Reconstruction morphologique sur architecture parallèle à grain fin

Christophe Laurent [1,2] **et Christian Bouville** [1]

[1] CNET/DSM - 4, Rue du Clos Courtel - 35 512 Cesson Sévigné Cedex - FRANCE
[2] LaBRI URA CNRS 1304 - 351, Cours de la Libération - 33 405 Talence Cedex - FRANCE

RÉSUMÉ

Récemment, les opérateurs morphologiques connexes ont suscité un grand intérêt dans le domaine du traitement d'image de part leur efficacité dans les applications où l'information de contour est fondamentale (reconnaissance de formes, segmentation ...). Cependant, la quantité de calcul nécessitée par ces tranformations demande un traitement parallèle afin d'en améliorer les performances. Dans ce papier, nous nous intéressons à la parallélisation des opérateurs connexes de type ouverture/fermeture par reconstruction, en se basant sur le multiprocesseur single-chip MVP (TMS320C80). Nous présentons les difficultés liées à la projection d'algorithmes séquentiels performants sur une architecture parallèle à grain fin de type MVP et nous validons les solutions apportées en expérimentant notre implémentation sur plusieurs images test réelles avec différentes complexités de reconstruction.

ABSTRACT

Recently, morphological connected operators have attracted a large amount of research because of their efficiency in all applications for which contour information is fundamental (pattern recognition, image segmentation ...). However, these transformations require a large amount of computation and parallel processing is a necessity in order to improve performances. In this paper, we are interested in the parallelisation of connected operators called filters by reconstruction based on the single-chip multiprocessor MVP (TMS320C80). We present the projection of efficient sequential implementation on this kind of architecture and we validate our implementation on several real test images with different reconstruction complexities.

1 Introduction

Les opérateurs morphologiques connexes [5] ont souvent prouvé leur efficacité dans les applications où l'information de contour est primordiale (segmentation d'image, reconnaissance de forme ...). En effet, ces opérateurs ont la propriété fondamentale de simplifier une image (suivant un critère prédéfini [6]) tout en préservant parfaitement les contours des objets non effacés par la simplification. Cependant, malgré leur efficacité, les opérateurs connexes demandent une quantité importante de calcul, de part leur nature itérative, séquentielle et irrégulière, et afin d'obtenir de bonnes performances, un support d'exécution parallèle est essentiel. Dans ce papier, nous nous intéressons à la parallélisation d'opérateurs connexes de type *filtres par reconstruction* qui utilisent un critère de simplification "orienté taille" et qui sont largement utilisés dans les applications de segmentation d'image [4]. L'architecture parallèle utilisée sera le multiprocesseur single-chip MVP (TMS320C80) [7] offrant un support d'exécution puissant pour les applications de traitement d'image grâce à son architecture à mémoire partagée.
Cet article présente en détail les solutions parallèles apportées au problème du filtrage morphologique par reconstruction sur ce type d'architecture et montre que les gains obtenus par rapports aux architectures séquentielles classiques sont très importants. D'autre part, les mesures expérimentales ont montré que l'approche proposée ne souffre pas de la complexité de la reconstruction, contrairement aux approches séquentielles, ce qui lui permet d'être extensible.
Cet article est organisé de la façon suivante : nous rappelons, en section 2, la notion d'opérateur connexe et de filtre par reconstruction. La section 3 donne une vue générale du multiprocesseur MVP et propose une méthodologie de programmation de ce type d'architecture. La parallélisation du processus de reconstruction est détaillée dans la section 4 et les résultats numériques sont présentés en section 5. Pour clôturer cet article, nous concluons en section 6.

2 Contexte théorique

Comme tout opérateur connexe [5], les filtres morphologiques par reconstruction peuvent être décomposés en deux étapes distinctes : une étape de *sélection* permettant d'éliminer tous les composants de l'image plus petits qu'une taille limite (donnée par la taille de l'élément structurant), et une étape de *décision* permettant de restaurer les contours des éléments non totalement effacés par la sélection.
Deux filtres par reconstruction sont couramment utilisés dans les applications : *l'ouverture par reconstruction* (notée $\gamma^{(rec)}$) et la *fermeture par reconstruction* (notée $\varphi^{(rec)}$). Ces filtres utilisent la notion de transformations géodésiques et sont définis par :

$$\gamma^{(rec)}(f,r) = \delta^{(\infty)}(f,r) = \delta^{(1)}(\delta^{(1)}(\ldots(\delta^{(1)}(f,r))\ldots))$$

$$\varphi^{(rec)}(f,r) = \varepsilon^{(\infty)}(f,r) = \varepsilon^{(1)}(\varepsilon^{(1)}(\ldots(\varepsilon^{(1)}(f,r))\ldots))$$

