Une rétine électronique automate cellulaire

A cellular automata smart sensor

P. GARDA

IEF (CNRS, LA 22), Bat. n° 220, Université de Paris-Sud, 91405 ORSAY CEDEX.

Agrégé de Mathématiques, titulaire d'une thèse de troisième cycle en électronique, P. Garda est Chargé de Recherches à l'I.E.F. (U.R.A. 22). Ses centres d'intérêt portent sur la conception et l'utilisation d'algorithmes et d'architectures cellulaires et neuronales pour la vision.

A. REICHART

Section d'étude et de fabrication des Télécommunications, 4, rue du Professeur Zamenhoff, 92130 ISSY-LES-MOULINEAUX.

Ancien élève de l'école Polytechnique, titulaire d'un diplôme d'ingénieur à l'ENSTA, d'un DEA en Reconnaissance de forme (Paris-VI, 1985), et d'une thèse de doctorat (Paris-XI, 1988). A travaillé à l'ETCA de 1985 à 1988 au laboratoire Systèmes de Perception.

H. RODRIGUEZ

IEF (CNRS, LA 22), Bat. n° 220, Université de Paris-Sud, 91405 ORSAY CEDEX.

H. Rodriguez prépare une thèse de doctorat à l'I.E.F. sur l'intégration de rétines de grande dimension.

F. DEVOS

IEF (CNRS, LA 22), Bat. n° 220, Université de Paris-Sud, 91405 ORSAY CEDEX.

Professeur d'Université à l'Institut d'Electronique Fondamentale (I.E.F.). Il anime une équipe d'architecture et de conception de machines dédiées. Ses sujets d'intérêt sont centrés sur les procédures à parallélisme massif.

B. ZAVIDOVIQUE

ETCA/CREA/SP et IEF.

Ayant obtenu un doctorat ès sciences consacré à la vision des robots (Besançon), il est, depuis octobre 1981, conseiller de la DRET (ETCA) (*) pour les problèmes de Traitement d'Images Temps Réel et Robotique. Ses recherches sont principalement axées sur la conception de système de perception, l'impact de leur organisation interne sur leur efficacité externe, leur implantation effective, et par ce biais sur les méthodes (modernes) de programmation, d'architecture et d'intégration.

RÉSUMÉ

Les capteurs intelligents optiques numériques sont utiles dans un grand nombre d'applications et peuvent être considérés à terme comme l'aboutissement ultime de tout système de vision. Les tableaux de processeurs constituent une architecture massivement parallèle séduisante pour de tels capteurs, puisqu'ils sont bien adaptés aux contraintes de l'intégration à la demande et à l'organisation naturelle des images. De plus ils tirent aisément parti des progrès des densités d'intégration. Dans cet article nous décrivons les interactions algorithme-architecture que nécessite la conception d'une rétine électronique monolithique organisée en tableau de processeurs permettant une vision d'alerte autonome. Nous présentons d'abord nos motivations, nous décrivons les opérateurs que notre tableau de processeurs pourra appliquer aux images qu'il a acquises et nous montrons comment ils peuvent être utilisés pour faire une recherche de motifs par coïncidence binaire. Nous proposons de premiers résultats de traitement d'images multiniveaux codées par des images binaires en demi-teinte. Finalement nous décrivons l'architecture de la rétine.

MOTS CLÉS

Capteurs intelligents, automates cellulaires, tableaux de processeurs, traitement d'images binaires, reconnaissance de formes.

SUMMARY

Optical digital smart sensors are useful in a lot of applications and can be considered in the long run as the potential ultimate implementation of all optical vision systems. Mesh arrays are an appealing massively parallel architecture for such sensors as they fit full custom integration constraints as well as image natural organization. Moreover they take easily full advantage of VLSI density increases. In this paper we describe the interactions between algorithm and architecture involved in the design of a monolithic mesh array smart sensor which supports some autonomous alarm vision. We first present our motivations, we formalize the operators carried out by the array on binary pictures and we explain how they perform logical template matching. We present first results of the processing of grey level pictures thanks to their halftone representation. Finally we detail the sensor architecture.

KEY WORDS

Smart sensors, cellular automata, processor arrays, binary image processing, pattern recognition.

1. Introduction

L'œil constitue une référence mythique pour quiconque s'intéresse à la vision artificielle, et concevoir un œil artificiel est un défi attrayant et inaccessible. Or, les semi-conducteurs intégrés fournissent un moyen de fabriquer des machines constituées de myriades de cellules toutes identiques. Par ailleurs, le silicium est naturellement photosensible – néanmoins sa réponse spectrale n'est pas nécessairement adaptée à toutes les applications –. Ainsi, les technologies électroniques intégrées offrent-elles la possibilité de concrétiser des rétines artificielles.

D'autres motivations plus pratiques suscitent une telle recherche. Tout d'abord, un système fondé sur des composants intégrés spécifiques est particulièrement compact. Or la compacité d'un équipement électronique en facilite la diffusion, ne serait-ce qu'en en diminuant le coût. D'ailleurs une telle compacité est requise dans la plupart des applications embarquées. Qui plus est, un tel système est aussi très rapide, ce qui est important pour tout système de traitement d'images. Au total, ces deux caractéristiques ouvrent potentiellement de nombreux domaines d'applications à de telles rétines. On peut imaginer, à titre d'exemple, l'intérêt de l'usage courant d'un tel composant dans un lecteur optique pour aveugles.

Qu'est-ce qui constitue une rétine électronique? Il s'agit d'un circuit intégré monolithique qui rassemble dans une même puce des opérateurs d'acquisition, de codage, de traitement d'images, ou une partie de ceux-ci. Force est de constater que même les centaines de milliers de transistors qui peuvent être rassemblés sur une puce réalisée aujourd'hui ne sont rien face au matériel requis par l'ensemble des opérateurs que nous venons de mentionner. Aussi la conception de ce composant implique-t-elle des compromis dramatiques entre la complexité de l'architecture retenue, sa facilité de programmation, ses domaines d'applications, ses performances.

Différents compromis donnent lieu à différents schémas de rétines conçues pour différents types de tâches. Nous allons décrire ceux présentés à ce jour.

Premièrement, un imageur matriciel à CCD peut légitimement être considéré comme une rétine électronique, dont la fonction est réduite à l'acquisition de l'image. Deuxièmement, on peut imaginer d'intégrer au capteur un codeur analogique-numérique pour lui permettre d'assumer en plus les fonctions d'échantillonnage spatial et temporel et de numérisation. Cette fonction de codage peut d'ailleurs être développée plus largement: par exemple [Cole 83] décrit un circuit qui fournit la Transformée de Hadamard des images qu'il a acquises.

Troisièmement, on peut associer à un capteur optique un opérateur de traitement d'images, comme par exemple un détecteur de contours. Une telle rétine s'insère alors dans un système de vision qui en contrôle les paramètres et exploite les données qu'elle

lui fournit. Elle allège la charge de ce système d'une part en effectuant elle-même des calculs lourds, d'autre part en réduisant le débit d'informations entre le capteur et le système de perception qui l'exploite. Ainsi elle lui permet de se consacrer à des tâches plus «intelligentes», et accroît donc l'efficacité de l'ensemble du système. Elle s'inscrit alors dans la ligne des capteurs intelligents — dont les signaux d'entrée ne sont pas des images — ou des frontaux de système de traitement de l'information.

Quatrièmement, ces capteurs peuvent être conçus comme de très petits systèmes entièrement autonomes. Dans ce cas, ils sont spécialisés pour effectuer complètement une fonction unique précise. C'est le cas par exemple de la souris optique de [Lyon 81] qui est utilisée comme un organe de pointage pour les stations de travail. Néanmoins il faut noter que l'algorithme de détection du mouvement de cette souris est câblé dans le silicium, à tel point que même la texture de la surface sur laquelle elle se déplace y est figée. Les rétines conçues selon ce schéma présentent les mêmes intérêts que celles conçues selon le troisième schéma.

Nous nous intéressons à une rétine qui se distingue de celles réalisées à ce jour par ses plus grandes possibilités, quitte à ce que sa réalisation à très court terme soit plus difficile. Nous souhaitons qu'elle constitue un système autonome, à l'instar du quatrième schéma proposé, mais sans pour autant qu'une application soit câblée dans le silicium qui la constitue. Pour cela, nous souhaitons tout d'abord qu'elle soit programmable et constitue ainsi un petit calculateur spécialisé en traitement d'images. Par comparaison, les rétines que nous venons de décrire ne sont que paramétrables — par exemple par le choix d'un gain —. Plus précisément, nous souhaitons pouvoir programmer la rétine à partir des données des applications auxquelles elle est destinée. De cette manière, la possibilité d'une auto-adaptation de la rétine à son environnement est préservée ([Garda 85-1], [Garda 85-3]). Ceci suppose que nous puissions écrire des algorithmes d'apprentissage permettant de construire des opérateurs à partir des images caractéristiques d'une application.

Néanmoins il est difficile d'implanter à court terme dans une rétine un système bouclé qui prenne lui-même en compte les conséquences des actions qu'il a décidées. Nous sommes donc conduits à mettre en œuvre cette rétine principalement pour des applications de détection ou de classification, dont un cas intéressant est celui d'une vision d'alerte autonome. En effet, un système de perception complet a des temps de réponse relativement longs. On peut donc imaginer de le compléter par un petit système fruste et autonome, qui soit chargé d'émettre très rapidement un signal d'alarme dès lors qu'il a détecté un *objet* inquiétant. Il faut noter que ceci est beaucoup plus difficile que de détecter une *caractéristique* inquiétante de l'image [Zavidovique 87]. Au total nous demandons ainsi à notre rétine de pouvoir exécuter des algorithmes de décision et de reconnaissance, tout en restant très rapide.

Cette contrainte de vitesse de traitement de l'image nous conduit à privilégier *a priori* les possibilités d'exécution parallèle. Certes, la plupart des algorithmes de traitement d'images bas niveau peuvent être exécutés en parallèle. Par contre, les algorithmes de détection et de classification sont presque toujours conçus pour être exécutés sur des ordinateurs séquentiels, et leur parallélisation n'est pas immédiate. Or, le problème de l'interface entre les architectures parallèles iconiques et symboliques est essentiellement ouvert [Tanimoto 88], et il paraît particulièrement difficile à résoudre dans le contexte d'une intégration monolithique. La direction qui consiste à rejeter à l'extérieur de la rétine la partie responsable de la décision nous ramène aux troisième et quatrième schémas de rétines que nous avons mentionnés, et nous ne la suivrons donc pas.

