Modèle symbolique de la donnée de traitement d'images

DEJEAN Ph., DALLE P.
IRIT-2IL/TCI Université Paul SABATIER
118, route de Narbonne
31062 TOULOUSE Cedex
Tél : 61 55 63 09
Fax : 61 55 62 58
e-mail : dejean@irit.fr
dalle@irit.fr

Résumé – Beaucoup de travaux ont été effectués pour essayer de modéliser le comportement et le raisonnement d'un utilisateur du traitement d'images ou pour faciliter la mise au point de ses programmes. Dans cet article, nous présentons un nouveau modèle de représentation symbolique de la donnée en traitement d'images. Il s'insère dans le processus de descriptions d'objectifs en termes d'indices visuels reconstruites par planification d'opérateurs par un système de résolution de problème.
Cette modélisation de la donnée en cinq champs -trame, cache, index, signal et codage- permet de mieux rendre compte des transformations d'informations effectuées le long d'une chaînes d'opérateurs de traitement d'images. De plus, elle offre plusieurs possibilités dans le cadre de l'implémentation d'outils pour l'ajustement de telles chaînes ou dans le cadre de stratégies de planification intégrant des connaissances propres au traitement d'images.

Mots clés – Modèle de données – Représentation symbolique du traitement d'images – Génération d'applications dédiées – Planification

1 Introduction

Beaucoup de travaux ont été effectués pour essayer de modéliser le comportement et le raisonnement d'un utilisateur du traitement d'images ou pour faciliter la mise au point de ses programmes. Toutes les approches tendent à rendre compte dans leurs propres termes (algorithmiques, fonctionnels, par types, flux de données, informations symboliques, ...) des manipulations d'information effectuées par le traitement d'images. Ainsi, de manière plus ou moins effective et explicite selon les cas, quatre centres d'intérêts s'en dégagent :

- Expliciter ou modéliser le raisonnement, c'est à dire donner la possibilité à un utilisateur de mieux se servir des outils proposés et de mieux appréhender le traitement d'images.
- Faciliter la mise au point interactive de chaînes d'opérateurs en explicitant les erreurs et en fournissant des outils de visualisation et de correction.
- Planifier des chaînes d'opérateurs, c'est à dire automatiser le processus de conception de ces chaînes.
- Contrôler (piloter) l'exécution d'un programme générique, adaptatif, en sélectionnant les bonnes branches du graphe des opérateurs et en ajustant les paramètres.

Pour cela, divers modèles de la donnée de traitement d'images ont été proposés, la plupart du temps au regard des outils de modélisation du traitement d'images lui-même.
Après avoir relevé les différents aspects de ces modèles -situés dans leur contexte d'utilisation-, nous détaillerons une nouvelle approche de la donnée. Celle-ci a été implantée dans une nouvelle version du système ExTI [Ing92] qui constitue le noyau d'un processus de coopération Homme-Machine construisant des applications de traitement d'images. Il lui procure un formalisme bas-niveau de description d'image pouvant être couvert par un langage plus représentatif du domaine d'application.

2 Représentation des données

En examinant les procédés de modélisation des connaissances en traitement d'images, on constate que l'objectif est le plus souvent de représenter les opérateurs de traitement ce qui entraîne alors un choix de représentation des données qu'ils manipulent.
Pour décrire un opérateur, on peut chercher à expliquer :

Pourquoi a-t-il été conçu : c'est l'approche "fonctionnelle". Le traitement d'images est décrit en termes de fonctions et les opérateurs sont les outils informatiques qui implémentent ces fonctions. Une description symbolique des données peut être introduite, sous la forme de description d'entrées/sorties, pour effectuer des contrôles syntaxiques, mais elle n'intervient pas dans la sélection des opérateurs. Cette approche est utilisée, par exemple, dans tous les environnement interactifs de traitement d'images.