Pour ces filtres, le signal f est appelé *signal marqueur* et le signal r est appelé *signal masque* (signal de référence). La transformation $\delta^{(1)}(f,r)$ (resp. $\varepsilon^{(1)}(f,r)$) définit

la dilatation (resp. l'érosion) géodésique unitaire et est définie par $\delta^{(1)}(f,r) = Min\{\delta_1(f), r\}$ (resp. $\varepsilon^{(1)}(f,r) = Max\{\varepsilon_1(f), r\}$) où l'opérateur δ_n (resp. ε_n) désigne la dilatation (resp. l'érosion) morphologique avec un élément structurant de taille n.
Dans la pratique, le signal f correspond à la sortie du processus de sélection de l'opérateur connexe et le signal r correspond à l'image originale. Généralement, f est calculé à partir d'opérateurs de type érosion/ouverture (resp. dilatation/fermeture) dans le cas d'ouverture (resp. de fermeture) par reconstruction.
Il est clair que le goulot d'étranglement, en termes de quantité de calcul, se trouve au niveau du processus de décision de part sa nature itérative et séquentielle, et afin d'en améliorer les performances, on se propose d'etudier sa parallélisation en utilisant le multiprocesseur single-chip MVP (*Multimedia Video Processor*). Il est à noter que la parallélisation du processus de sélection sur ce type d'architecture peut être touvée dans [2].

3 Le multiprocesseur MVP

Le MVP [7] est un multiprocesseur "single chip" puissant et spécialement conçu pour le traitement du signal. Il est composé de cinq processeurs programmables connectés à une mémoire partagée via un réseau crossbar (figure 1). Quatre de ces processeurs sont identiques et de type DSP (*Digital Signal Processor*) spécialement conçus pour le calcul intensif. Ces processeurs seront appelés par la suite PP (*Parallel Processor*). Le dernier processeur (appelé MP pour *Master Processor*) est de type RISC 32 bits. Chacun de ces processeurs possède sa propre unité de controle, ce qui permet d'obtenir un comportement totalement asynchrone de type MIMD (*Multiple Instruction Multiple Data*).

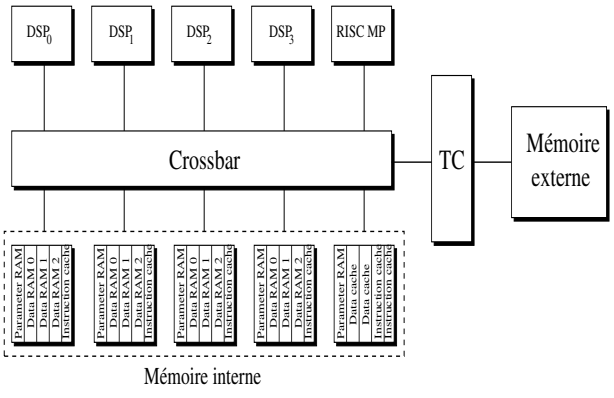


FIG. 1 — Architecture du multiprocesseur MVP

Sur la figure 1, on peut distinguer deux types de mémoires : une mémoire interne de 50 Ko avec un accés mono-cycle et une importante mémoire externe à débit plus lent. Les transferts de données entre ces deux mémoires passent par l'utilisation du TC (*Transfer Controler*) qui agit comme une interface mémoire intelligente en traitant des transferts rapides de paquets à partir de composants adressables.
La mémoire interne est divisée en blocs mémoire (chacun de taille 10 Ko) qui sont distribués de façon "logique" (dû à l'architecture à mémoire partagée) aux DSPs. Ainsi, chaque DSP possède un bloc de mémoire interne qui est à son tour divisé en cinq bancs mémoire de taille 2 Ko : trois de ces bancs sont dédiés au traitement des données ($DRAM_i$, $0 \leqslant i \leqslant 3$), le banc "parameter RAM" est partiellement utilisé pour stocker le vecteur d'interruption et les paramètres de programmation du TC ; le reste (environ 1,5 Ko) étant laissé à l'utilisateur et à la pile d'appels, et le dernier banc est utilisé comme cache d'intruction.