Dans cet article, nous décrivons une expérience de réalisation des traitements de l'image et des opérations de décision qui les suivent avec les mêmes opérateurs exécutés par la même architecture parallèle. La description de l'étude est organisée comme suit. Dans le deuxième chapitre, nous décrivons le choix des opérateurs sur lesquels nous nous appuyons, appelés Traitements Combinatoires Locaux (TCL). Dans le troisième chapitre, nous montrons comment nous les utilisons pour construire une méthode fruste et parallèle de reconnaissance de formes. Dans le quatrième chapitre, nous montrons comment les méthodes du troisième chapitre peuvent être étendues aux images binaires en demi-teinte. Dans le cinquième chapitre, nous décrivons l'architecture de rétine électronique qui permet leur mise en œuvre.

2. Traitements combinatoires locaux

Dans ce chapitre, nous choisissons les opérateurs implantés sur la rétine, puis nous les définissons mathématiquement, enfin nous les situons par rapport aux opérateurs analogues de la morphologie mathématique.

2.1. Choix des opérateurs

D'une part, les options que nous avons prises dans l'introduction sont autant de contraintes que doivent satisfaire les opérateurs que nous allons utiliser.

D'autre part, le choix d'une réalisation immédiate introduit des bornes du fait des technologies électroniques disponibles. Le nombre de transistors que peut contenir une puce unique reste limité en regard de ce qu'exige l'implantation d'un calculateur parallèle, et ceci doit être pris en compte par une approche qui s'appuie sur des validations concrètes à court terme.

Une première restriction est donc de n'utiliser que des opérateurs booléens manipulant des images binaires, parce que les premiers exigent peu de matériel pour l'implantation des unités de calcul, et les seconds peu de matériel pour leur mémorisation. Les images binaires ont d'ailleurs été abondamment étudiées, du fait de leur plus grande simplicité. Néanmoins les outils les plus fins utilisés pour analyser leur structure et leur architecture ([Chassery 87], [Serra 82]) ont été développés sur des calculateurs séquentiels, sans prendre en considération quelque implantation paral-

lèle que ce soit. Une seconde restriction est donc de ne s'appuyer que sur une famille unique d'opérateurs, de manière à en réussir l'implantation sur un calculateur parallèle monolithique. En effet la manipulation d'un ensemble de types de données (tableaux, listes, graphes, ...) par une architecture unique la complique d'une manière inacceptable. Une troisième restriction consiste alors à demander que ces opérateurs soient homogènes avec les données qu'ils auront à traiter, de manière à faciliter la conception d'algorithmes d'apprentissage. Dans ce cas il est plausible que la rétine elle-même assume ces algorithmes.

Il faut alors considérer les conséquences du choix d'une famille d'opérateurs unique que nous venons de faire.

D'une part la philosophie d'un algorithme de décision s'appuyant sur une famille *unique* d'opérateurs va forcément être proche de celle des méthodes de corrélation. Un tel algorithme consistera en la comparaison, par l'intermédiaire de l'application d'opérateurs, du contenu de l'image à un catalogue d'images de référence.

D'autre part la mécanique la plus simple s'appuyant sur l'application d'opérateurs en est la composition. Ceci résulte simplement du fait que la seule action possible, si l'on ne dispose que d'opérateurs « nus », est de les appliquer les uns après les autres, ce qui se traduit, en termes algébriques, par leur composition. Donc les algorithmes d'apprentissage s'appuieront *a priori* sur la composition ou la décomposition des opérateurs pour leur construction à partir d'exemples.

Finalement, nous cherchons des opérateurs qui traitent des images binaires, pour lesquels la composition soit une mécanique adaptée, et qui se prêtent à une implantation parallèle. Nous avons choisi [Zavidovique 81] une partie des opérateurs géométriques classiques de traitement d'images binaires, que nous avons nommée TCL, qui a pour particularité d'être clairement caractérisée, d'être stable par composition, et de se prêter à une reconnaissance de formes fruste basée sur cette propriété.

2.2. Définition des opérateurs

L'objet de cet alinéa, assez mathématique, est de définir précisément les notions d'image binaire discrète et de TCL.

$\mathbf{Z}$ désigne l'ensemble des entiers relatifs. Les éléments de $\mathbf{Z}\times\mathbf{Z}$ sont appelés des sites. Ici l'addition désigne l'addition de points dans $\mathbf{Z}\times\mathbf{Z}$, c'est-à-dire que si $z_1=(x_1,y_1)$ et $z_2=(x_2,y_2)$ sont deux sites, $z=z_1+z_2$ est le site (x_1+x_2, y_1+y_2). Par extension, si z est un site et V une partie de $\mathbf{Z}\times\mathbf{Z}$, nous notons $z+\mathrm{V}$ l'ensemble des sites de la forme $z+v$, où v décrit V:

$$z+\mathrm{V}=\{z+v, v\in\mathrm{V}\}$$

Une image binaire est représentée par une application de $\mathbf{Z}\times\mathbf{Z}$ dans $\{0,1\}$. Nous nous appuyons implicitement sur un réseau à maille carrée.

Un Traitement Combinatoire Local (TCL) est défini suivant [Zavidovique 81] à l'aide de deux paramètres:

— un voisinage V, qui est une partie finie de $\mathbf{Z}\times\mathbf{Z}$; il désigne l'ensemble des sites qui peuvent servir au calcul du TCL;

— une fonction booléenne e, dont les variables binaires sont indicées par les sites qui appartiennent à V: $e((\mathrm{X}_v)_{v\in\mathrm{V}})$; elle peut donc être considérée comme une fonction $e:\{0,1\}^{\mathrm{V}}\to\{0,1\}$.

L'application du TCL t de paramètres (V, e) à l'image binaire I (considérée comme une application de $\mathbf{Z}\times\mathbf{Z}$ dans $\{0,1\}$) est une image binaire $t(\mathrm{I})$ (qui est elle-même une application de $\mathbf{Z}\times\mathbf{Z}$ dans $\{0,1\}$). Pour calculer la valeur de l'image binaire $t(\mathrm{I})$ au site z, on applique la fonction booléenne e au voisinage $(z+\mathrm{V})$ du site z dans l'image binaire $\mathrm{I}: t(\mathrm{I})(z)=e((\mathrm{I}(z+v))_{v\in\mathrm{V}})$, ce qui est illustré figure 1.

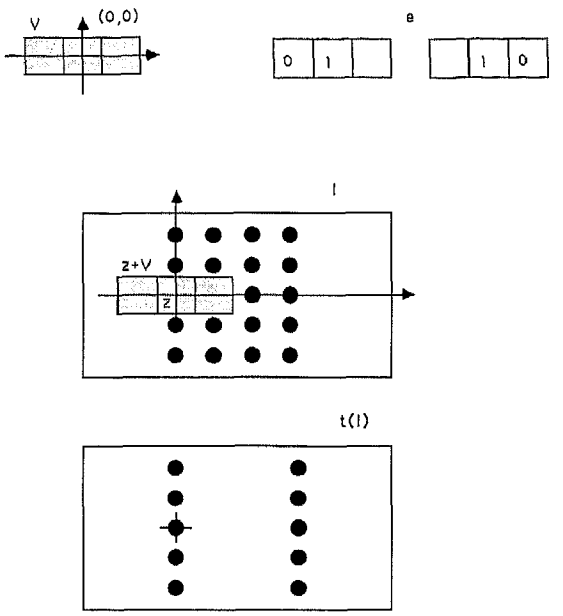

Fig. 1. — Définition d'un TCL de paramètres (V, e).

Si τ_z est la translation par z dans $\mathbf{Z}\times\mathbf{Z}$, ceci peut s'écrire en disant que la valeur de l'image binaire $t(\mathrm{I})$ au site z est l'élément binaire $e(\mathrm{I}\circ\tau_z)$ de $\{0,1\}$. A titre d'exemple, la détection de fronts montants ou descendants horizontaux peut être faite à l'aide du voisinage $\mathrm{V}=\{-1,0,1\}$ et de la fonction booléenne

$$e((\mathrm{X}_{(-1,0)}, \mathrm{X}_{(0,0)}, \mathrm{X}_{(1,0)})) = \bar{\mathrm{X}}_{(-1,0)} * \mathrm{X}_{(0,0)} + \mathrm{X}_{(0,0)} * \bar{\mathrm{X}}_{(1,0)}$$

Il faut noter qu'aucune contrainte n'est faite ni sur la forme ou la taille du voisinage V, ni sur la fonction booléenne e. Par ailleurs il est clair que plusieurs couples (V, e) différents peuvent définir le même TCL. On peut naturellement introduire une notion de couple « minimal » définissant un TCL t donné.

Il apparaît clairement sur cette définition que les TCL sont des opérateurs locaux, selon le sens suivant: pour un opérateur t fixé, il existe une partie finie V de $\mathbf{Z}\times\mathbf{Z}$ telle que pour toute image I et pour tout site z de $\mathbf{Z}\times\mathbf{Z}$, $t(\mathrm{I})$ au site z ne dépend que du voisinage $(z+\mathrm{V})$ de z dans $\mathbf{Z}\times\mathbf{Z}$. De même les TCL sont clairement invariants par translation.

Réciproquement un opérateur t local au sens précédent et invariant par translation est un TCL. En effet dans ce cas, du fait de l'invariance par translation, $t(\mathrm{I})$ au site z coïncide avec $t(\mathrm{I}\circ\tau)$ au site (0,0). Si on

appelle e la fonction booléenne $\{0,1\}^V \to \{0,1\}$ qui à J image binaire incluse dans V associe $t(J)$, on voit que $t(I)$ au site z coïncide avec $e(I \circ \tau_z)$ et donc que t est le TCL de paramètres (V, e).

Dès lors si t_1 et t_2 sont deux opérateurs locaux invariants par translation, le «ou logique», le «et logique» (calculés site par site) et la composée de t_1 et de t_2 sont des opérateurs locaux et invariants par translation, donc l'ensemble des TCL est stable par «ou», «et» et composition.

