Implantations temps réel du filtre de lissage d'images de Nagao

Real Time Implementations of the Nagao Image Smoothing Filter

par D. Demigny, J. Devars, L. Kessal, J.F. Quesne

Equipe Traitement des Images et du Signal (ETIS)
Ecole Nationale Supérieure de l'Electronique et de ses Applications (ENSEA)
Allée des chênes pourpre – 95014 Cergy Pontoise Cedex
Tél : 30 73 66 17 – 30 73 66 66

Résumé

Dans ce papier, nous présentons différentes implantations ASIC d'une version modifiée du filtre de lissage de Nagao utilisé dans une chaîne de segmentation par les contours d'images non texturées pour le contrôle de fabrication. Différentes variantes de l'algorithme initial de Nagao sont comparées en performance et complexité de réalisation. Différentes solutions architecturales matérielles d'une de ces variantes appelée *Nagmod* sont critiquées en terme de surface et rapidité. Certains aspects du design : l'exploitation du pipeline, la structure d'un opérateur de recherche de maximum, sont discutés.

Mots clés : Segmentation, Traitement d'images, Processeur temps réel, Détection de contours

Abstract

We present various ASIC implementations of the Nagao smoothing filter. This filter is used to segment non textural pictures with an edge detection approach. Some changes of the initial algorithm are tested both in term of complexity and performance. Hardware implementations are compared for speed and for silicon area used. Design aspects, like pipelining and maxima computation are discussed.

***Key words** : Segmentation, Image Processing, Real time processor, Edge detection*

1. Introduction

Les avancées de la technologie dans le domaine des VLSI permettent d'implanter des algorithmes de plus en plus complexes dans les processeurs de traitement du signal (DSP). Il y a quelques années, la puissance nécessaire au traitement temps réel des images (25 images par seconde) imposait l'usage de circuits spécifiques cablant l'algorithme. Aujourd'hui, l'association de circuits programmables : convolueurs [1], corrélateurs binaires [2] pour la segmentation, réseaux de transputers, machine RISC pour l'extraction de caractéristiques et l'interprétation satisfont à la contrainte de rapidité. Toutefois, à performances identiques, le coût reste le critère de décision pour les applications industrielles. Ainsi, la réduction du nombre de circuits, du nombre des entrées et sorties de chaque circuit, la limitation de la puissance dissipée et la surface de silicium utilisée sont déterminants. Dans le domaine de la segmentation des images, l'usage d'ASIC satisfait au mieux ces critères, les évolutions futures de la technologie permettront d'associer de plus en plus de traitements dans un même circuit. Si on réalise une étude pour minimiser la surface occupée par un opérateur (lissage, gradients, norme, fermeture de contours, etc) il ne faut pas penser que cet effort a pour but d'atteindre un seuil au delà duquel à un instant donné de l'évolution technologique le traitement devient intégrable dans un seul circuit, mais bien chercher la surface la plus faible de façon à inclure de plus en plus de traitements sur le même circuit.

Dans ce papier, nous présentons différentes implantations d'une version modifiée du filtre de lissage de Nagao utilisé dans une chaîne de segmentation par les contours d'images non texturées pour le contrôle de fabrication. Le critère de rapidité étant aisément satisfait, nous comparons ces architectures en termes de surface. Dans la section suivante, nous décrivons une chaîne d'analyse d'images étudiée par la société Allen Bradley Servovision et notre équipe [4].

2. Système de vision pour le contrôle de fabrication

Typiquement un système de contrôle de fabrication peut être décomposé en trois parties distinctes.

• Le bloc de segmentation fournit à partir d'une image d'intensité en niveau de gris, une image de contours fermés et une image de régions.

• Le bloc d'extraction des caractéristiques détermine pour chaque région des paramètres tels que surface, périmètre, moments d'ordre 2.

• Le bloc de reconnaissance de formes et d'interprétation associe par exemple un ensemble de régions de l'image à une forme apprise.

Ce dernier bloc, compte tenu de la versatilité des stratégies utilisables, de la complexité des traitements et de la plus faible quantité de données manipulées est réalisé à l'aide de processeurs de traitement du signal ou même de machines d'usage général (RISC). L'extraction des caractéristiques de l'image a été réalisée par Allen Bradley en un seul ASIC comprenant un codage des contours en chaînes de Freeman et un calcul des paramètres précédemment définis à partir des chaînes de Freeman. Nous précisons maintenant l'organisation et les contraintes de la chaîne de segmentation qu'on peut décomposer en six étapes :

• un lissage de l'image destiné à éliminer une partie du bruit (homogénéisation des régions);

• le calcul des gradients directionnels;

• la suppression des non maxima locaux du gradient dans la direction du gradient destinée à fournir des contours fins;

• un double seuillage de l'image du gradient;

• la fermeture des contours;

• l'étiquetage des régions.

Deux problèmes surviennent liés à l'approche consistant à détecter les contours par le gradient : un seuil élevé élimine les faux contours à gradient faible mais laisse les contours non fermés dans les zones de plus faible contraste. Un seuil faible donne des contours fermés mais laisse subsister de nombreux faux contours dus au bruit. Ce dernier problème survient aussi dans les approches de détection de contours par le passage à zéro du Laplacien. Pour contourner ces difficultés et rendre moins critique le choix du seuil, nous proposons d'utiliser deux seuils. Tous les contours supérieurs au seuil haut seront considérés comme contours sûrs, les contours dont le gradient est compris entre les deux seuils que nous appellerons crêtes pour les différencier des précédents pourront être utilisés dans la phase de fermeture. Les contours inférieurs au seuil bas sont transformés en fond. On définit l'extrémité d'un contour sûr comme un pixel ayant un seul voisin de type contour sûr, la fermeture des contours consiste alors à ne retenir parmi les crêtes que celles connexes d'un côté à une extrémité et de l'autre à une extrémité où à un contour sûr [figure 1]. L'étiquetage des régions affecte alors une étiquette différente à chaque région délimitée par un contour fermé.

Les contraintes de réalisation sont les suivantes : Un débit de 25 images de 512 × 512 pixels par seconde soit une fréquence

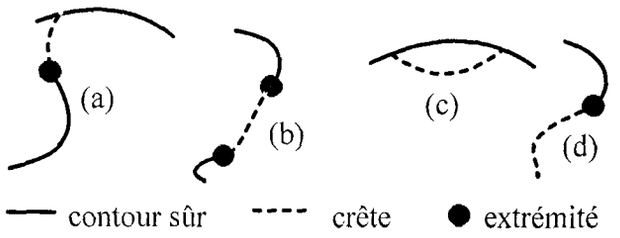

Figure 1. – Les crêtes ne seront conservées que dans les cas (a) et (b).

d'horloge pixel de 10 MHz, un temps de latence maximum (traversée de la chaîne de segmentation) d'une trame et quelques lignes. Le critère industriel de coût impose que l'ensemble des opérateurs soit intégré sur deux ASICS en technologie CMOS 1,2 μm. Pour satisfaire ces contraintes et minimiser le nombre de mémoires utilisées, les traitements sont effectués sur le flot mono-dimensionnel des pixels issus d'un balayage ligne de l'image. L'exécution de la chaîne de segmentation complète sur un HP9000-825 (7Mips) requiert 16 s, c'est donc un facteur 400 qui est à gagner en rapidité; soit encore l'équivalent de 30 convolutions 3 × 3 par pixel à la cadence vidéo. Au delà de ce projet, il est intéressant de systématiquement rechercher une implantation des opérateurs utilisant une surface de silicium la plus faible possible : l'évolution se faisant vers une plus grande complexité des traitements conduit à associer un nombre croissant d'opérateurs sur un même circuit.