Afin d'obtenir de bonnes performances sur ce type d'architecture, la conception d'algorithmes parallèles doit appliquer une méthodologie de programmation stricte et précise. Ainsi, la mémoire doit être gérée avec finesse afin de réduire le nombre d'accés en mémoire externe. Pour ce faire, toutes les données nécessaires au traitement doivent être transférées en mémoire interne avant le début du traitement. De plus, une utilisation intelligente du TC est recquise afin de recouvrir les transferts mémoire (effectués par le TC) et le traitement (effectué par les DSPs). Une description plus précise de la méthodologie de programmation du MVP peut être trouvée dans [7, 2].

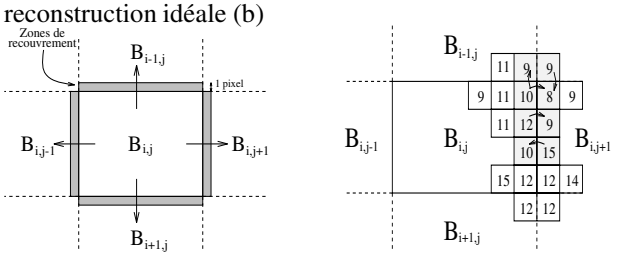
4 Processus de reconstruction morphologique sur MVP

4.1 Parallélisation de l'algorithme

Dans le cas séquentiel, les implémentations les plus efficaces du filtrage morphologique par reconstruction reposent sur l'utilisation de files d'attente de pixels [3] représentées par des structures FIFO (*First In First Out*). Cependant, l'utilisation de telles structures entraîne une irrégularité dans le comportement des algorithmes parallèles car l'évolution spatiale des FIFOs est diffuse et non concentrée en une région de l'espace. Le problème qui se pose donc est d'étudier l'implémentation d'algorithmes à base de FIFO sur une architecture parallèle à grain fin de type MVP.
Comme nous l'avons mentionné en section 3, la mémoire interne du MVP est de taille limitée. De ce fait, l'image masque I et l'image marqueur J sont initialement stockées en mémoire externe et afin de trouver une solution efficace au problème de reconstruction sur MVP, nous devons prendre en compte à la fois les avantages apportés par les algorithmes à base de FIFO et la granularité de l'architecture considérée. Pour ce faire, chaque image I et J est d'abord divisée en "sous images" I_i et J_i ($0 \leqslant i \leqslant 3$) distribuées aux PPs. Chaque DSP sera alors responsable de la reconstruction d'une sous image. Puis, chaque sous image est à son tour divisée en un ensemble de blocs $\mathcal{B}_{k,l}$ de taille suffisamment faible pour pouvoir être transférés en mémoire interne.
Ainsi, chaque DSP aura pour tâche de reconstruire l'ensemble des blocs localisés dans son image locale. Cependant, la reconstruction bloc par bloc de façon indépendante génère des artifacts sur l'image reconstruite (figure 2) dûs à l'évolution spatiale non régulière de la structure FIFO et à la non prise en

(a) (b)

FIG. 2 — reconstruction des blocs de façon indépendante (a) et reconstruction idéale (b)



(a) (b)

FIG. 3 — Structure interne d'un bloc avec les zones de recouvrement (a) propagation inter-blocs (b) : les pixels hachurés représentent la localisation de la propagation pour une reconstruction par dilatation et les flèches représentent le sens de la propagation

compte de cette évolution en reconstruisant les blocs indépendamment les uns des autres.

En effet, l'utilisation d'une structure FIFO ne garantit pas le principe de localité spatiale contraint par l'utilisation du MVP et, de ce fait, la reconstruction d'un bloc peut générer des propagations sur les blocs voisins. Afin de prendre en compte ce phénomène de propagation, chaque bloc se voit attribuer des zones de recouvrement sur ses blocs voisins (figure 3(a)). Grâce à ces zones, l'algorithme de reconstruction pourra facilement détecter les propagations inter-blocs (figure 3(b)).

On obtient ainsi un algorithme parallèle de reconstruction orienté bloc dans lequel les blocs sont reconstruits en utilisant un algorithme puissant à base de FIFO. Une description détaillée de l'algorithme peut être trouvée dans [1].