On peut maintenant chercher à décrire une paramétrisation de la composée de deux TCL. Pour cela une définition ensembliste est plus aisément manipulée. Deux définitions équivalentes des TCL peuvent être proposées dans ce cadre. Tout d'abord rappelons qu'à toute partie A de $\mathbf{Z} \times \mathbf{Z}$ nous pouvons associer sa fonction caractéristique $I_A : \mathbf{Z} \times \mathbf{Z} \to \{0,1\}$ qui a tout élément z de $\mathbf{Z} \times \mathbf{Z}$ associe 1 s'il appartient à A et 0 sinon. Représenter une image binaire par une partie de $\mathbf{Z} \times \mathbf{Z}$ équivaut donc à la représenter par une application $\mathbf{Z} \times \mathbf{Z} \to \{0,1\}$. Proposons maintenant une description équivalente des paramètres d'un TCL. Soit V une partie finie de $\mathbf{Z} \times \mathbf{Z}$. Soit e une fonction booléenne $\{0,1\}^V \to \{0,1\}$; appelons support de e l'ensemble des imagettes que e «reconnaît», c'est-à-dire l'ensemble U de toutes les parties X de V telles que $e(X) = 1$. Alors si I est une image binaire, considérée comme une partie de $\mathbf{Z} \times \mathbf{Z}$, l'image $t(I)$ résultant de l'application du TCL t de paramètres $[V, U]$ à l'image binaire I est définie de manière équivalente comme suit:

$$t(I) = \{ z \in \mathbf{Z} \times \mathbf{Z}, (-z + I) \cap V \in U \}.$$

Ceci permet de donner une paramétrisation de la composée $t_2 \circ t_1$ de deux TCL t_1 et t_2 de paramètres $[V_1, U_1]$ et $[V_2, U_2]$ respectivement, ce qui est illustré figure 2. Il s'agit de $[V_1 \oplus V_2, U]$, où U est:

$$U = \{ X \in V_1 \oplus V_2, t_1(X) \in U_2 \}.$$

2.3. Lien avec la morphologie mathématique

Il est clair que les TCL sont liés aux opérateurs de logique cellulaire [Preston 79] et de morphologie mathématique [Serra 82]. C'est ce que nous précisons ici.

Représentons la fonction booléenne e sous forme normale disjonctive, c'est-à-dire comme «somme» logique («ou logique») de L «produits» logiques e_λ («et logique»):

$$e((X_z)_{z \in V}) = \sum_{1 \leq \lambda \leq L} e_\lambda((X_z)_{z \in V})$$

Naturellement t est le «ou logique» des TCL t_λ de paramètres (V, e_λ):

$$t = \sum_{1 \leq \lambda \leq L} t_\lambda$$

Maintenant, chacun des termes produits e_λ est défini grâce à deux parties de V qui sont l'ensemble U_λ des variables X_z qui apparaissent dans e_λ et l'ensemble Z_λ des variables X_z qui y apparaissent en étant conjuguées («complémentées», ce qui est noté par $\bar{X}_z$).

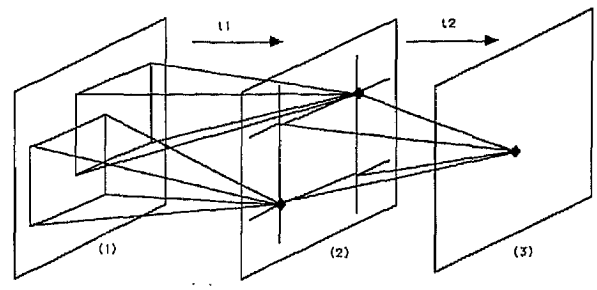

Fig. 2. – Composition de deux TCL t_1 et t_2.
L'application de t_1 aux motifs de $t_2 \circ t_1$ produit des motifs de t_2.

Ainsi e_λ s'écrit:

$$e_\lambda((X_z)_{z \in V}) = \prod_{z \in U_\lambda} X_z * \prod_{z \in Z_\lambda} \bar{X}_z$$

Considérons une partie Y de V. La condition $e_\lambda(1_Y)$ vaut 1 équivaut à:

– pour tout z dans $U_\lambda : 1_Y(z) = 1$ donc $z \in Y$ *i.e.* $Y \supset U_\lambda$;

– pour tout z dans $Z_\lambda : 1_Y(z) = 0$ donc $z \notin Y$ *i.e.* $Y^c \supset Z_\lambda$.

Ainsi le TCL t_λ de paramètre (V, e_λ) coïncide avec la transformation par tout ou rien (TTR) d'élément structurant (U_λ, Z_λ), au sens où elle est définie en morphologie mathématique sur des images binaires discrètes [Serra 82].

De ce fait il apparaît que tout TCL est la réunion d'un nombre fini de transformations par tout ou rien. Réciproquement, puisque l'ensemble des TCL est stable par union, intersection et composition en nombre fini, tout opérateur morphologique obtenu par composition d'un nombre fini de transformations par tout ou rien est un TCL.

2.4. Conclusion

Au total, nous avons donné une définition des TCL, puis une caractérisation de leur ensemble, et nous avons montré qu'ils sont stables pour la composition. Nous avons ensuite montré qu'ils peuvent être obtenus comme union de transformations par tout ou rien. C'est finalement cette dernière façon de les définir qui sera la plus utilisée, les TTR servant de filtre, et leurs éléments structurants de motifs.

D'une manière plus philosophique, on peut dire d'une part que les TCL correspondent à une classe d'opérateurs morphologiques qui satisfont les trois premiers principes de la morphologie exposés par [Serra 82], mais pas le quatrième (ils ne sont pas croissants). A l'inverse, leur ensemble a la particularité d'être stable par composition, propriété qui n'est pas cruciale en morphologie mathématique. On peut dire d'autre part que les approches de la reconnaissance sont différentes, puisque la reconnaissance à partir de TCL utilise volontairement une représentation explicite de la forme à travers des opérateurs géométriques, alors qu'en morphologie on utilise volontairement une représentation implicite de la forme, à travers un ensemble de mesures de caractéristiques.

3. Images binaires seuillées

Dans ce chapitre, nous expliquons comment les TCL peuvent être utilisés pour la reconnaissance de motifs dans des images binaires seuillées.

3.1. Méthode booléenne des masques

Nous allons expliquer maintenant comment ces opérateurs peuvent être utilisés pour implanter une méthode booléenne des masques. Le procédé utilisé consiste à construire à partir d'images du motif à reconnaître des suites d'opérateurs qui marquent les sites où un motif voisin du motif recherché a été trouvé. Pour cela, considérons d'abord une petite image binaire P incluse dans une fenêtre rectangulaire R. Soit alors t la TTR de motif (P, R-P), où R-P désigne le complémentaire de P dans R, comme illustré figure 3. L'application de t à une image binaire I est:

$$t(\mathrm{I}) = \{ z \in \mathbf{Z} \times \mathbf{Z}, (-z+\mathrm{I}) \cap \mathrm{R} = \mathrm{P} \}.$$

Ainsi t marque les sites z dont le voisinage $(z+\mathrm{R})$ coïncide pixel à pixel avec le motif défini par (P, R-P). Bien entendu, si l'on cherche des motifs ressemblant à P dans une autre image acquise J par l'application de t, on n'obtiendra aucune coïncidence. A ce stade cette approche est très semblable à celle de la corrélation [Barnea 72]. Il nous faudrait donc définir un coefficient de ressemblance entre images, analogue au coefficient de corrélation. Néanmoins les TCL ne permettent pas d'effectuer directement des opérations arithmétiques, et donc nous devons nous appuyer sur un coefficient de similitude qui puisse être calculé géométriquement.

Une première approche dans cette direction consiste à remplacer t défini par le motif (P, R-P) par t' de motif $(\mathrm{P}_n, (\mathrm{R-P})_n)$, où P_n et $(\mathrm{R-P})_n$ sont l'érosion de P et de R-P respectivement par des éléments structurants de taille inférieure à n, typiquement carrés, comme illustré figure 3. Les images qui sont acceptées par ce motif sont semblables, en un sens très géométrique, au motif (P, R-P).

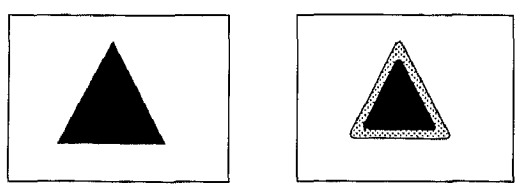

Fig. 3. — Définition d'un TCL associé à un motif. Ici on représente sur la même image binaire les éléments U et Z de t, ainsi que leurs versions érodées U′ et Z′ de t'.

Une approche plus précise s'inspire de la distance de Hausdorff dans le plan $\mathbf{R} \times \mathbf{R}$. En effet, la distance de Hausdorff de deux parties compactes A et B de $\mathbf{R} \times \mathbf{R}$ est définie par:

$$d(\mathrm{A}, \mathrm{B}) = \inf \{ \varepsilon \in \mathrm{R}, (\mathrm{B} \oplus \mathrm{D}_\varepsilon) \supset \mathrm{A} \text{ et } (\mathrm{A} \oplus \mathrm{D}_\varepsilon) \supset \mathrm{B} \}$$

où $\oplus$ représente la somme de Minkowski et D_ε est un disque de diamètre ε. En particulier, si ε est un nombre positif, les conditions:

$$(\mathrm{B} \oplus \mathrm{D}_\varepsilon) \supset \mathrm{A} \qquad \text{et} \qquad (\mathrm{A} \oplus \mathrm{D}_\varepsilon) \supset \mathrm{B}$$

équivalent à ce que la distance de Hausdorff de A et B soit plus petite que ε.

Par analogie, identifions $\mathbf{Z} \times \mathbf{Z}$ aux sites à coordonnées entières de $\mathbf{R} \times \mathbf{R}$, considérons la discrétisation du disque carré à l'origine de rayon n:

$$\mathrm{D}_n = \{ z \in \mathbf{Z} \times \mathbf{Z}, d(0, z) \leqq n \}$$

et marquons les sites z qui satisfont:

$$(z+\mathrm{P}) \oplus \mathrm{D}_n \supset (z+\mathrm{V}) \cap \mathrm{I}$$
$$\mathrm{I} \oplus \mathrm{D}_n \supset z+\mathrm{P}$$

Ceci vérifie que la distance de Hausdorff entre $(z+\mathrm{V}) \cap \mathrm{I}$ et $(z+\mathrm{P})$ est inférieure à n. Le calcul de la représentation discrétisée d'un disque de rayon n est assez lourde, mais il est raisonnable de supposer que le motif correspondant est mémorisé une fois pour toutes.

Pour aller au-delà, nous souhaitons introduire une similarité plus structurelle entre les motifs tout en s'appuyant sur des TCL. Pour cela, choisissons une fenêtre carrée R_1 et G une grille régulière à maille carrée incluses dans R, comme illustré figure 5.1. Considérons maintenant les fenêtres de taille R_1 extraites aux sites de G dans P: en chaque site z de G appelons «tuile» W_z l'image binaire $\mathrm{R}_1 \cap (-z+\mathrm{P})$. Soit aussi T_z le motif $(\mathrm{W}_z, \mathrm{R}_1 - \mathrm{W}_z)$ et t_1 le TCL défini par l'ensemble des motifs $(\mathrm{T}_z)_{z \in \mathrm{G}}$, comme illustré figure 5.2. Ce TCL t_1 est ainsi constitué des motifs extraits de P aux sites de la grille G. Soit aussi t_2 le TCL de motif (G, $\varnothing$), comme illustré figure 5.3. Ce TCL t_2 est constitué d'un unique motif repérant la présence de tous les sites de G.