L'exploitation de l'algorithme de Rosenfeld [3] nécessitant un unique balayage de l'image pour l'étiquetage des composantes connexes a été utilisé tant pour l'étiquetage des régions que pour la fermeture des contours. La fusion de ces deux traitements et l'étude de l'architecture résultante [4] a conduit à la définition et à la simulation d'un ASIC de 60 mm^2. Une autre approche pour la fermeture des contours basée sur l'utilisation d'automates cellulaires exploitant aussi le double seuillage a été évaluée [5]. Les quatre étapes restantes consistent en une succession de traitements locaux. Nous avons étudié leur implantation en un seul circuit.

Dans la suite de ce papier, nous nous intéressons à l'implantation du filtre de Nagao. Dans la section 3, nous décrivons l'algorithme de Nagao, et montrons l'influence des modifications que nous y avons apportées sur la qualité du traitement. Précisons tout de suite que nous ne cherchons pas à situer l'intérêt du filtre de Nagao par rapport à d'autres types de filtres possibles, ceci ayant été largement étudié par ailleurs [7], [8], [9]. Dans la section 4, nous décrivons et critiquons trois architectures possibles en termes de ressources et d'opérateurs. Les aspects particuliers de la conception sont décrits dans la section 5. Enfin, une évaluation des performances est donnée dans la dernière section.

3. L'algorithme de Nagao

De nombreux articles ont exposé des méthodes de filtrage tentant de résoudre le conflit entre un bon filtrage et la préservation du contraste. Un lissage efficace s'accompagne généralement d'une

érosion (diffusion) des contours. Les filtres de Sobel réalisent un lissage directionnel conjugué à un calcul du gradient dans la direction orthogonale. Le faible nombre de points intervenant dans le lissage limite la qualité de celui-ci. Le filtre de Deriche reprend cette idée en exploitant un voisinage de plus grande dimension associé à un paramétrage de la réponse impulsionnelle qui permettent d'une part un meilleur lissage, et d'autre part une détection optimale d'un certain type de contour. L'implantation non récursive de ce filtre avec un noyau de taille 9×9 requiert, compte tenu des symétries, 10 multiplieurs. La forme récursive de ce filtre réduit ce nombre mais augmente les contraintes de rapidité sur les multiplieurs et nécessite des balayages plus complexes de l'image : horizontal dans les deux sens et vertical dans les deux sens.

Une autre approche développée initialement par Tsuji et Tomita [6] consiste à affecter au point central d'un voisinage 5×5, la moyenne du voisinage 3×3 parmi ceux de centre (I, A, B, C, D) [figure 2a] qui présente la variance la plus faible. L'idée directrice est qu'un voisinage 3×3 incluant un contour conduira à une variance plus forte qu'un voisinage uniquement perturbé par le bruit. Nagao [7] a ensuite montré que la forme des voisinages 3×3 du filtre de Tsuji induisant la suppression des petites régions ce qui devient dommageable si l'algorithme est itéré pour améliorer le lissage. Dans ce cas, de proche en proche, des régions de plus en plus grandes seront fusionnées. Il propose alors de choisir les masques [figure 2b], [figure 2c] associés au masque carré 3×3 centré sur le point à traiter pour les calculs de variance et de moyenne : ce qui constitue le filtre de Nagao original. Ces formes de voisinage permettent une meilleure détection des angles [7]. Devars et Cocquerez [8] ont notamment montré que ce lissage associé à un gradient directionnel est bien adapté à la détection des contours dans les images aériennes où certaines régions sont de largeur réduite (routes, carrefours).

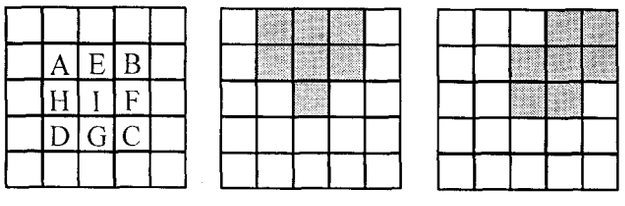

Figure 2a. – Les centres A, B, C, D et I des voisinages 3×3 utilisés dans le filtre de Tsuji, on y adjoint les voisinages de centre E, H, F, G pour le filtre Nagmod.

Figure 2b et c. – Par rotations multiples de 90° des voisinages des figures (b) et (c), et ajout du voisinage 3×3 central, on définit l'ensemble des voisinages utilisés par le filtre de Nagao original.

La réalisation du filtre de Nagao est rendue difficile quoique réalisable par l'irrégularité des voisinages et le calcul de la variance. Dans un but de simplification de la réalisation, nous proposons un nouveau filtre que nous appellerons «**Nagmod**» que nous définissons de la manière suivante : pour chacun des 9 voisinages 3×3, A, B, C, D, E, F, G, H, I [figure 2a], nous calculons la somme et l'étendue des intensités des pixels, puis nous affectons au point central du voisinage 5×5, la somme calculée pour le voisinage 3×3 qui possède l'étendue la plus faible. L'étendue comme critère d'homogénéité est aussi significative que la variance sur un voisinage de taille réduite et est aussi plus simple à calculer. Nous avons complété les voisinages définis par Tsuji [figure 2a] en exploitant aussi les voisinages 3×3 de centre E, F, G, H. D'autre part, la division par 9 dans le calcul de la moyenne des intensités d'un voisinage 3×3 a pour but de se ramener à la dynamique de l'image de départ. Si pour une image initiale codée sur 8 bits, on conserve la moyenne sur 8 bits alors une erreur de troncature importante est commise. En n'effectuant pas la division et en conservant la somme exacte sur 12 bits, l'erreur de troncature est supprimée. Bien sûr en conservant 4 bits supplémentaires dans le calcul de la moyenne (notation en virgule fixe), l'erreur commise sur le calcul de la moyenne devient négligeable. Nous avons préféré supprimer la division inutile et conserver le résultat sur 12 bits.

