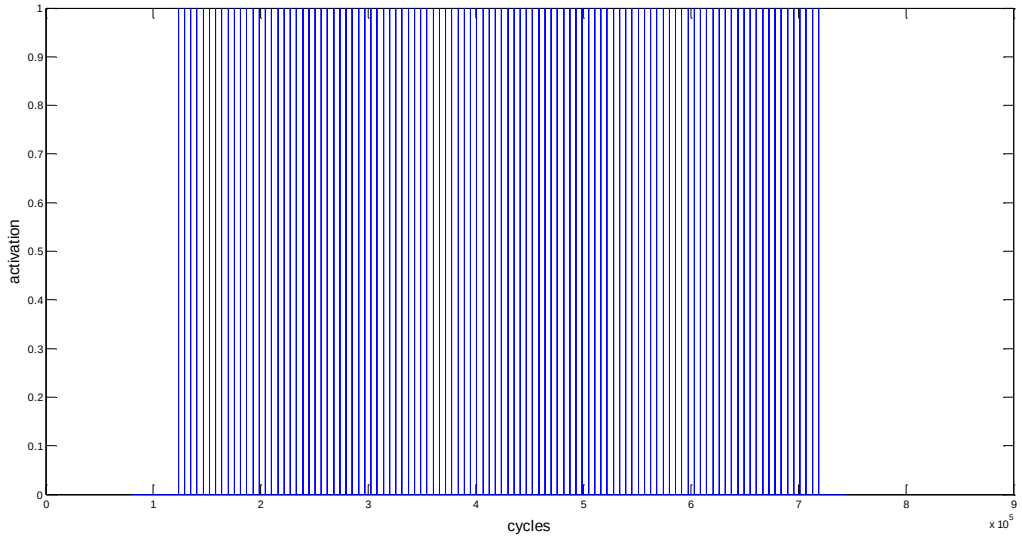


Figure 4.12 Signal d'activation qui permet d'identifier les valeurs de disparités à retenir.

ANNEXE2- Dérivation de la mémoire associative quantique de Hopfield

La mémoire associative de Hopfield quantique prend son origine d'une formule d'intégration du parcours d'une onde faite par Feynman (Feynman and Hibbs 1965). Elle est une reformulation de l'équation de Schrödinger et prend la forme :

$$\Psi(\vec{r}_2, t_2) = \iint G(\vec{r}_1, t_1, \vec{r}_2, t_2)\Psi(\vec{r}_1, t_1)\, d\vec{r}_1\, dt_1 \tag{5.1}$$

Ici, $\vec{r}$ est un vecteur-position, t est un indice temporel et G est connue sous le nom de propagateur de Green et prend la forme d'une superposition de P ondes ψ_k qui sont des fonctions-ondes propres:

$$G(\vec{r}_1, t_1, \vec{r}_2, t_2) = \sum_{k=1}^{P} \psi_k(\vec{r}_1, t_1)^* \psi_k(\vec{r}_2, t_2) \tag{5.2}$$

Et :

$$\psi_k(\vec{r}, t) = A_k(\vec{r}, t)e^{iS_k(\vec{r},t)} = A_k(\vec{r}, t)e^{i\left[\frac{1}{\hbar}(p^k\vec{r} - E^k t)\right]} \tag{5.3}$$

Où A_k et S_k sont respectivement l'amplitude et la phase et k est l'indice de l'état propre ψ_k. L'état propre ψ_k représente la distribution de probabilité –qui varie de façon sinusoïdale- de mesurer le mode k d'un photon de moment p^k et énergie E^k au moment t à l'endroit $\vec{r}$. Normalement $\hbar$ représente la constante de Planck, mais ici on pose $\hbar = 1$ pour simplification.

Dans l'équation (5.1), lorsque $t_1 = t_2$, le propagateur G doit reproduire l'état initiale. Ainsi, les contraintes suivantes doivent être respectées (Perus and Dey 2000):

$$\sum_{k=1}^{P} \psi_k(\vec{r}_1, t)^* \psi_k(\vec{r}_2, t) = \delta(\vec{r}_1 - \vec{r}_2) \quad ou \quad \sum_{k=1}^{P} \psi_k(\vec{r}_1)^* \psi_k(\vec{r}_2) = \delta(\vec{r}_1 - \vec{r}_2) \tag{5.4}$$

Ce qui veut dire que, si les ψ_k sont bien choisis, ils peuvent constituer une base orthonormale dans laquelle toute fonction d'onde propre peut être exprimée : $\Psi = \sum_k c_k \psi_k$.

Vu que le propagateur dans (5.2) est constitué essentiellement d'un produit externe qui ressemble à celui du Hebbien de la mémoire associative de Hopfield, on sait qu'il peut implémenter une mémoire associative lui-même. La différence majeure c'est qu'elle comprend des nombres complexes. Substituant (5.3) dans (5.2), on obtient:

$$G(\vec{r}_1, t_1, \vec{r}_2, t_2) = \sum_{k=1}^{P} A_k(\vec{r}_1, t_1) A_k(\vec{r}_2, t_2) e^{i(S_k(\vec{r}_2, t_2) - S_k(\vec{r}_1, t_1))} \tag{5.5}$$

Ceci veut dire que la mémoire G est encodée sur l'amplitude mais aussi sur la différence de phase $S_k(\vec{r}_2, t_2) - S_k(\vec{r}_1, t_1)$.

Si on dérive la phase S par rapport au temps, et qu'on note par δ l'operateur de différentiation temporelle, on obtient:

$$\delta S = \delta S_a + \delta S_b = \delta E t + E \delta t \tag{5.6}$$

Ce qui donne deux variations de G. La première, notée G_a est :

$$G_a(\vec{r}_1, \vec{r}_2) = \sum_{k=1}^{P} A_k(\vec{r}_1) A_k(\vec{r}_2) e^{i(E_2 - E_1)t} \tag{5.7}$$

Il s'agit de comparer deux niveaux d'énergie à un instant donné. La deuxième variation, G_b est :

$$G_b(\vec{r}_1, t_1, \vec{r}_2, t_2) = \sum_{k=1}^{P} A_k(\vec{r}_1) A_k(\vec{r}_2) e^{i(t_2 - t_1)E_k} \tag{5.8}$$

Dans cette dernière variation, la différence de phase est due à la différence temporelle entre deux instants et d'une énergie E_k constante.