Choisissons le pas de la grille G et la taille de la fenêtre R_1 de manière à ce que les tuiles associées à deux sites z_1 et z_2 voisins dans G se chevauchent et de manière à ce que: $\mathrm{R}_1 \oplus \mathrm{G} \supset \mathrm{P}$; c'est-à-dire que G doit être recouvert par l'union des tuiles $(\mathrm{W}_z)_{z \in \mathrm{G}}$ extraites aux sites de G.

D'après la caractérisation de la composition que nous avons donnée dans le paragraphe 2.2, $t_2 \circ t_1$ a pour paramètre $[\mathrm{R}_1 \oplus \mathrm{R}_2, \mathrm{U}]$, où U est l'ensemble des parties X de R que t_1 transforme en G: ces parties X sont donc obtenues en plaçant l'une des tuiles de $\{\mathrm{W}_z\}_{z \in \mathrm{G}}$ en chaque site z' de G. Dans le cas d'images réelles il n'y a généralement qu'une manière unique

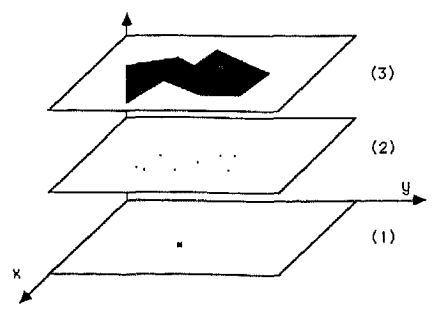

Fig. 4. — Reconnaissance de motifs par application de t_1 et de t_2. (1) Image résultante de l'application de t_1 et t_2. (2) Image résultante de l'application de t_1 seulement. (3) Image initiale.

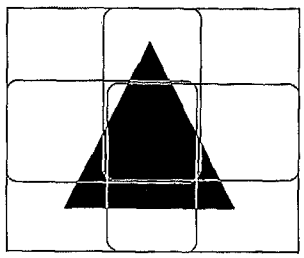

Fig. 5.1. — Fenêtres $W_z = z + R_1$ situées aux points de G.

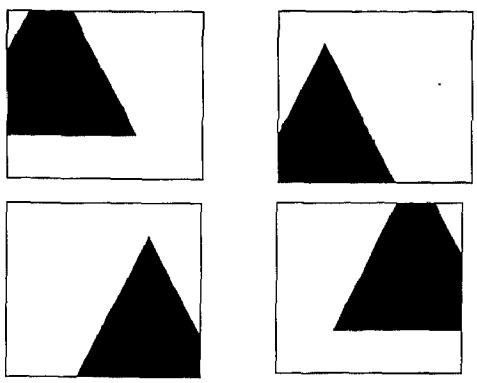

Fig. 5.2. — Quatre motifs de t_1.

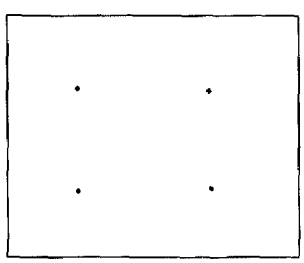

Fig. 5.3. — t_2 est défini par un motif unique.

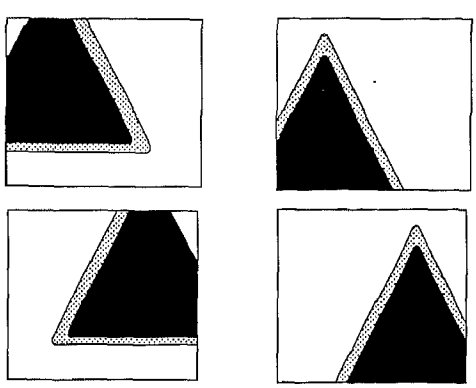

Fig. 5.4. — Les motifs de t'_1 sont ceux de t_1 après érosion.

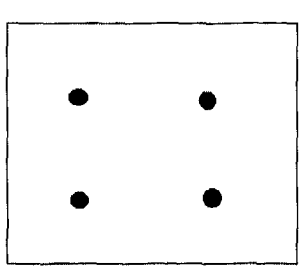

Fig. 5.5. — Les motifs de t'_2 sont obtenus par dilatation puis suppression.

Fig. 5. — Définition d'une suite de TCL associée à un motif.

de réarranger ces tuiles, et celle-ci reconstitue le motif initial P. Alors l'application successive de t_1 et de t_2 marque les sites où se retrouve exactement P (voir *fig.* 4).

Il est alors possible d'introduire de plus une notion de similarité. On remplace t_1 par t'_1 en utilisant les procédés géométriques décrits précédemment: on applique à ses motifs une érosion, comme illustré figure 5.4, ou on les remplace de manière à tester la distance de Hausdorff.

On remplace de même t_2 par t'_2 en introduisant une notion de similarité plus structurelle, comme illustré figure 5.5. D'une part les sites résultant de l'application de t_1 peuvent être situés au voisinage de ceux de G: ceci se vérifie en appliquant une dilatation au motif de t_2, ou de manière équivalente en appliquant une dilatation t'_3 d'élément structurant un petit disque à l'image résultant de l'application de t_1 avant l'application de t_2. D'autre part un petit nombre de sites de G peut ne pas être retrouvé par l'application de t_1. Pour cela t_2 est remplacé par un TCL t'_2 qui accepte qu'un petit nombre de points de G soient absents. Ceci signifie que l'unique motif de t_2 est remplacé par une suite de motifs, chacun étant obtenu en ôtant à G l'un de ses sites. La combinatoire ainsi introduite reste faible pour un petit nombre d'absences acceptées.

Alors la composée $t'_2 \circ t'_3 \circ t'_1$ définit une notion de similarité plus riche quoique encore très géométrique. Une étude des paramètres décidant du chevauchement et du recouvrement a été effectuée sur des images de petits motifs géométriques et est décrite dans [Reichart 87].

3.2. Exemples

Nous proposons le cas d'une séquence d'images infrarouges de faible résolution et de petite taille. Il s'agit

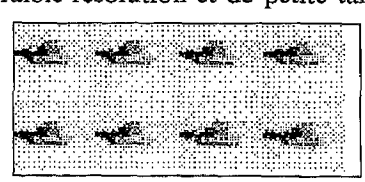

Fig. 6.1. — Image initiale.

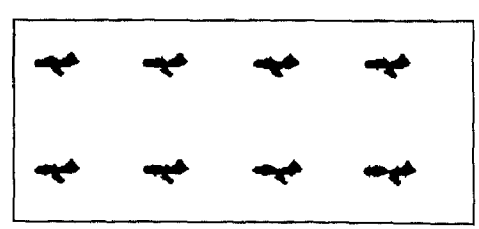

Fig. 6.2. — Après application de t_1.

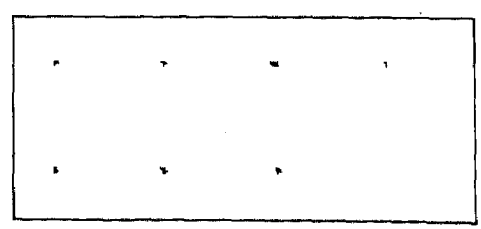

Fig. 6.3. — Image finale des marques.

Fig. 6. — Exemple de traitement d'images seuillées.

d'images naturelles numérisées et seuillées. L'image originale est montrée tramée en figure 6.1. Elle contient 8 imagettes 64×64. Le premier char sert à la construction des TCL t_1 et t_2. Une érosion de support de taille 1 est appliquée aux motifs de t_1, et une dilatation de support de taille 1, ainsi que la tolérance d'une absence, est appliquée au motif de t_2. Le résultat de l'application de t'_1 est montré figure 6.2, celui de t'_2 figure 6.3. On constate un taux de reconnaissance très acceptable pour des motifs raisonnablement semblables.

3.3. Conclusion

Les procédés de détection que nous venons de décrire sont extrêmement frustes. De nombreuses méthodes ont été décrites pour la reconnaissance d'images binaires, dont [Montanvert 87] donne une synthèse. Elles consistent à prétraiter l'image binaire (filtrages, détection de contours, ...), puis à en calculer une description structurelle qui s'appuie sur des contours, des approximations polygonales, des squelettes ou lignes médianes, si l'on s'intéresse à des représentations linéaires construites à partir des contours ou des régions. D'autres descriptions structurelles peuvent être construites à partir des régions elles-mêmes, telles que des graphes de relation de parties convexes résultant d'une partition ou d'un recouvrement de l'image. Ces méthodes plus sophistiquées sont naturellement beaucoup plus fines et beaucoup plus efficaces. Néanmoins notre propos n'est pas ici de les remplacer, mais de prendre en compte les particularités architecturales d'une rétine électronique parallèle. Or, d'une part ces méthodes sont exploitées dans un environnement très proche d'un expert humain, qui peut intervenir dans les prises de décision en s'appuyant sur son savoir-faire. Cette situation est très différente de celle d'un système qui prend ses décisions de façon automatique. D'autre part elles ont l'inconvénient de ne pas pouvoir s'implanter immédiatement sur un calculateur parallèle, et surtout de ne pas pouvoir s'insérer dans une architecture monolithique réalisable à court terme.

4. Images binaires en demi-teinte

Les applications des images binaires seuillées sont limitées aux cas où les silhouettes des objets sont suffisantes pour représenter les objets à examiner. Nous voulons dépasser au moins partiellement ces restrictions tout en ne considérant que des opérateurs TCL. Pour cela nous envisageons de traiter des images en demi-teinte, c'est-à-dire dans lesquelles l'information multiniveau est codée par la densité de pixels blancs («valant 1») d'une image binaire discrète. Ceci permet tout d'abord de conserver davantage d'information multiniveau tout en manipulant des images binaires. De plus, l'occupation mémoire et la durée des traitements de telles images ne sont pas réduites par des codages du type «run-length coding», et de telles images exploitent donc davantage le parallélisme de la rétine que nous avons décrite. Dans ce chapitre, nous mentionnons d'abord quelques procédés de demi-teinte bien adaptés à l'implantation VLSI, puis nous décrivons le traitement des images en demi-teinte par TCL et enfin nous commentons quelques exemples.