Différents filtres ont été testés dans le rapport segmentation publié par le groupe GT8 du Greco 134 [9] et notamment la comparaison du filtre de Deriche au filtre de Nagao. Nous proposons ici de simplement vérifier que les modifications introduites par rapport au filtre de Nagao original ne dégradent pas la qualité du résultat. Nous présentons une comparaison des filtres de Tsuji, Nagao et Nagmod avec un résultat de type somme (sans troncature sur 12 bits) ou moyenne (avec troncature sur 8 bits). L'image de test contient deux zones homogènes séparées par une transition oblique à 60° d'amplitude 6 à laquelle on superpose un bruit gaussien ($\sigma = 2,2$). Cette image de synthèse utilisée dans [9] est donc largement connue de la communauté Image. Différents lissages dans une zone homogène de l'image de 750 pixels sont comparés [tableau 1].

Tableau 1. – Rapport σ/σ_f du bruit avant filtrage au bruit après filtrage avec et sans troncature

Filtre	moyenne	somme
Tsuji variance	2,33	3,75
Tsuji étendue	2,52	3,20
Nagao variance	2,90	3,50
Nagao étendue	2,28	3,44
Nagmod variance	2,53	3,19
Nagmod étendue	2,50	3,19
Moyenne/somme	2,54	3,24

La première colonne indique le type de filtre et le critère d'homogénéité retenu. Les colonnes 2 et 3 donnent le rapport σ/σ_f du bruit avant filtrage au bruit après filtrage. La dernière ligne du tableau correspond à un filtre moyenneur 3×3 classique. Le meilleur filtre calculant une moyenne est celui de Nagao avec critère d'homogénéité de type variance. Par contre, la suppression de

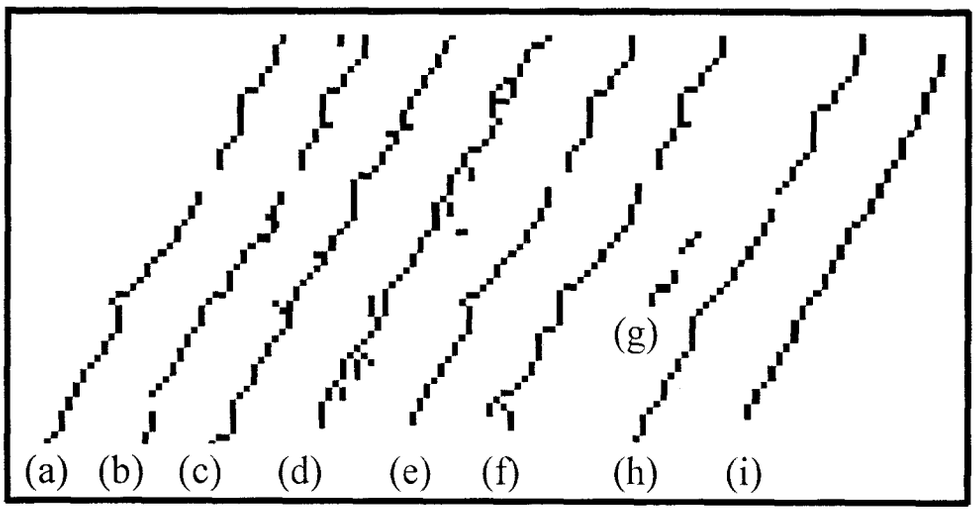

Figure 4. – (a) Tsuji étendue, (b) Tsuji variance, (c) Nagao étendue, (d) Nagao variance, (e) Nagmod étendue, (f) Nagmod variance, (g) moyenne, (h) Nagmod étendue non tronqué, (i) Deriche $\alpha = 0,5$.

la troncature amène Nagmod avec calcul d'étendue à un résultat meilleur que le Nagao original tronqué à 8 bits. Le filtre de Tsuji utilisant comme critère la variance semble en l'absence de troncature présenter la qualité de filtrage la meilleure, par contre le faible nombre de voisinages utilisés peut induire un décalage de la valeur moyenne en sortie en augmentant la probabilité qu'un voisinage décentré par rapport à la moyenne, de faible variance, soit retenu pour plusieurs pixels voisins. En tout état de cause, du point de vue du lissage, les résultats ne sont guère meilleurs que pour un simple moyenneur!

Ces différents filtres sont aussi comparés du point de vue de la détection des contours [figure 4] toujours pour la même image de synthèse. Dans chaque cas, les contours sont détectés par calcul des gradients directionnels en utilisant les filtres [figure 3], puis par calcul de la norme, suppression des non maxima locaux du gradient dans la direction du gradient et seuillage (seuil = 3).

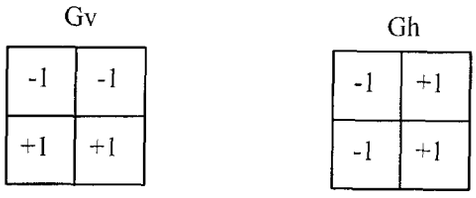

Figure 3. – Masques de calcul des gradients directionnels.

On constate que la troncature introduit des égalités dans les gradients gênant la suppression des non maxima locaux en (b) et (d). En (g), le moyenneur émousse les contours. En (h) le filtre Nagmod donne un résultat correct sans atteindre les performances du filtre de Deriche (i). Nous avons réalisé une analyse statistique [tableau 2] des images de la figure 4. Pour cela nous avons utilisé les critères P1 et P2 de Fram et Deutsch [13] et [9]. P1 mesure la robustesse du détecteur de contours en présence de bruit et P2 caractérise la répartition des points détectés appartenant réellement au signal dans la zone contour. La zone contour est définie comme ayant une largeur de 2 pixels ce qui est plus contraignant que le choix initial de Fram et Deutsch : ± 1 pixel autour du contour théorique. On constate l'intérêt de conserver une information complète sur 12 bits.

Tableau 2. – Critères de Fram et Deutsch pour les variantes du filtre de Nagao

Filtre	P1	P2
Tsuji variance (8 bits)	0,67	0,55
Tsuji étendu (8 bits)	0,76	0,64
Nagao variance (8 bits)	0,69	0,88
Nagao étendue (8 bits)	0,68	0,88
Nagmod variance (8 bits)	0,71	0,83
Nagmod étendue (8 bits)	0,65	0,69
Nagmod étendue (12 bits)	0,85	0,84

Les filtres de Nagao et Nagmod sont appliqués à une image réelle issue de la banque d'images du Greco 134 [figure 5]. Les trois images (b), (c), (d) ont été obtenues avec le même seuil S_0 de détection des contours. Pour l'image (e), les seuils bas et haut correspondent respectivement à 50% et 150% de S_0. L'image (e) conduit après fermeture à l'image (f). On constate qu'à seuil identique, le filtre de Nagao fournit des contours plus bruités que Nagmod. Le double seuillage permet de ne conserver que