Comment a-t-il été conçu : c'est une décomposition algorithmique qui peut être décrite en termes de flux de données et être utilisée pour organiser de manière homogène la description d'une bibliothèque de traitement d'images. Elle permet une meilleure compréhension des algorithmes, l'insertion de nouveaux opérateurs dans la bibliothèque ou le développement d'une nouvelle fonction par analogie [Ser89]. Cette analyse algorithmique peut aussi viser un but de standardisation de représentation des techniques du traitement d'images et introduit la notion d'algèbre d'images (AFATL [1] Standard Image Algebra [Rit90, Dav93]).

Ce qu'il produit,

- *en termes de types de données.* Cette approche peut être utilisée,
  - pour standardiser les représentations des données et construire des environnements d'interprétation d'images, comme dans le programme [IUE94],
  - pour définir des langages de traitement d'images, construits sur des types abstraits de données comme dans SAPIN [Bel85], pour faciliter le développement de programmes spécifiques,
  - pour planifier des opérateurs, en structurant l'espace des données sous la forme d'un graphe d'entités. Les opérateurs sont alors des processus de transfert entre ces types, comme dans LLVE [Mat90].
- *en termes d'effets* : on construit un vocabulaire décrivant les caractéristiques géométriques extraites ou mises en valeur par les opérateurs [Sch89]. Si un objectif peut être décrit à l'aide de ces effets prédéfinis, des règles de raisonnement permettront alors de sélectionner les opérateurs qui les produisent.

Quelles sont ses propriétés, en termes mathématiques [Sch92]. Ceci permet de contrôler l'utilisation des opérateurs, en exploitant par exemple des propriétés d'idempotence ou d'associativité, pour contrôler le nombre d'itérations ou les combinaisons d'opérateurs.

Comment l'utilise-t-on. Cette information sera exploitée dans un système de contrôle d'exécution tel qu'OCAPI [Cle93]. Pour cela, on établit des liens, lors de la conception d'un système spécialisé, entre les objectifs à atteindre et les opérateurs résolvant cet objectif. La donnée de traitement d'images est perçue comme un ensemble de caractéristiques mesurables. Des règles expertes évalueront ces caractéristiques et le contexte d'application, pour sélectionner les opérateurs ou ajuster leurs paramètres [Tho93]. De même, le système CONNY [Lie92] intègre des mécanismes d'utilisation des données considérées comme éléments d'apprentissage ou de test, pour valider ou adapter les chaînes préétablies qu'il contient.

Signalons enfin que des descriptions originales de données de traitement d'images peuvent être utilisées :

- pour décrire les objets de l'image : par exemple Jungert [Jun93] utilise une algèbre d'image pour représenter symboliquement les relations spatiales entre objets.
- pour implanter des algorithmes particuliers et performants : modélisation de l'image binaire de Boomgard [vdB92], représentations pyramidales, en lacets, en files d'attentes, représentation polynômiale de GRASP [Ebe91] pour modéliser les couples (données,filtres), ...

Nous pensons que le formalisme utilisé pour représenter les données de traitement d'images doit être de très bas niveau, à la fois pour décrire de façon très fine le traitement effectué par les opérateurs et pour éviter d'établir a priori des liens entre opérateurs et objectifs de traitement d'images. Le modèle de description des opérateurs symboliques [Dal91] s'appuiera sur ce formalisme et permettra de propager les transformations opérées par une chaîne de traitements. La description de ces macro-opérateurs en termes d'effets produits permet d'introduire une expertise en traitement d'images (liens opérateurs-caractéristiques produites). Enfin la description des relations entre objets du domaine et ces indices visuels constitue l'expertise du domaine. Notre modèle évite d'introduire ces expertises dans le formalisme de base. Par contre il permet de les introduire à l'aide d'environnement de conception (SATI [Cap94]) pour produire des applications génériques ou dédiées, tout en gardant le lien avec le traitement d'images. Ceci permet d'une part de formuler des objectifs de traitement d'images et d'autre part de configurer le système dédié en planifiant la chaîne qui reconstruit la description de ces objectifs.