4.1. Procédés de demi-teinte

De nombreux procédés numériques de binarisation en demi-teinte ont été étudiés [Stoffel 81]. Ils ont été développés pour la visualisation d'images multiniveaux sur des dispositifs biniveaux, tels que les écrans noir et blanc «bitmap» ou les imprimantes. Les plus utilisés sont les techniques de tramage et de diffusion d'erreur. Dans cet article, nous utilisons les techniques de tramage, qui consistent à binariser l'image multiniveau par comparaison pixel par pixel à un signal bidimensionnel périodique (*fig.* 7.1). Pour ce procédé il est nécessaire de choisir une matrice de seuils (θ_{kl}) avec $1 \leqq k \leqq M$ et $1 \leqq l \leqq M$ appelée trame. A chaque pixel analogique est associé un pixel binaire. La méthode consiste alors à recouvrir l'image analogique (A_{ij}) avec $1 \leqq i \leqq N$ et $1 \leqq j \leqq N$ par juxtaposition de la trame résultant en une image (T_{ij}) avec $1 \leqq i \leqq N$ et $1 \leqq j \leqq N$ où $T_{ij} = \theta_{(i \bmod M),\ (j \bmod M)}$.

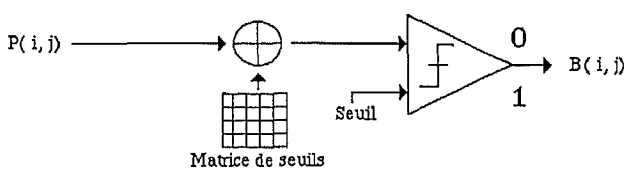

Fig. 7.1. — Principe du tramage.

0	32	8	40	2	34	10	42
48	16	56	24	50	18	58	26
12	44	4	36	14	46	6	38
60	28	52	20	62	30	54	22
3	35	11	43	1	33	9	41
51	19	59	27	49	17	57	25
15	47	7	39	13	45	5	37
63	31	55	23	61	29	53	21

Fig. 7.2. — Matrice de Bayer d'ordre 8.

Fig. 7. — Tramage.

Chaque pixel analogique A_{ij} est alors comparé au seuil T_{ij} qui lui fait face, et le pixel binaire B_{ij} correspondant vaut:

0 si $A_{ij} \leqq T_{ij} = \theta_{(i \bmod M),\ (j \bmod M)}$;

1 si $A_{ij} > T_{ij} = \theta_{(i \bmod M),\ (j \bmod M)}$.

Plusieurs trames ont été proposées [Judice 76], et nous utilisons ici celle décrite par [Bayer 73] (*fig.* 7.2). Nous décrivons dans le chapitre architecture une implantation de ces procédés. Une technique itérative de binarisation en demi-teinte, qu'on peut considérer comme une parallélisation de la diffusion d'erreur bien adaptée à l'implantation VLSI, est décrite dans [Bernard 88].

4.2. Méthode booléenne des masques en demi-teinte

L'approche suivie pour le traitement TCL d'images en demi-teintes consiste à construire des TCL qui détectent des régions, caractérisées par leur niveau de gris, dont la variation est supposée très faible.

Pour cela, il est tentant de détecter l'appartenance à un niveau de gris donné par le repérage de certaines imagettes. Observons ce qui se passe dans une région dont le niveau de gris est supposé constant: binarisons-la par tramage à l'aide de la trame de Bayer. Mesurons alors la loi de répartition d'imagettes de petite taille, 2×2 par exemple, dans l'image binaire résultante. Leurs distributions ne sont pas équivalentes. L'occurrence du motif entièrement blanc est en moyenne plusieurs fois plus grande que l'occurrence de celui avec un seul point blanc. Nous illustrons le cas $n=2$, $m=2$ en figure 8 pour une image multiniveau uniforme (*i. e.* dont le niveau de gris est constant) de taille 64×64 dont le niveau de gris varie de 0 à 31. Soit en effet b le nombre de points noirs pour un support $n \times m$ de p points ($p=n \times m$). Nous avons C_p^b combinaisons de points noirs et blancs.

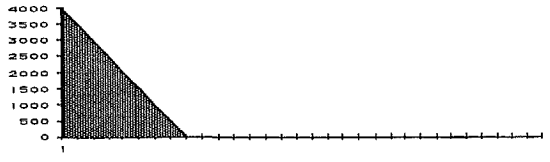

Fig. 8.1. — Courbe de répartition de l'échantillon blanc.

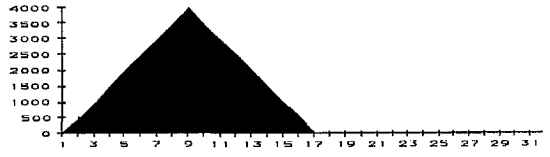

Fig. 8.2. — Courbe de répartition de l'échantillon avec un seul point noir.

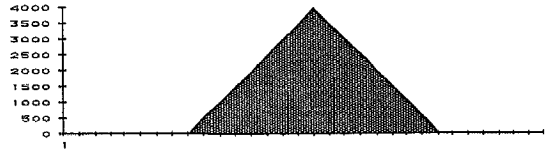

Fig. 8.3. — Courbe de répartition de l'échantillon avec deux-points-noirs. Courbes de répartition d'imagettes 2×2 dans une image de niveau de gris constant. En abscisse : niveau de gris de l'image. En ordonnée : occurrence de l'imagette 2×2 dans l'image binaire en demi-teinte.

Fig. 8. — Répartition d'imagettes.

Nous pouvons alors associer à un niveau de gris N un TCL F qui constitue un compteur spatial, dont les motifs sont toutes les images de taille $n \times m$ et de densité N. Grâce à cet artifice, il est possible de tenter une technique analogue à celles du chapitre 2. L'introduction de points indifférents dans les motifs de F permet une tolérance vis-à-vis de la densité de points à 1, et donc vis-à-vis du niveau de gris N. Dans le cas d'un TCL isotrope, tous les arrangements de point indifférents sont à considérer (*fig.* 9).

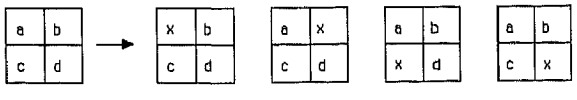

Fig. 9. — Introduction d'un site indifférent dans une imagette 2×2.

En fait, les motifs de F sont générés par l'introduction de q points indifférents. Pour avoir un TCL F isotrope, le nombre des arrangements est C_p^q. Il peut être observé expérimentalement que les modes des distributions d'imagettes sont élargis proportionnellement au nombre de pixels indifférents (*fig.* 10). Ainsi, les points indifférents introduisent une notion de «similarité» des niveaux de gris.

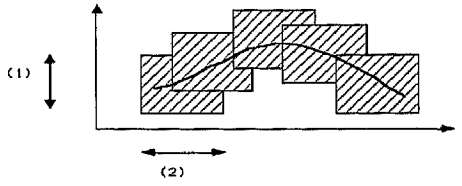

Fig. 10.1. — Courbe de répartition de l'échantillon à trois points «indifférents». (1) densité: la hauteur des rectangles représente la tolérance pour W. (2) position spatiale: la largeur caractérise la tolérance sur M. Un échantillon doit être dans la zone hachurée pour être reconnu.

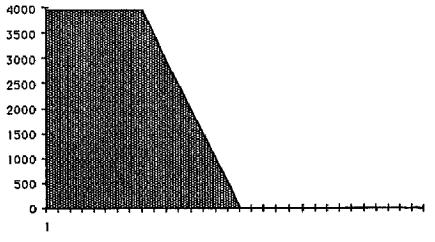

Fig. 10.2. — Courbe de répartition de l'échantillon à un point «indifférent».

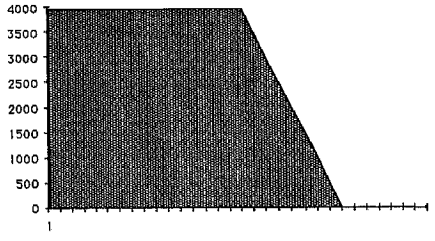

Fig. 10.3. — Courbe de répartition de l'échantillon à deux points «indifférents».

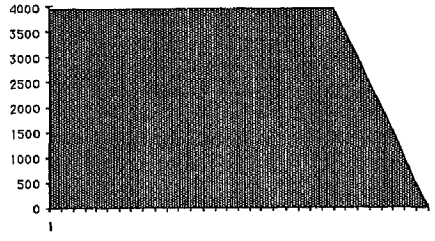

Fig. 10.4. — Courbe de répartition de l'échantillon à trois points «indifférents». En abscisse: niveau de gris de l'image. En coordonnée: occurrence de l'imagette 2×2 dans l'image binaire en demi-teinte.

Fig. 10. — Influence de la tolérance sur les motifs.

Néanmoins les TCL F engendrés d'une manière aussi brutale ont un nombre inacceptable de motifs. Par exemple, les C_p^q motifs du cas isotrope sont trop nombreux pour des images de taille raisonnable. Là encore une solution consiste à décomposer le TCL F. Ce TCL est décomposé en deux parties W et M. Le support de W étant petit, et la complexité de M faible, nous pouvons ainsi introduire sans explosion combinatoire une tolérance dans chacun de ces deux TCL (*fig.* 11).

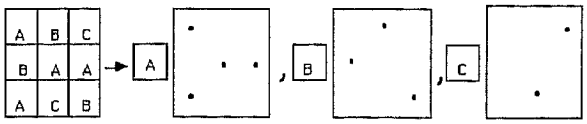

Fig. 11. — Décomposition d'un motif en neuf fenêtres élémentaires dans une image de niveau de gris constant. A, B, C sont les différentes fenêtres, c'est-à-dire de petites images binaires.

Chaque motif f de F est partitionné en ω fenêtres élémentaires juxtaposées w_j de p pixels chacune. Ceci rappelle le procédé de grille précédent, la grille étant choisie cette fois de manière à éviter tout recouvrement et à recouvrir entièrement f. Le nombre des instances de fenêtres w_j est ainsi limité à 2^p *i. e.* $\omega \leqq 2^p$. Pour chaque w_j, nous marquons ses occurrences dans f par un motif m_j. Ainsi, le motif initial f est décomposé en fenêtres élémentaires et en masques, et nous avons donc, en appelant W_j le TCL de motif w_j et M_j celui de motif m_j:

$$F = \prod_{1 \leq j \leq J} (M_j \circ W_j)$$

où Π représente le «et logique» et «$\circ$» la composition.

Une tolérance sur les niveaux de gris est alors rendue possible. Tout d'abord l'introduction de points indifférents dans les motifs w_j permet à chaque W_j de repérer un ensemble de niveaux de gris proches à une échelle microscopique. De plus l'introduction — via une dilatation — d'une tolérance en position spatiale dans m_j permet à M_j d'accepter des assemblages de trames microscopiques différents.