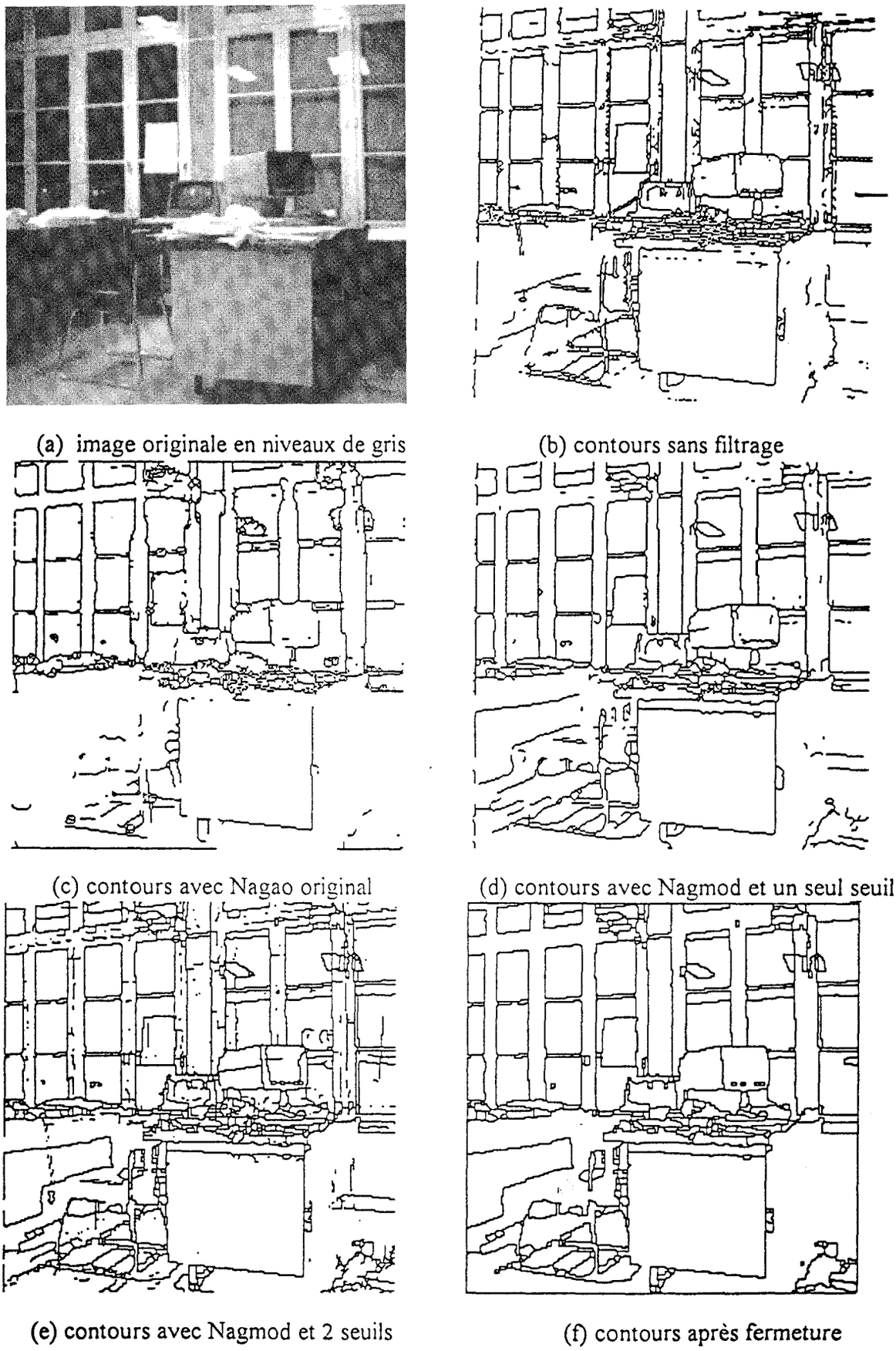

Figure 5. – (a) image originale en niveaux de gris; (b) contours sans filtrage; (c) *contours* avec Nagao original; (d) contours avec Nagmod et un seul seuil; (e) contours avec Nagmod et 2 seuils; (f) contours après fermeture.

les crêtes qui sont connexes à l'*extrémité* d'un contour sûr, ce qui constitue un critère plus contraignant que le seuillage à hystérésis.

On peut remarquer que le lissage s'accompagne d'un rehaussement du contraste qui est illustré [figure 6] dans le cas mono-dimensionnel pour Nagmod.

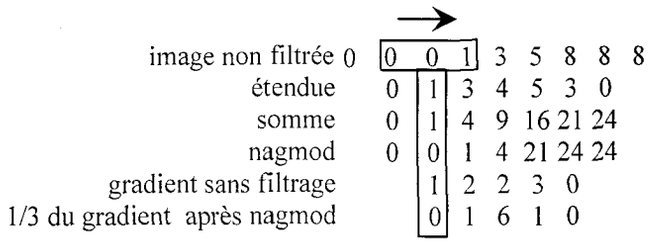

Figure 6. – Rehaussement du contraste par traitement d'un voisinage 3 × 1 avec Nagmod.

Dans le cadre de la vision industrielle, la résolution peut être adaptée ce qui limite l'intérêt de la détection des petites régions, et d'autre part le processus de lissage n'est pas itéré. Le filtre de Nagao peut alors être remplacé par Nagmod. La régularité des voisinages de Nagmod induit une implantation matérielle plus simple. Les tests ont montré que l'étendue est un critère aussi valable que la variance pour décider de l'homogénéité ce qui va là encore dans le sens d'une simplification de l'architecture matérielle. Dans le cas d'images fortement bruitées, le critère spécifiant qu'une région homogène possède une variance ou une étendue plus faible qu'une zone de contours ne tient plus, dans ce cas l'emploi du filtre de Deriche s'impose.

4. Implantation de «Nagmod»

La version de Nagmod retenue consiste à affecter au point central du voisinage 5 × 5, la *somme* du voisinage 3 × 3 parmi ceux [figure 2a] qui présentent l'*étendue* minimale. Dans cette section, nous proposons trois solutions sans pipeline, de rapidité équivalente pour implanter ce filtre. Le critère de comparaison est la surface de silicium utilisée. Dans la section suivante, nous discuterons des possibilités améliorant la rapidité du traitement. Ces solutions sont décrites ici en termes d'opérateurs (add, max, min, sélection) et de ressources de stockage (registre et mémoire). Pour raisonner de manière relativement indépendante des progrès de la technologie, mais suffisamment quantitative, nous calculons les coûts en terme de surface d'inverseurs équivalent (Ie) en supposant que la réduction de surface d'un inverseur réduira dans les mêmes proportions celle des différents opérateurs. Une estimation du nombre d'Ie de chaque opérateur et ressource est donnée [tableau 3]. Bien sûr, ces estimations varient d'une technologie à l'autre et ces chiffres ne sont intéressants que relativement et non en valeur absolue.

Nous allons montrer que dans le cas du filtre de Nagao, le coût en mémoire est prépondérant devant le coût en opérateurs. Cette remarque peut être appliquée à d'autres filtres : par exemple l'intérêt de l'utilisation des méthodes de factorisation des filtres de Sobel mais aussi des filtres à noyaux larges discutées dans [12] peut être analysé sous un autre aspect grâce à ce critère. D'autre part, des progrès dans les outils de conception conduiront à un renforcement de ce résultat. En effet, l'implantation optimale en surface des mémoires a été obtenue aisément de part leur organisation régulière. Par contre, des progrès restent à faire en placement et routage afin de compacter davantage les opérateurs. Leur coût en surface devrait donc diminuer relativement plus vite que celui des mémoires.