3 Modèle symbolique de la donnée

Le modèle présenté ici s'insère dans le processus de descriptions symboliques de données de traitement d'images exprimées en termes d'indices visuels. Ce processus permet ainsi de décrire des objectifs qui seront reconstruits par planification d'opérateurs par un système de résolution de problème.
Comme tout vecteur de communication une image peut être considérée comme étant composée de deux éléments :

Le support : c'est la grille rectangulaire ou hexagonale en traitement d'image qui va permettre de manipuler et de véhiculer le signal.

L'information : il peut s'agir d'une fonction mathématique, d'un simple codage, ou de tout autre système de représentation.

Lorsqu'on examine le travail effectué par un expert du domaine au cours d'un processus d'interprétation d'image, on s'aperçoit qu'il occulte, consciemment ou non, certaines parties de cette image. Les parties délaissées et celles qui déterminent une bonne interprétation peuvent être considérées comme des composantes particulières du support, masques ne laissant apparaître que l'information à traiter -un exemple simple est la mise à l'écart, lors d'une détection d'objets déterminés, des indices visuels touchant les bords d'une image.
C'est à partir de cette idée de cache laissant apparaître

1. Air Force Armament Laboratory

l'information pertinente et de la décomposition d'image en un couple (support, information) que nous avons développé ce nouveau modèle de représentation de l'objet image et des opérateurs qui le manipulent. Il rend mieux compte de la nature des transformations et de leurs incidences sur les objets. De plus, il offre des possibilités nouvelles pour la conception du planifieur de chaînes d'opérateurs et pour la mise en œuvre de plusieurs stratégies.

Nous utiliserons dans cet article les termes de trame, de cache, d'index, de signal et de codage pour décrire les éléments constituant la donnée D que nous souhaitons manipuler. Avant de détailler le modèle lui-même, il nous faut définir trois notions que nous retrouverons à tous les niveaux de la description : la dimension, le point et le référentiel.

3.1 Éléments mathématiques liés à D

Avant de décrire le modèle de la donnée -qui n'est pas nécessairement une image-, nous allons définir trois éléments qui vont entrer en jeu dans cette représentation.

3.1.1 La dimension

La dimension est le nombre d'éléments de la base de l'espace vectoriel supportant la donnée. Soit D^n une donnée de traitement d'images. Nous dirons que D^n est de dimension n, avec $n \in \mathbb{N}$, si tout élément π^n de D^n est déterminé de façon unique par un n-uplet appelé coordonnées de π^n dans D^n, ces coordonnées étant exprimées dans un repère R_{D^n} lié à D^n.
La donnée de traitement d'images peut correspondre alors à l'ensemble des plans d'une séquence d'images RVB (n=4), aux plans images RVB, aux deux plans d'une image complexe, ... (n=3), à une image de niveaux de gris, une coupe d'image couleur, ... (n=2), à une coupe d'image ou au tableau de valeurs, ... (n=1) ou enfin à une valeur de mesure ou à une intensité de pixels, ... (n=0). Ce dernier cas, particulier, permet d'uniformiser le modèle. En effet nous pouvons établir une analogie entre une considération (support, information) du signal image et la variable numérique -mesure sur une donnée, coordonnée, ...- au sens informatique du terme et qui consiste en un doublet (identificateur, valeur).

3.1.2 Le point

Le point désignera l'entité la plus élémentaire de la donnée de traitement d'images. Soit π^n un point de la donnée D^n à traiter, (plan image, image, ...) et n la dimension de D^n. Nous dirons que π^n est de dimension n s'il existe un unique n-uplet $(x_1, \cdots, x_n)$ de $\mathbb{N}^n$ déterminant les coordonnées de π^n dans R_{D^n}.