4.2 Adéquation algorithme-architecture

4.2.1 Gestion de la mémoire interne

Afin de réduire le nombre d'accés en mémoire externe, toutes les données manipulées par l'algorithme de reconstruction doivent être transférées en mémoire interne. Ces données sont représentées par le bloc masque et le bloc marqueur. Ainsi, avant le début des traitements, chaque tâche élémentaire (consistant en la reconstruction d'un bloc) devra posséder en mémoire interne ces deux blocs. Cependant, afin d'optimiser le temps de traitement des DSPs, les transferts mémoire et la gestion des propagations inter-blocs sont laissés à la charge du MP qui doit donc transférer les deux blocs d'image avant que ceux-ci soient utilisés par les DSPs. D'autre part, afin d'éviter aux DSPs d'attendre la fin d'un transfert et pour permettre un recouvrement entre les transferts mémoire (amorcés par le MP et effectués par le TC) et le traitement (effectué par les DSPs),

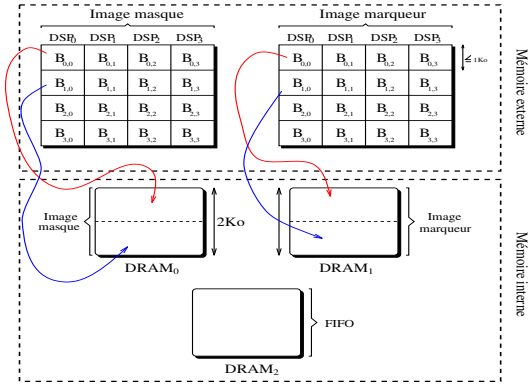


FIG. 4 — Architecture logicielle de l'algorithme parallèle de reconstruction orienté bloc sur MVP

le MP calcule une réorganisation complète de la mémoire interne attachée aux PPs détaillée ci-après.
Les bancs mémoire $DRAM_0$ et $DRAM_1$ sont utilisés pour stocker respectivement les blocs masques et les blocs marqueurs. Le recouvrement transfert mémoire/traitement est obtenu en divisant chacun de ces bancs en deux demi-bancs (de taille 1Ko). Ainsi, la taille d'un bloc correspond à un ensemble de pixels de taille maximale pouvant être stocké dans un demi-banc mémoire.
Le dernier banc $DRAM_2$ est utilisé pour stocker la structure FIFO. On obtient alors une architecture logicielle comme illustré sur la figure 4.

4.2.2 Communications interprocesseurs

Afin que le MP détecte la fin de la reconstruction d'un bloc ainsi que les propagations inter-blocs, des communications entre le MP et les DSPs doivent être insérées.
Sur une architecture de type MVP, les communications interprocesseurs sont implémentées via la mémoire partagée en utilisant un mécanisme d'interruption matérielle interprocesseurs [7].
Le MP peut recevoir deux types de messages provenant d'un DSP : lorsqu'un DSP détecte une propagation inter-bloc, il émet une interruption en direction du MP afin qu'il sauvegarde l'information de propagation. De la même façon, lorsqu'un DSP termine la reconstruction d'un bloc, il en informe le MP afin qu'il transfère un nouveau bloc dans le demi-banc mémoire devenu libre. On obtient ainsi une architecture de type client/serveur dans laquelle le MP fournit une liste de tâches élémentaires exécutées par les DSPs.

4.2.3 Optimisation des communications

Afin de minimiser la quantité de communication circulant sur le réseau, une technique d'empaquettage de message peut être utilisée. Pour ce faire, chaque DSP utilise deux mémoires tampons T_1 et T_2 de taille n ($n \geqslant 1$) permettant de stocker les informations de propagations inter-blocs. Ainsi, lorsqu'un des tampons est plein, le DSP émet une interruption vers le MP lui indiquant une propagation. Pendant que le MP traite le message, le DSP peut continuer à stocker les nouvelles

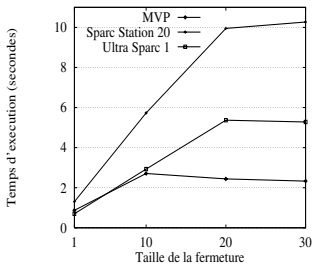


FIG. 5 — Temps d'exécution de la reconstruction orientée bloc sur l'image *Mobile & Calendar* et comparatif

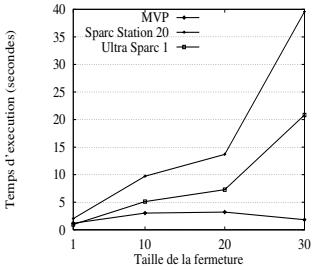


FIG. 6 — Temps d'exécution de la reconstruction orientée bloc sur l'image *Table tennis* et comparatif

propagations dans le tampon libre. Ce mécanisme permet donc de réduire le nombre d'interruptions des DSPs vers le MP puisque les messages sont émis uniquement lorsque l'un des tampons est plein et d'éviter l'attente des DSPs après la prise en compte des messages par le MP.