4.3. Exemples

Nous examinons tout d'abord un exemple jouet pour illustrer l'utilisation de la décomposition précédente (*fig.* 12). Les images correspondantes sont des photos de jouets en plastique, de couleur uniforme, bien exposés, et pour lesquels l'hypothèse de niveau de gris constant est plausible. Ici l'image de départ est numérisée en 512×512. Elle est tramée à l'aide d'une trame de Bayer.

Pour construire le TCL F_1 repérant le niveau de gris de la face supérieure du troisième objet de la figure 12.1, un échantillon de 4×4 points est pris dans l'image de cette face supérieure. W_1 est construit à partir d'une fenêtre de 2×2 points. Le nombre q d'indifférents introduits est 1. La tolérance sur M_1 est obtenue par l'application d'une dilatation dont le support est de taille 2. Le même algorithme est appliqué à la construction d'une suite W_2, M_2 repérant le niveau de gris de la face avant du troisième objet.

Les suites de TCL $M_1 \circ W_1$ et $M_2 \circ W_2$ sont alors appliquées à l'image initiale, les résultats étant respectivement les images 12.3 et 12.4. Il apparaît que, bien que les niveaux de gris des deux régions soient très voisins, ils sont bien séparés.

La même expérience est effectuée avec le cylindre, et est illustrée figures 12.2, 12.5 et 12.6.

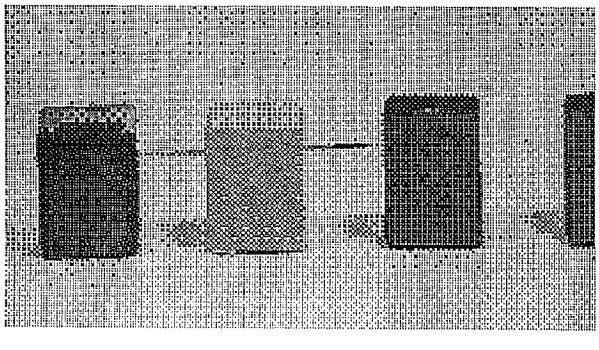

Fig. 12.1. — Image initiale «carré».

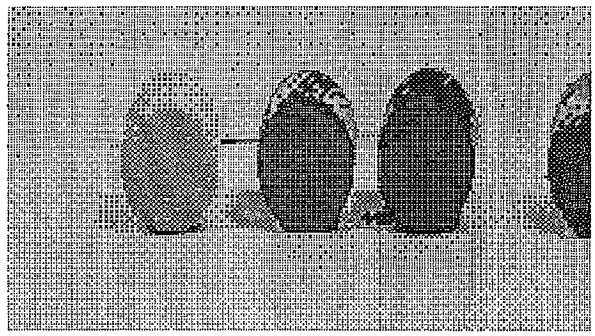

Fig. 12.2. — Image initiale «rond».

Fig. 12.3. — Image «cube» après (M, W).
Le motif appris est extrait du sommet du troisième cube.

Fig. 12.4. — Image «cube» après (M, W).
Le motif appris est extrait de la face du troisième cube.

Fig. 12.5. — Image «rond» après (M, W).
Le motif appris est extrait du sommet du troisième cylindre.

Fig. 12.6. — Image «rond» après (M, W).
Le motif appris est extrait de la face du troisième cylindre.

Fig. 12. — Expérimentation images jouets.
Trame de Bayer, image 64×64 pixels.

Nous examinons ensuite l'exemple d'images infrarouges (*fig.* 13). Il s'agit cette fois d'images naturelles numérisées et tramées à l'aide d'une trame de Bayer. Il s'agit de la même séquence d'images que dans le chapitre 3.

Tout d'abord nous construisons une grille g de taille 5×5 sur les points chauds du motif à apprendre. Nous définissons un recouvrement de l'image du char par des tuiles de taille 6×6 posées aux points de cette grille. Ensuite chaque tuile est décomposée en fenêtres w de taille 3×1 et en masques m comme nous venons de le décrire. Une tolérance est introduite au niveau de chacun des trois TCL G, M, W: le nombre q d'indifférents introduits dans W est 1, le nombre m de dilatations introduites dans M est 2. L'application du TCL $M \circ W$ à l'image initiale de la figure 13.1 résulte en l'image de la figure 13.2; l'application de G modifié produit l'image de la figure 13.3, où les motifs reconnus sont marqués par des points noirs. Les résultats sont analogues à ceux des images seuillées, parce qu'on ne s'est appuyé que sur une seule région.

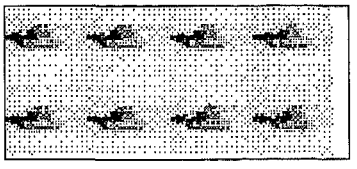

Fig. 13.1. — Image de char infrarouge *b*-codée.

Fig. 13.2. — Grille tracée sur la première image.

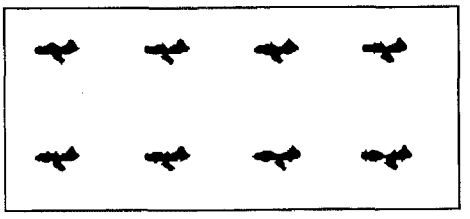

Fig. 13.3. — Application de (M, W).
Le dernier motif n'est pas détecté.

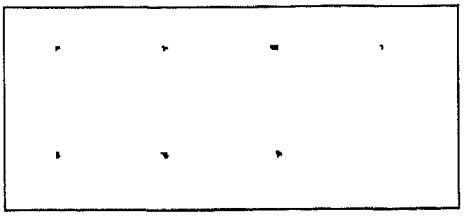

Fig. 13.4. — Dernière itération.

Fig. 13. — Expérimentation images chars tramées.

4.4. Conclusion

Nous avons montré comment il est possible de retrouver des niveaux de gris dans une image binaire en demi-teinte à l'aide de TCL en échappant à l'explosion combinatoire dans des cas simples. Là encore ce procédé est extrêmement fruste, et sans doute bien moins performant que les procédés classiques de recherche de zones homogènes par croissance de régions ou partage et réunion.

A l'inverse il permet l'implantation de tels procédés dans un processeur parallèle monolithique. Qui plus est il permet de différencier des motifs du fait d'un assemblage de zones homogènes de niveau de gris constants mais différents, ce qui n'est pas possible avec les procédés exposés dans le chapitre 3.

5. Architecture de la rétine

La rétine à concevoir doit pouvoir acquérir une image binaire discrète par seuillage ou tramage, lui appliquer une itération de TCL arbitraire, et être parallèle. Nous exposons d'abord l'implantation du tramage, ensuite les motivations ayant présidé au choix de l'architecture de la rétine, puis nous détaillons celle-ci.

5.1. Implantation du tramage

Plusieurs cellules de tramage ont été réalisées. L'opération d'acquisition du pixel analogique A_{ij}, de comparaison au seuil T_{ij} et de mémorisation du pixel binaire B_{ij} est effectuée par une cellule appelée micropixel. Ces cellules sont tout d'abord rangées en une matrice de la taille de la trame, de façon à ce que chaque cellule corresponde à un seuil différent: ce groupe est appelé un macropixel. A la juxtaposition périodique de la trame sur l'image analogique à binariser correspond alors la duplication périodique 2-D des macropixels, qui constitue la réalisation matérielle du tramage: nous avons alors cherché des moyens aussi économiques que possible qui permettent de différencier les seuils des micropixels d'un macropixel.

L'acquisition d'un pixel est effectuée par une photodiode fonctionnant en mode dynamique. Elle est alors associée à un seuillage et à une mémorisation statique de la valeur binaire obtenue, tous deux locaux au micropixel. Examinons le fonctionnement de celui-ci, dont le schéma est représenté figure 14.1, et où la photodiode est représentée par une source de courant en parallèle sur une jonction. Le chronogramme de fonctionnement est donné figure 14.2, où f_1 et f_2 sont des flux d'intensités lumineuses différentes, auxquels est soumise la photodiode. Tout d'abord la photodiode est préchargée ($\varphi_1 = 1, \varphi_2 = 0, \varphi_3 = 0$) via le transistor T_1 à la tension (VDD-Vts). Ensuite le flux éclairant la photodiode est intégré ($\varphi_1 = 0, \varphi_2 = 1, \varphi_3 = 0$) et il décharge proportionnellement la photodiode. Enfin la tension aux bornes de la photodiode est comparée au seuil de l'inverseur, et la valeur binaire résultante est mémorisée ($\varphi_1 = 0, \varphi_2 = 0, \varphi_3 = 1$).

La différenciation des seuils des micropixels d'un macropixel s'effectue donc par la tension de seuil du premier inverseur du point mémoire. Pour que les micropixels du macropixel répondent différemment pour un même flux lumineux, trois possibilités ont été envisagées.

Une première solution est de figer les seuils à la construction de la cellule dans la géométrie de l'inverseur d'entrée de la cellule, via le rapport des (W/L) des transistors qui le constituent. Cette solution a

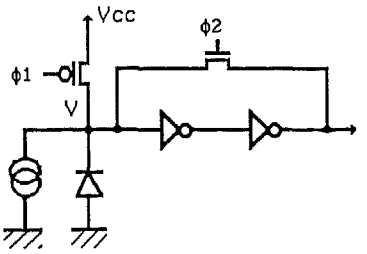

Fig. 14.1. — Seuil fixé à la construction.

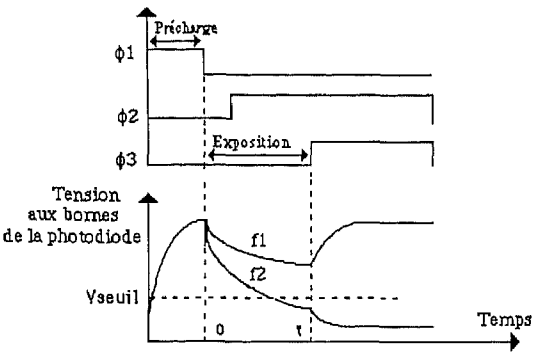

Fig. 14.2. — Chronogramme de la cellule.

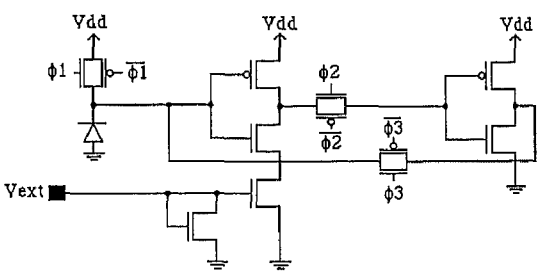

Fig. 14.3. — Seuil programmable par une tension externe.