Tableau 3. – Nombre d'inverseurs équivalents (*Ie*) de différents opérateurs

Type d'opérateur (cellule binaire cascadable)	*Ie*
Additionneur	14
Recherche du min ou du max de 3 nombres	27
Sélection d'un mot parmi 3	13
Registre D	13
Point mémoire	1

Nous montrons dans la section suivante qu'il est possible de réaliser le minimum ou le maximum de trois mots avec un seul opérateur. L'opérateur de sélection fournit la somme associée à l'étendue minimale. En cas d'égalité entre plusieurs étendues, un mécanisme de priorité arbitraire sélectionne une somme parmi celles ayant des étendues identiques.

Dans tous ces cas envisagés, les pixels de l'image d'intensité arrivent dans l'ordre d'un balayage ligne. On peut ainsi considérer l'image comme un flot mono-dimensionnel de pixels. L'arrivée de deux pixels consécutifs d'une même ligne est séparée d'un coup d'horloge (retard z^{-1} au sens de la transformée en z), l'arrivée de deux pixels d'une même colonne de deux lignes consécutives de N coups d'horloge (retard z^{-N}) où N est le nombre de pixels par ligne. Si un retard de z^{-1} est matérialisé par un registre, un retard z^{-N} utilise par contre une mémoire de N mots organisée en «fifo» (First In First Out Memory) pour stocker l'ensemble des pixels d'une ligne.

Cette façon de considérer l'image rend compte de la manière dont sont traités les bords verticaux de l'image. Le pixel le plus à droite est voisin du pixel de la ligne suivante le plus à gauche et les deux dernières lignes d'une image sont voisines des deux premières de l'image suivante. Bien sûr le résultat des deux premières et des deux dernières lignes est erroné mais ceci est peu important puisqu'un bord de type contour sera créé artificiellement pour l'étiquetage des régions. Le problème des bords reste localisé à la demi-largeur du masque de convolution pour les traitements non récursifs. Le problème est plus critique avec des filtres récursifs tels que le filtre de Deriche pour lequel avec un paramètre α de valeur 0, 1 l'influence du bord est non négligeable même à 40 pixels de celui-ci !

D'un point de vue temporel, la traversée d'un opérateur flot de données introduit un temps de latence entre entrée et sortie, ceci

est pris en compte en retardant le signal vidéo de blank ou de validation pixel entre entrée et sortie.

Nous comparons les solutions en fonction de N et montrons que le choix est d'autant plus évident que N est grand. Quelle que soit la solution, le traitement fait appel à deux blocs : un premier bloc B1 [figure 7] calcule somme et étendue d'un voisinage 3×3, le deuxième B2 [figure 8] détermine le voisinage d'«étendue minimale parmi trois voisinages 3×3 puis sélectionne la somme associée à ce voisinage.

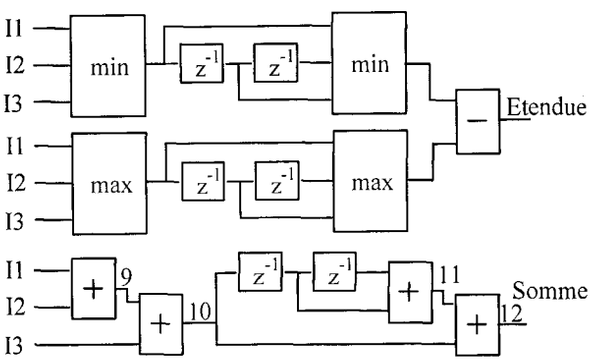

Figure 7. – Bloc B1, calcul de la somme et de l'étendue d'un voisinage 3×3.

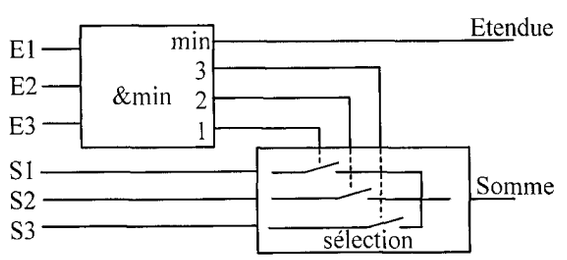

Figure 8. – Bloc B2, Sélection de la somme associée à l'étendue minimale parmi 3.

Sauf indication contraire sur les schémas, les bus ont une taille de 8 bits.

Nous présentons maintenant trois solutions d'implantation du filtre Nagmod.

Solution n° 1

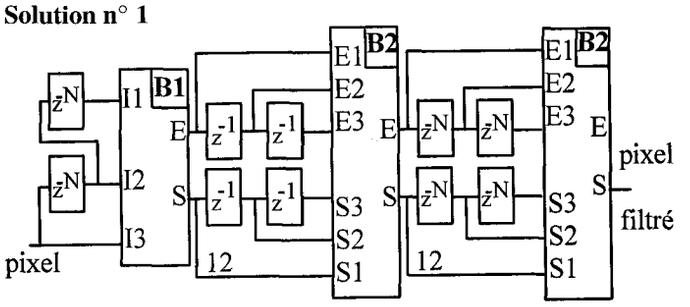

Figure 9. – Solution n° 1, 6 fifos utilisées.

On peut dire que cette solution [figure 9] calque l'algorithme dans la mesure où on commence à élaborer avec un bloc B1 les informations d'étendue et de somme, un premier bloc B2 sélectionne le meilleur voisinage de la ligne courante parmi les trois colonnes consécutives, le second bloc B2 sélectionne parmi ceux-ci le meilleur parmi trois lignes consécutives.

Solution n° 2

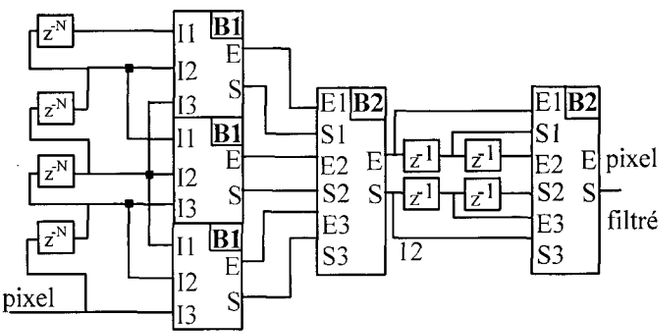

Figure 10. – Solution n° 2, 4 fifos utilisées.

On choisit de positionner l'ensemble des fifos de ligne en entrée [figure 10]. On peut alors considérer qu'à chaque cycle entre un vecteur colonne de pixels de cinq lignes consécutives. On calcule alors, somme et étendue de trois voisinages 3×3 d'une même colonne à l'aide des trois blocs B1. Le premier bloc B2 sélectionne le meilleur d'entre eux. Le second bloc B2 sélectionne parmi ceux-ci le meilleur parmi les trois colonnes consécutives.