4.2.4 Equilibrage de charge

Les techniques mises en œuvre jusqu'à présent ne garantissent pas une distribution équitable de la charge sur l'ensemble des processeurs puisque certains processeurs peuvent terminer la reconstruction avant d'autres. Pour éviter cette situation, on suggère d'utiliser le processeur MP pour permettre un équilibrage dynamique de charge. Ainsi, lorsque le MP reçoit un message de fin de reconstruction de la part d'un processeur PP_i et qu'il n'y a plus de bloc à reconstruire dans la sous-image I_i, le MP peut rechercher un bloc à reconstruire dans une sous image I_j ($j \neq i$). De ce fait, le processeur PP_i ne restera pas inactif.

5 Résultats numériques

Cette partie est dédiée à l'étude du comportement de notre algorithme parallèle de reconstruction à partir des images test bien connues *Mobile & Calendar* et *Table tennis*. Ces images sont au format CIF ($352\times288\times8$) et la plateforme MVP utilisée est cadencée à 30 MHz. D'autre part, tous les algorithmes ont été implémentés en langage C.
Afin de valider notre approche parallèle, il nous a semblé intéressant d'expérimenter notre implémentation à partir de plusieurs signaux marqueurs et de comparer les temps obtenus sur architecture MVP avec les temps obtenus sur des architectures séquentielles puissantes (*Sparc Station 20* et *Ultra Sparc 1*). Dans ce papier, les signaux marqueurs considérés sont de type fermeture morphologique et le filtre testé est donc une fermeture par reconstruction. D'autre mesures expérimentales sur d'autres images test et utilisant d'autres signaux marqueurs peuvent être trouvées dans [1] et l'impact de l'empaquettage de messages (section 4.2.3) sur les performances est discuté dans [1]. Les figures 5 et 6 présentent ce comparatif pour toutes les images test et pour tous les signaux marqueurs considérés dans cet article.

On peut remarquer que dans tous les cas, l'approche parallèle orientée bloc donne les meilleurs résultats. D'autre part, notre approche est extensible puisqu'elle ne souffre pas de la complexité de la reconstruction, contrairement aux approches séquentielles.

6 Conclusion

Dans ce papier, nous avons présenté une approche parallèle à grain fin pour le filtrage morphologique par reconstruction en se basant sur le multiprocesseur single chip MVP. La projection du problème de reconstruction morphologique sur ce type d'architecture nous a d'abord amené à réduire la granularité du calcul afin de prendre en compte les contraintes matérielles du processeur. D'autre part, nous avions fixé une contrainte algorithmique supplémentaire en se basant sur les approches de reconstruction utilisant les files d'attente de pixels. Toutes ces contraintes nous ont amené à proposer une approche parallèle de reconstruction orientée bloc qui a prouvé son efficacité, comparée aux approches séquentielles.

Références

[1] C. Laurent. Une approche parallèle à grain fin pour le filtrage morphologique par reconstruction. Technical Report 1169-97, LaBRI-Université de Bordeaux I, Mars 1997.

[2] C. Laurent and C. Bouville. Parallel Implementation of Greyscale Morphological Operators on MVP. In *VCIP'97*, volume 3024, pages 502–513, San Jose, Février 1997.

[3] L. Vincent. Morphological Grayscale Reconstruction in Image Analysis : Applications and Efficient Algorithms. *IEEE Trans. on Image Proc.*, 2(2) :176–201, Avril 1993.

[4] P. Salembier and J. Serra. Morphological Multiscale Image Segmentation. In *VCIP'92*, volume 1818, pages 620–631, Boston, 1992. SPIE.

[5] P. Salembier and J. Serra. Flat Zones Filtering, Connected Operators and Filters by Reconstruction. *IEEE Trans. on Image Proc.*, 4(8) :1153–1160, Août 1995.

[6] P. Salembier. Multi-criterion segmentation for image coding. In *Int. Workshop on Math. Morphology and Its Applications to Signal Proc.*, pages 40–45, Barcelone, 1993.

[7] Texas Instrument. *TMS320C80 Multimedia Video Processor : Technical Brief*, 1994.