Fig. 14. — Cellules de tramage.

l'inconvénient d'être figée et l'avantage d'être économe en surface. Une seconde solution est d'attribuer à chaque micropixel sa propre horloge de temps d'intégration. Ainsi les seuils sont programmables et l'encombrement des cellules reste réduit. Une troisième solution est de contrôler le seuil de l'inverseur à l'aide d'un miroir de courant commandé par une tension externe, chaque micropixel ayant sa propre commande. La cellule correspondante, illustrée figure 14.3, est plus difficile à mettre au point que la seconde sans permettre plus de seuils différents. C'est donc la seconde solution qui est préférable. Une étude plus détaillée est fournie dans [Rodriguez 86].

5.2. Choix d'une architecture

En ce qui concerne l'architecture du processeur, d'innombrables choix sont possibles. Bien que certains capteurs aient été conçus à l'aide de technologies CCD spécialisées, tels que ceux décrits par [Nudd 78] par exemple, nous souhaitons nous appuyer sur une technologie MOS standard, à l'instar de [Cole 83] et de [Lyon 81]. Ce choix va accroître la liberté du concepteur d'architecture aux dépends des qualités propres du capteur, telle que sa définition. Nous avons déjà mentionné le fait qu'un traitement « temps réel » de l'image en nécessite un traitement parallèle. D'une part, le parallélisme de l'image est particulièrement intéressant [Danielsson 82]. D'autre part, du point de vue du concepteur, il est séduisant de définir des architectures qui s'adaptent naturellement à l'évolution de la technologie. De cette manière les progrès en CAO et en VLSI permettent de réduire les délais de conception. [Seitz 84] s'est ainsi intéressé aux contraintes que l'intégration à haute densité impose aux architectures « concurrentes ». Bien entendu une architecture très répétitive est immédiatement attrayante par la simplification de la conception du circuit qu'elle apporte. Parmi les architectures possibles, les cas d'un processeur conventionnel, d'un processeur ligne ou d'un processeur tableau sont plus particulièrement remarquables. Une architecture conventionnelle, comme un microprocesseur de traitement de signal, ne paraît pas adaptée à la réalisation d'une rétine monolithique. Les tableaux linéaires de processeurs ont l'avantage d'apporter un gain linéaire en temps de calcul tout en n'augmentant que linéairement la quantité de matériel installée. Ils connaissent actuellement un regain d'intérêt [Fountain 86], [Linskog 86], [Basille 87]. Néanmoins dans le cas d'une réalisation monolithique une telle architecture consomme une partie de la surface en espace mémoire tampons et en séquenceur plus importante que dans le cas d'un processeur bidimensionnel. Enfin, même si l'on choisit une organisation bidimensionnelle, on peut choisir entre un grand nombre de petits processeurs et un petit nombre de gros processeurs. Ceci est lié au choix entre une organisation dense ou distribuée des processeurs [Danielsson 84]. Comme notre intention est de réaliser une rétine électronique, il nous a semblé que le choix d'associer un capteur par processeur élémentaire était plus naturel et plus efficace. Qui plus est, ce choix maximise le nombre de PE pouvant appliquer en parallèle un TCL.

L'idée de construire une machine cellulaire incluant une entrée optique remonte au projet UCPR1 de M. Duff [Duff 67]. Les évolutions successives de la machine cellulaire CLIP (numéros 1, 2 et 3) ont conduit à la machine CLIP4, la plus connue [Duff 86]. Celle-ci est bien adaptée aux opérateurs arithmétiques et logiques sur des images binaires ou multiniveaux. D'autres automates cellulaires ont été construits, tels que DAP [Reddaway 79] et MPP [Batcher 82]. Finalement le GAPP [GAPP 84] peut être considéré comme la brique de base d'un jeu de construction de processeurs tableaux. Néanmoins ces architectures s'appuient sur des processeurs élémentaires (PE) dont la complexité est de plusieurs ordres de grandeur supérieure à celle que nous pouvons accepter pour une réalisation monolithique.

5.3. Description de l'architecture

L'architecture de la rétine découle du jeu d'opérateurs qu'elle a à implanter. Elle a été esquissée dans [Garda 85-2]. La définition des TCL permet leur application parallèle sur une machine SIMD [Flynn 72] de type tableau de processeurs à communications locales (*fig.* 15.1). Pour rassembler le maximum de processeurs sur une même puce nous avons décidé de réaliser

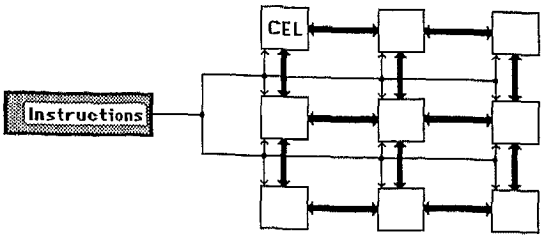

Fig. 15.1. — Tableau de processeurs.

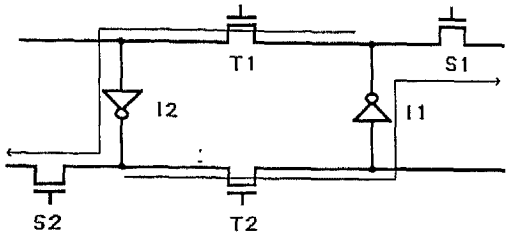

Fig. 15.2. — Registre à décalage bidirectionnel monodimentionnel.

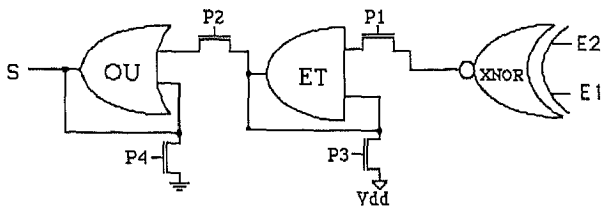

Fig. 15.3. — Processeur élémentaire.

Fig. 15. — Architecture de la rétine.

des processeurs les plus petits possibles, et ceci au détriment de la puissance de leur jeu d'instruction. [Batcher 82] Il s'agit donc de processeurs bit-série effectuant leurs communications en série. Dès lors l'application des TCL se fait en série sur les motifs, en série sur les pixels de chaque motif et en parallèle sur les pixels de l'image. Les éléments qui composent le processeur élémentaire se déduisent du calcul à effectuer pour appliquer un TCL à une image.

Un TCL est déterminé par un voisinage V et une suite de motifs $(T_\lambda)_{\lambda \in \Lambda}$, chacun composé de 1, 0 ou «indifférents». Pour appliquer cet opérateur à une image binaire I, on applique séquentiellement chaque motif. Pour appliquer un motif, on applique séquentiellement chacun de ses éléments. Pour cela, le PE au site z doit effectuer séquentiellement la comparaison de son voisinage $(z+V)$ au motif courant T_λ. Pour le site p du motif, le PE du site z doit comparer le pixel de I au site $z+p$ au pixel du motif, qui vaut 1, 0 ou indifférent. Pour cela le pixel au site p du motif doit être diffusé à tous les PE d'une part, et d'autre part le PE situé en z doit pouvoir accéder à la valeur du pixel situé au site $(z+p)$ et la comparer au pixel diffusé, par exemple via un «ou-exclusif» avec son complémentaire. Ensuite, le PE doit effectuer cette vérification pour tous les pixels du motif, et pour cela il doit effectuer le «et» des résultats. Enfin, il doit savoir si une coïncidence au moins a eu lieu dans la suite des comparaisons aux différents motifs, et pour cela il doit effectuer le «ou» des valeurs successives.

Pour ce qui est de l'accès aux valeurs des pixels, il faut noter que tous les PE effectuent simultanément l'accès aux pixels situés à des sites présentant la même translation. De plus n'importe quelle translation peut être effectuée par itération de translations élémentaires dans l'une des quatre directions Nord, Sud, Est ou Ouest. Enfin, ces translations élémentaires peuvent être effectuées à l'aide d'un registre à décalage bidirectionnel et bidimensionnel, lequel peut être réalisé par la connexion de deux registres à décalage bidirectionnels monodimensionnels tels que ceux décrits dans [Mead 81].

Ce registre bidirectionnel est décrit dans une version monodimensionnelle figure 15.2. Il fonctionne sous une horloge biphase démultiplexée en six lignes de commandes: les lignes non recouvrantes t_1 et t_2 alternent pour mémoriser une valeur binaire et cela constitue le point mémoire P. La succession de t_1 et de s_1 provoque un décalage vers la droite et de la même façon la succession de t_2 et de s_2 provoque un décalage vers la gauche. Symétriquement les suites t_2, v_2 et t_1, v_1 permettent des décalages unitaires vers le haut et vers le bas dans la version bidimensionnelle.

La partie traitement du processeur élémentaire est insérée en parallèle avec l'interrupteur T_1 (*fig.* 15.3). La sortie du point mémoire P est tout d'abord comparée par un «ou-exclusif» à la valeur diffusée e_2: ainsi si le complémentaire du pixel du motif est diffusé, la sortie du ou-exclusif vaut 1 si et seulement si les deux valeurs coïncident. La sortie de ce «ou-exclusif» est multipliée logiquement avec le contenu d'un point mémoire P_1, qui peut par ailleurs être initialisé à la valeur 1. Enfin le contenu de P_1 peut être additionné logiquement avec le contenu d'un point mémoire P_2, qui peut par ailleurs être remis à zéro. Enfin le contenu de P_2 peut être transféré dans P de manière à permettre l'application d'une suite de TCL à une même image.

Par ailleurs, une photodiode est située sur le registre à décalage, et sa sortie peut être seuillée et mémorisée dans le point P. L'étude de la réalisation dans une technologie MOS standard de photodiodes a été décrite dans [Garando 85].

Cette architecture est bien moins complexe que celle des tableaux de processeurs mentionnés précédemment. L'accès sériel par des registres à décalage aux données des voisins diffère de la porte logique d'entrée de CLIP4, des multiplexeurs de MPP et des registres NS/EW du GAPP. Certes la propagation combinatoire de CLIP4 n'est plus possible. D'un autre côté, cette implantation est particulièrement peu complexe puisqu'elle fournit l'accès aux données des voisins via ce qui est en fait une structure de mémorisation.

L'implantation du circuit intégré suit la conception logique. La taille et la forme du PE sont déterminées principalement par le nombre des lignes de contrôle et le nombre de niveaux d'interconnexion métallique disponibles. En NMOS le PE comprend moins de 25 transistors. Un prototype 8×10 PE a été réalisé en NMOS 4,5 µ,1 métal et a fonctionné en 1984. La surface de ce PE est de $35000\,\mu^2$ et la fréquence maximale d'horloge est de 5 MHz. En CMOS statique le PE comprend moins de 40 transistors. Avec une technologie CMOS 2µ,2 métaux une rétine

60×60 PE occupant moins de $50\,mm^2$ a été réalisée en novembre 1987. La surface de ce PE est de $11\,000\,\mu^2$ et la fréquence maximale d'horloge est de 20 MHz.