Solution n° 3

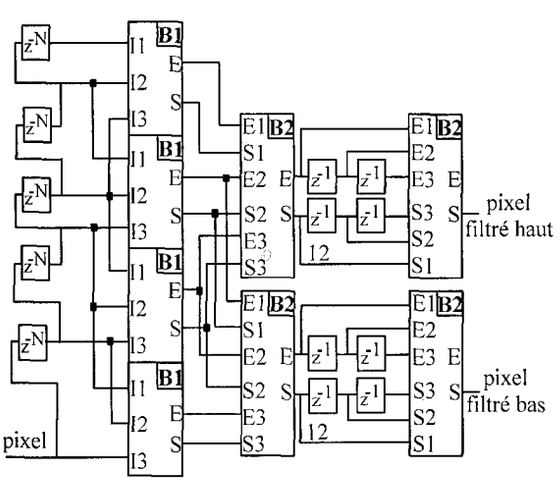

Figure 11. – Solution n° 3, calcul de 2 points lissés.

Cette solution [figure 11] reprend le principe de la solution 2 mais calcule deux points lissés de deux lignes consécutives. On peut donc sans fifo supplémentaire calculer les composantes du gradient à l'aide de l'opérateur défini [figure 3] alors que dans

les solutions 1 et 2 une fifo supplémentaire de N mots de 12 bits stockant la dernière ligne lissée est nécessaire.

En progressant de la solution 1 à la solution 3, on augmente le nombre d'opérateurs et un même calcul est effectué plusieurs fois (redondance) par contre le nombre des fifos de ligne diminue. On peut dresser un bilan des coûts des différentes solutions en terme d'opérateurs et ressources ainsi qu'en terme de surface [tableau 4].

Tableau 4. – Nombre d'opérateurs/nombre d'inverseurs équivalents

Solution	add	min et max	sélection 1 sur 3	registre	fifo 1 bit taille N	total
1	46/644	48/1296	24/312	92/1196	56/56.N	3448 + 56.N
2	138/1932	112/3024	24/312	196/2548	32/32.N	7816 + 32.N
3	184/2576	160/4320	48/624	288/3744	40/40.N	11264 + 40.N

On peut constater l'importance de la surface occupée par les fifos qui rend préférable la solution 2 dès que $N > 182$. La solution 3 devient meilleure que la solution 2 pour $N > 862$. Le fait de placer les fifos entre des blocs de calcul a deux inconvénients : l'augmentation de la largeur de celles-ci pour éviter les problèmes de troncature (exemple : calcul de la somme) et l'augmentation de leur nombre lorsque le stockage de plusieurs résultats intermédiaires est nécessaire (exemple : étendue et somme).

Remarquons que même si nous n'avions conservé que la moyenne sur 8 bits à la place de la somme sur 12 bits, la solution 1 aurait été plus coûteuse pour $N > 273$. Le calcul des coûts des solutions 2 et 3 est pessimiste. Il est en effet possible de réduire le nombre des opérateurs affectés aux calculs des blocs B1 puisque certains calculs internes à ceux-ci sont redondants entre blocs.

Par soucis de simplicité de la discussion, nous avons négligé les opérateurs de gestion des mémoires (calcul d'adresse) qui permettent d'appeler celles-ci fifos. Ces opérateurs n'influent pas sur la sélection d'une solution puisque un seul générateur d'adresse est nécessaire quelle que soit celle-ci. Le choix de la solution optimale en surface n'est pas remis en cause, si on souhaite placer les fifos à l'extérieur du circuit. Outre une meilleure adéquation aux circuits fifos existant [2] les plots d'entrées sorties nécessaires aux données sont réduits de la même manière [tableau 5]. Pour les solutions 1 et 2, on a fait l'hypothèse qu'on ne sort que le pixel lissé non tronqué alors que pour la troisième solution on ne sort que le gradient sur 12 bits.

L'implantation des fifos sur le circuit est préférable. Pour la solution 2 par exemple, la surface du circuit n'est réduite que de 4000 Ie soit 17% si on place les fifos à l'extérieur, car il faut tenir compte de l'augmentation du nombre de plots d'entrées, de sorties et d'alimentation qui en résulte. D'autre part il faut ajouter un circuit «fifo».

Tableau 5. – Nombre de plots pour les entrées et sorties des données

Solution	fifos externes	fifos internes
1	96	20
2	52	20
3	60	20

5. Design : diastolique ou systolique?

Quelle que soit la solution retenue parmi celles présentées, il existe un chemin combinatoire entre l'entrée des pixels et la sortie qui définit le chemin critique (temps de propagation le plus long). Ce chemin critique est indépendant de la solution puisqu'il traverse toujours un bloc B1 et deux blocs B2.

Si le temps de propagation est supérieur à la période d'échantillonnage des pixels d'entrée (100 ns) il sera nécessaire d'introduire des registres «pipeline» sur ce chemin.

Notons que l'association de deux opérations combinatoires A et B piplelinées, de temps de traversée respectifs t_A et t_B, conduit à une période minimale : $T_{\min} = \max(t_A, t_B)$. Par contre, sans pipeline on a : $T_{\min} \leqslant (t_A + t_B)$. Nous allons montrer qu'un choix judicieux des structures des opérateurs permet d'obtenir $T_{\min}$ bien inférieur à $t_A + t_B$ ce qui supprime dans certains cas l'intérêt du pipeline.

COMBINAISON D'ADDITIONS

Intéressons-nous à la combinaison des quatre additions du bloc B1 [figure 12]. Nous considérons des additions en ripple carry et supposons pour simplifier que pour un additionneur 1 bit tous les temps de propagation entre entrées et sorties sont égaux à «Tadd». Alors le temps pour obtenir le résultat en sortie d'un additionneur de n bits est : n Tadd. Le temps de traversée des quatre additionneurs de la figure 12 n'est que de 13 Tadd et non pas de 42 Tadd comme la simple addition des temps de traversée de chaque opérateur le laisserait supposer.

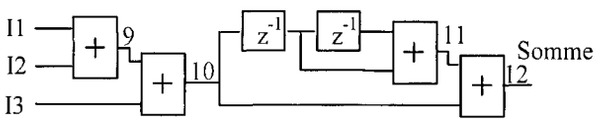

Figure 12. – Cascade d'additionneurs du bloc B1.

L'ajout d'un registre mémorisant le résultat complet de chaque addition, qu'on peut qualifier de structure *pipeline au niveau mot* ne conduirait au mieux qu'à une période d'échantillonnage de 11 Tadds. L'emploi de mécanismes de retenues anticipées compliquant la structure permettrait d'améliorer légèrement la

rapidité. Seule, la réalisation d'un *pipeline au niveau bit* isolant par des registres des additionneurs à sauvegarde de retenue (CSA) améliorait considérablement les performances. La période d'échantillonnage deviendrait de l'ordre de Tadd. On aboutirait alors aux structures utilisés dans les multiplieurs rapides [10]. L'ajout systématique de registres pipeline entre opérateurs conduit aux architectures dites systoliques (phase de contraction rapide du cœur).