5.4. Utilisation de la rétine

L'utilisateur de la rétine se borne à spécifier les suites de motifs qui déterminent un TCL. Un séquenceur associé à la rétine traduit alors d'une manière systématique ces suites de motifs en commandes de décalage et de comparaison de pixels, selon le schéma décrit en 5.3.

Naturellement les images binaires ont été employées depuis fort longtemps en traitement d'images, et de nombreux opérateurs ont été développés. Telle qu'elle est, la rétine peut appliquer des opérateurs tels que érosion, dilatation et leurs itérations. Elle peut aussi implanter des opérateurs s'appuyant sur la détection de masques, tels que l'amincissement.

De plus l'adjonction d'une primitive de comptage global, telle que celles étudiées par [Reeves 80], permettrait la réalisation d'une sorte d'analyseur monolithique pour des images binaires. Une telle primitive permettrait l'extraction de caractéristiques fondées sur des mesures et des opérateurs géométriques telles que surface, degré de connexité, histogrammes, granulométries, ... On peut alors imaginer de transmettre les résultats fournis par la rétine à une autre puce chargée de la classification par exemple.

5.5. Conclusion

Les réalisations que nous venons de décrire confirment la validité de nos choix architecturaux. Elles sont originales tant par leur faible complexité que par leur très grande rapidité. A titre de comparaison les processeurs élémentaires de DAP et de CLIP4 comportent 3 000 transistors, celui de MPP 8 000. Le PE que nous avons décrit en comporte moins de 40, et sa vitesse d'exécution d'opérateurs morphologiques géométriques ou de TCL est excessivement élevée (quelques centaines de nanosecondes à quelques microsecondes).

6. Conclusion

Les rétines ne sont pas des systèmes universels et ne cherchent pas à remplacer tout système de vision; elles peuvent les décharger de certaines tâches ou assurer des fonctions complémentaires. Nous avons décrit la conception de l'architecture d'un tableau de processeurs en interaction avec les opérateurs qu'elle doit supporter. D'une part, une implantation monolithique d'un tableau de processeurs est une conséquence logique des contraintes de l'intégration à grande échelle sur l'évolution des tableaux de processeurs. D'autre part, une implantation d'aujourd'hui est limitée à des opérateurs booléens et à la mémorisation d'images binaires pour permettre la manipulation d'images d'une taille raisonnable. La validité des choix que nous avons effectués est confirmée par l'intégration d'une rétine rassemblant 60×60 PE sur une surface de $50\,mm^2$ dans une technologie CMOS $2\,\mu$,2 métal que nous avons effectuée en novembre 1987. Qui plus est, la richesse de ces schémas est confirmée par les nombreuses extensions qui peuvent lui être faites de manière à satisfaire aux contraintes d'une classe donnée d'algorithmes [Milgram 88] ou d'une classe donnée d'applications (étude en cours en collaboration avec l'Aérospatiale). L'intérêt des architectures cellulaires «à grain très fin» est encore renforcé par la possibilité de les implanter dans des technologies *a priori* très différentes et porteuses d'avenir, telles que par exemple le calcul optique [Chavel 87].

Remerciements

Ce travail a été partiellement supporté par plusieurs contrats ETCA. Nous remercions particulièrement T. Bernard, C. Coquelet et E. Belhaire pour de fructueuses discussions.

Manuscrit reçu le 15 *janvier* 1988.

BIBLIOGRAPHIE

[Barnea 72] D. I. Barnea et H. F. Silverman, A Class of algorithms for fast digital image registration *IEEE Trans. on Computers*, 21, 2, Feb. 1972.

[Batcher 82] K. E. Batcher, Bit serial processing systems, *IEEE Trans on Computers*, C-31, 5, May 1982, p. 377-384.

[Bayer 73] B. E. Bayer, An optimum method for the two level rendition of continuous tone pictures, *Proc. IEEE Int. Conf. Comm. Conf. Record*, 1973, p. 26-11, 26-15.

[Bernard 88] T. Bernard, P. Garda, A. Reichart, B. Zavidovique et F. Devos, A family of analog neutral halftoning techniques, EUSIPCO, Grenoble, 6-9 septembre 1988.

[Basille 87] S. Basille, Les architectures lignes en traitement d'images 11° GRETSI, Nice, 1-5 juin 1987.

[Chavel 87] J. Taboury, J. M. Wang, P. Chavel, F. Devos et P. Garda, An Optical Cellular Processor Architecture. Part I: Principles. Applied Optics, Vol. 27, n° 9, mai 1988, pp. 1643-1650.

[Montanvert 87] A. Montanvert et J. M. Chassery, Architecture d'images discrètes binaires: outils géométriques, *Traitement du signal*, 4, 3, numéro spécial 1987, p. 205-216.

[Cole 83] T. N. Cole, Smart sensors-a combination of digital logic and photodiode arrays. Digest of papers of 19th Int'l Electronics convention and Exhibition (IREECON), Sydney, Australia, 5-9.9.83, p. 550-2.

[Danielsson 80] P. E. Danielsson, Euclidian distance mapping, *CGIP*, 14, 1980, p. 227-248.

[Danielsson 82] P. E. Danielsson, Vices and virtues of image parallel machines, *II Conf. on Image analysis and processing*, Selva di Fasano, Italy, Nov. 15-18, 1982.

[Danielsson 84] P. E. Danielsson, Algorithms and architectures for image processing *First image Symp.*, Biarritz, France, May 21-25, 1984.

[Duff 67] M. J. B. Duff, B. M. Jones et L. J. Townsend, Parallel processing pattern recognition system, *Nucl. Instr. Methods*, 52, 1967, p. 284-288.

[Duff 86] M. J. B. DUFF et T. J. FOUNTAIN, *Cellular logic image processing*, Academic Press, London, 1986.

[Flynn 72] M. J. FLYNN, Sarre computer and their effectiveness, *IEEE Tr. on Computers*, 21, 9, p. 948-960.

[Fountain 86] T. J. FOUNTAIN, Array architectures for iconic and symbolic image processing, *8th ICPR*, Paris 27-31 octobre 1986, p. 24-33.

[GAPP 84] NCR Corporation, Dayton, Ohio, Geometric Arithmetic Parallel Processor NCR45CG72.

[Garando 85] L. GARANDO, P. GARDA, B. ZAVIDOVIQUE et F. DEVOS, Une rétine B-code programmable, *L'Onde Électrique*, 65, 6, novembre 1985, p. 116-121.

[Garda 85-1] P. GARDA, B. ZAVIDOVIQUE, F. DEVOS, Cellular hardware for a NCL based vision, *3rd CAPAIDM*, 17-20 nov. 1985, Miami.

[Garda 85-2] P. GARDA, B. ZAVIDOVIQUE et F. DEVOS, Integrated cellular array performing neighborhood combinatorial logic on binary pictures, *11th ESSCIRC*, 16-18 sept. 1985, Toulouse.

[Garda 85-3] P. GARDA et B. ZAVIDOVIQUE, Reconnaissance de Formes par Traitements Combinatoires Locaux, *Cognitiva* 85, Paris, 4-7 juin 1985.

[Judice 76] C. N. JUDICE, J. F. JARVIS et W. H. NIKE, A survey of techniques for the display of continuous tone pictures on bilevel displays, *CGIP*, 5, 1976, p. 13-40.

[Linskog 86] B. LINSKOG et P. E. DANIELSSON, PICAP 3: a parallel processor timed for 3-D image operations, *8th ICPR*, Paris 27-31 octobre 1986, p. 1248-1250.

[Lyon 81] R. F. LYON, The optical mouse and an architectural methodology for smart digital sensors, *Proc. CMU Conf VLSI System Comput.*, Rockville.

[Mead 81] C. MEAD et L. CONWAY, *VLSI System design*, 1981.

[Milgram 88] TOSCA, *IEEE Tr. on PAMI* (à paraître).

[Nudd 78] G. R. NUDD, P. A. NYGAARD, G. D. THURMOND et S. D. FOUSE, A charge coupled device image processor for smart sensor applications, *Procs of SPIE*, 155, "Image Understanding Systems and Industrial Applications", San Diego, CA, USA, 30-31.8.78, p. 15-22.

[Preston 79] K. PRESTON *et al.*, Basics of cellular logic with some applications in medical image processing, *Proc. IEEE*, 27, 5, May 79, p. 826-856.

[Reddaway 79] S. F. REDDAWAY, The DAP approach, Infotech state of the Art Report on Supercomputers, Infotech Ltd, Maidenhead, 1979, 2, p. 309-329.

[Reichart 87] A. REICHART, P. GARDA, B. ZAVIDOVIQUE, Des rétines pour voir..., *MARI* 87, 18-22 mai 1987, Paris.

[Reeves 80] A. P. REEVES, On efficient global information extraction methods for parallel processors, *CGIP*, 14, 2, oct. 1980, p. 159-169.

[Rodriguez 86] H. RODRIGUEZ, L. GARANDO, P. GARDA, B. ZAVIDOVIQUE et F. DEVOS, Réseau de codeurs analogiques-numériques pour l'imagerie B-codée, *GCIA* 86, 3-4 octobre 1986, Orsay.

[Seitz 1984] SEITZ, Concurrent VLSI architectures, *IEEE Trans. Computers*, 33, 12, dec. 1984.

[Serra 82] J. SERRA, *Image analysis and mathematical morphology*, Academic Press, London 1982.

[Stoffel 81] J. C. STOFFEL et J. F. MORELAND, A survey of electronic techniques for two-level rendition of continuous tone pictures, *IEEE Tr. on Comm.*, 29, 1981, p. 1898-1925.

[Tanimoto 88] S. L. TANIMOTO, Iconic versus symbolic architectures, *Proc. of NATO Workshop on Real Time Object Recognition*, Maratea, Italy, 28.8.87-3.9.87, ASI, Springer Verlag, Series F, Vol. 42.

[Zavidovique 81] B. ZAVIDOVIQUE, Contribution à la vision des robots, *Thèse d'état*, Besançon, juillet 1981.

[Zavidovique 87] B. ZAVIDOVIQUE, A. LANUSSE et P. GARDA, Perception systems: some design issues, *Real-Time Object Measurement and Classification*, Ed. A. K. Jain, Springer Verlag, NATO ASI Series F, Vol. 42, Berlin 1988, pp. 93-110.