Si ces structures sont optimales en terme de rapidité, l'ajout de registres a deux inconvénients non négligeables : l'augmentation de surface et l'existence d'un pic de consommation dans le voisinage temporel du front de l'horloge de chargement des registres. Par contre, un avantage de cette approche est qu'elle est bien formalisée [11].

Par opposition, on pourrait qualifier les structures combinatoires où on cherche à profiter du décalage temporel introduit dans la première couche d'opérateur : d'*architecture diastolique* (détente des données à travers le réseau combinatoire).

Les propagations entre les cellules binaires qui constituent un opérateur font que le résultat s'établit temporellement dans un sens (des poids faibles vers les poids forts ou inversement). Si dans une suite d'opérations en cascade, les propagations s'effectuent dans le même sens, on profite dans les couches d'opérateurs successifs du premier décalage temporel introduit. Par contre, la combinaison de deux opérateurs ayant des propagations en sens contraire conduit à sommer les temps de propagation. Il en va de même si l'opération précédente ne fournit un résultat qu'après propagation à travers le dernier opérateur binaire.

En résumé, si la contrainte temporelle est très forte, le pipeline au niveau bit s'impose. Sinon, on cherchera à exploiter au mieux les propagations dans le réseau combinatoire; le pipeline au niveau mot n'étant introduit qu'*a posteriori* si nécessaire.

Nous allons illustrer notre propos par le choix de structure de l'opérateur de recherche de maximum et considérerons l'influence de ce choix sur la rapidité et le nombre d'étage de pipeline de la structure du filtre Nagmod. Pour comparer les structures on se référera à la durée de traversée d'un Nand 2 entrées notée Tn.

OPÉRATEURS DE RECHERCHE DE MAXIMUM

Une première méthode du calcul du maximum de A et B, que nous appellerons M1, consiste à sélectionner A ou B en fonction du signe du résultat de la soustraction de B à A [figure 13].

Dans ce cas, la sortie ne devient valide qu'après calcul du signe soit après propagation de la retenue à travers tout le soustracteur puis traversée du multiplexeur. Sur 8 bits, ce temps vaut approximativement 19 Tn.

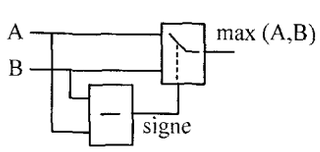

Figure 13. – Recherche du maximum de A et B par calcul du signe de la différence A-B.

Une deuxième méthode : M2 permet de calculer le maximum d'un ensemble de nombres (mots) avec un seul opérateur. Cette méthode exploite l'algorithme suivant :

B : nombre de bits par mot, b : indice bit ($b = 1$: LSB; $b = B$: MSB);

M : nombre de mots, m : indice mot;

$bit_{b,m}$: bit b du mot m;

$mask_{b,m}$: à l'état logique 1 indique que le mot m est non masqué au rang b;

max_b : bit b du maximum;

Les opérateurs somme et produit sont ceux de la logique booléenne.

$\forall m \quad mask_{B,m} = 1$

Pour b de B a 1

$$max_b = \sum_{m=1}^{M} (bit_{b,m} \cdot mask_{b,m})$$

$$mask_{b-1,m} = mask_{b,m}(bit_{b,m} + \overline{max_b})$$

FinPour

Un exemple est donné [figure 14] pour 5 mots de 4 bits.

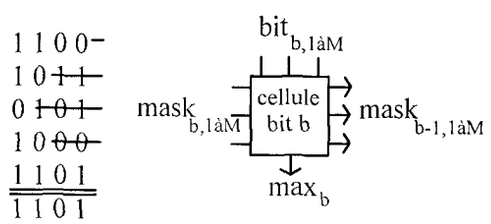

Figure 14. – Recherche de maximum par suppression progressive de mots.

On obtient ainsi une structure itérative où les informations $mask_{b,m}$ se propagent dans le sens des poids forts vers les poids faibles. Pour 1 bit, les temps de calcul sont les suivants :

$$mask_{b,m} \rightarrow \max_b : 2 \text{ Tn}$$

$$mask_{b,m} \rightarrow mask_{b-1,m} : 4 \text{ Tn}$$

Le calcul d'un max sur 8 bits prend donc 30 Tn. Mais cette structure est capable dans le même temps de traiter plusieurs mots, ce qui n'est pas le cas de la structure M1. D'autre part, dès que le bit de sortie $\max_b$ est disponible le calcul suivant exploitant $\max_b$ peut être entrepris. Ainsi pour le calcul du maximum d'un voisinage 3×3 dans le bloc B1, la méthode M1 conduit à 57 Tn et M2 à 34 Tn. Il serait bien sûr possible d'améliorer M1 en accélérant les calculs de signe (retenue anticipée). La méthode M1 atteint alors, pour une surface de silicium plus grande, les performances de la deuxième méthode.

M2 se prête naturellement à une réalisation pipeline au niveau bit dans le cas d'une contrainte temporelle plus forte et à une structure série plus économique dans le cas contraire. On vérifiera aisément que cela reste possible avec M1 mais au prix d'un nombre de

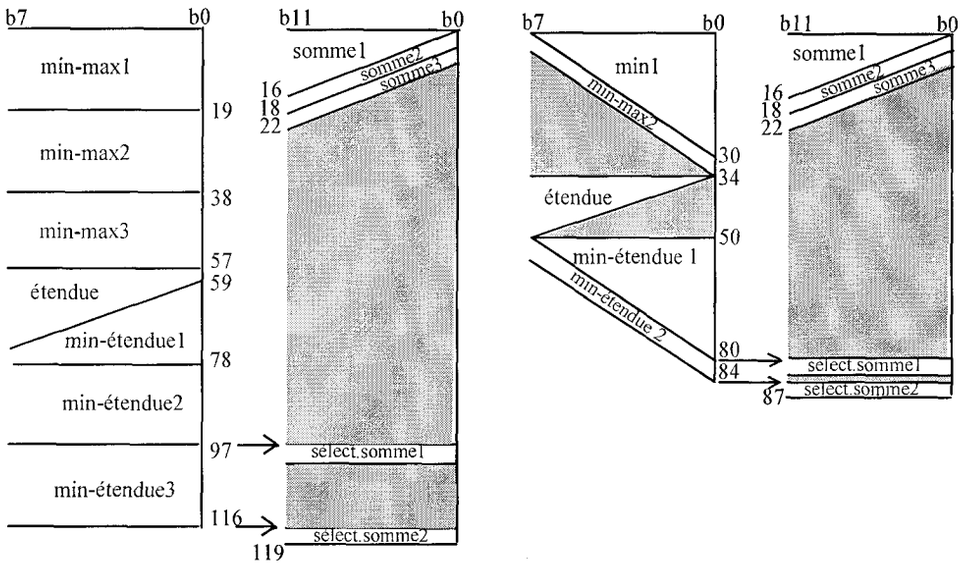

Figure 15. – Temps de traitement du pixel lissé indiqué en multiple de Tn : (temps de traversée d'un nand 2 entrées) (a) : calcul de min-max avec M1, (b) : calcul de min-max avec M2.

registres et d'un temps de latence nettement plus importants. Pour la méthode M1 [figure 15a] et pour la méthode M2 [figure 15b], on montre le calcul du temps d'exécution complet de l'opérateur de Nagao. Les zones en grisé correspondent aux durées pendant lesquelles le résultat est stable mais pas exploité. On constate l'absence de contrainte temporelle sur le calcul de somme. En CMOS $1,5\,\mu$m : Tn $\approx 1,8$ ns. Un seul niveau de pipeline après calcul de l'étendue est nécessaire avec la méthode M2 contre deux niveaux avec M1.

6. Résultats

Pour valider notre architecture, une version correspondant à la solution n° 2 incluant les mémoires de ligne a été réalisée en CMOS 1,5 μm. Ce circuit contient l'équivalent de 30 000 portes et occupe une surface de 47 mm^2. Il fournit en sortie le pixel lissé sous les formes : somme (sur 12 bits) et moyenne (sur 8 bits).

Les simulations ont montré que le temps de propagation le long du chemin critique était au maximum de 90 ns, ce qui est compatible avec un fonctionnement à une fréquence pixel de 10 Mhz. L'ensemble des vecteurs de test que nous avons défini donne un taux de couverture de test de 96%.

Actuellement ce circuit, mis en œuvre dans une chaîne de segmentation temps réel fonctionne parfaitement.

7. Conclusion

Différentes variantes de l'algorithme initial de Nagao ont été comparées, montrant que le choix de voisinages 3×3 réguliers, associé à un critère d'homogénéité du type étendue, offrait un bon compromis performance – complexité. L'influence de la troncature sur la qualité de détection des contours a été étudiée. L'architecture du filtre *Nagmod* a été discutée en terme de surface et rapidité, ce qui a montré l'importance de coût des surfaces mémoires sur la réalisation. Pour le design, l'exploitation du pipeline, liée à la structure fine des opérateurs (notamment de recherche de maximum), a été critiquée. Il faut noter que l'architecture retenue, se prête naturellement à une structure pipeline au niveau bit, ce qui permettrait d'étendre l'utilisation de celle-ci à des fréquences pixels supérieures à 100 Mhz.

BIBLIOGRAPHIE

[1] C. JOANBLANQ, P. SENN, M.J. COLAITIS. A 54 Mhz CMOS programmable Video Signal Processor for HDTV Applications, IEEE journal of Solid state circuits, vol. 25, n° 3, juin 1990.

[2] LSI lOGIC DIGITAL SIGNAL PROCESSING DATABOOK. Tour Chenonceau, 204 rond point du pont de Sèvres, 92516 Boulogne, France.

[3] A. ROSENFELD. Connectivity in digital pictures, Journal of ACM, vol. n° 17, p. 146-160, janvier 1970.

[4] J.F. QUESNE, D. DEMIGNY, J. DEVARS, J.P. COCQUEREZ. Architectures temps réel pour la fermeture des contours et l'étiquetage des régions, RFIA, 25-29 novembre 91, Villeurbanne.

[5] D. DEMIGNY, J.F. QUESNE, J. DEVARS. Boundary closing with asynchronous cellular automata, Computer Architecture for Machine Perception (CAMP), 16-18 décembre 91, Paris.

[6] F. TOMITA, S. TSUJI. Extraction of multiple regions by smoothing in selected neighborhoods, IEEE trans. System, Man and Cybernetics SMC-7, p. 107-109, 1977.

[7] M. NAGAO, T. MATSUYAMA. Edge preserving smoothing, Computer graphics and image processing, vol. 9, p. 394-407, 1979.

[8] J.P. COCQUEREZ, J. DEVARS. Détection de contours dans les images aériennes : nouveaux opérateurs, Traitement du signal, vol. 2, n° 1, p. 45-65, 1985.

[9] J.P. COCQUEREZ, S. PHILIPP. Rapport segmentation, Greco GDR 134 GT8, ENSEA, Cergy Pontoise.

[10] M. HATAMIAN, G.L. CASH. High speed Signal Processing, Pipelining and VLSI, ICASSP 86, Tokyo.

[11] S.Y. KUNG. On supercomputing with systolic/wavefront array processors, IEEE Proceedings, vol. 72, n° 7, p. 867, 884, 1984.

[12] A. PIRSON, D. DAVID, J.L. JACQUOT, T. COURT. Formalisation en Occam sur un réseau de transputers d'un processeur systolique de traitements d'images, Traitement du signal, vol. 7, n° 1, p. 41 à 51, 1990.

[13] J.R. FRAM, E.S. DEUTSCH. On the quantitative study of the orientation bias of some edge detection schemes, IEEE transaction on computers, C27, p. 205-213, 1978.

CURRICULUM VITAE

Didier DEMIGNY

Ses activités sont dirigées vers l'architecture des machines d'exécution. Après avoir mené une thèse dans le groupe de micro électronique à l'Institut d'Electronique Fondamentale (Paris XI) portant sur la conception d'un processeur RISC dédié aux applications en Intelligence Artificielle, il intègre l'Equipe Traitement des Images et du Signal (ETIS) à l'ENSEA en 89. Depuis cette date, il dirige l'axe architecture, axe orienté vers la conception de systèmes temps réel de traitement d'images.

Jean DEVARS

Jean Devars est professeur des Universités. En poste à l'Ecole Nationale Supérieure de l'Electronique et de ses Applications (ENSEA) de 85 à 92, il exerce maintenant à l'Université Pierre et Marie Curie (UP6). Ses activités de recherche sont orientées vers la vision par ordinateur et plus particulièrement vers les traitements bas niveaux et l'interface algorithme-architecture.

Lounis KESSAL

Après avoir mené une thèse dans le groupe de micro électronique à l'Institut d'Electronique Fondamentale (Paris XI) portant sur l'implantation d'un interprète LISP sur processeur RISC dédié aux applications en Intelligence Artificielle, il intègre en tant que maître de conférences l'axe architecture de l'Equipe Traitement des Images et du Signal à l'ENSEA en 90. Son principal pôle d'intérêt est la conception de circuits précaractérisés pour la segmentation temps réel des images.

Jean François QUESNE

Jean François Quesne a mené au sein de l'équipe (ETIS) une thèse portant sur les architectures dataflow pour le traitement temps réel des images. Il s'occupe actuellement de parallélisation d'algorithmes sur réseaux de transputers.

Manuscrit reçu le 2 décembre 1992