3.1.3 Le référentiel

Comme chaque donnée d'une même dimension possède son propre repère, nous allons par la suite appeler référentiel l'entité géométrique Υ^n de dimension n sur laquelle nous définirons le repère $R^n = (0, \vec{\imath_o}, \cdots, \vec{\imath_n})$. Pour toute dimension n fixée, ce référentiel sera considéré comme unique et invariant pour tout traitement, R^n étant lui-même invariant sur Υ^n.

$$\Upsilon^n = (n, R^n)$$

<u>Remarque :</u> *Pour plus de simplicité dans l'écriture, et lorsqu'il n'y aura pas d'ambiguïté, nous omettrons de mentionner la dimension n des entités rencontrées.*

3.2 Donnée de traitement d'images

Toute donnée D sera considérée sous la forme d'une structure composée des cinq éléments suivants :

3.2.1 La Trame

Une donnée D de traitement d'images est supportée par une trame τ, de même dimension, définie par l'ensemble des points π de D. Toute donnée étant finie, il existe pour chaque dimension i de D un intervalle fini $[I_i, S_i[$ de $\mathbb{N}$ tel que la composante x_i des coordonnées de π dans D prend ses valeurs dans cet intervalle.
Cette trame τ est donc la restriction finie du référentiel Υ de même dimension n (fig.1). La trame τ supportera le repère R_D lié à D.

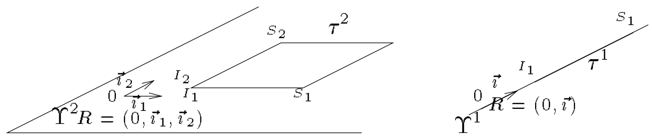

FIG. 1 - *Trames τ^2 et τ^1*

$$\tau^n = \{ \quad \pi^n = (x_1, \cdots, x_i, \cdots, x_n) \; / \; \forall n \in \mathbb{N}, \forall i \in [1, n], \exists (I_i, S_i) \in \mathbb{N}^2, x_i \in [I_i, S_i[\; \}$$

Nous appellerons taille de la trame τ de dimension n le n-uplet $\mathcal{T} = (t_1, \cdots, t_i, \cdots, t_n)$ dans lequel chaque t_i est égal à la différence des bornes du couple (I_i, S_i) déterminant l'intervalle de validité $[I_i, S_i[$ de la composante x_i de tout point π de τ.

$$\mathcal{T}(\tau^n) = \{ \quad (S_i - I_i)_{i \in [1,n]} \; / \; \forall \pi^n = (x_1, \cdots, x_i, \cdots, x_n) \in \tau^n, x_i \in [I_i, S_i[\; \}$$

Toute trame τ peut être alors déterminée en fonction des couples déterminant sa taille (fig.2). La notation proposée indique à la fois la taille de la trame et la position du repère R_D lié à la donnée. Il faut donc noter que le point origine d'un tel repère peut ne pas appartenir à la trame elle-même.

$$\tau^n = (\; (I_1, S_1), \cdots, (I_i, S_i), \cdots, (I_n, S_n) \;)$$

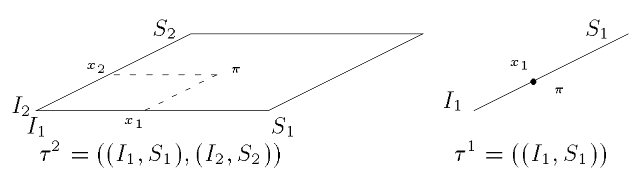

FIG. 2 - *Trame de dimension 2, Trame de dimension 1*

Nous appellerons T_n l'ensemble des trames τ de dimension n, c'est à dire l'ensemble des trames dont les points ont un système de coordonnées à n composantes : $\mathcal{T}(\tau^n) \in \mathbb{N}^n$.

$$T_n = \{ \; \tau^n \; / \; \forall \pi^n \in \tau^n, \pi^n = (x_1, \cdots, x_i, \cdots, x_n) \quad \}$$

3.2.2 Le Cache

□*Fonction cache*
Comme nous l'avons évoqué dans l'introduction de cette section, le processus d'interprétation d'images ne s'intéresse qu'à certaines parties de l'image. Ces zones d'intérêt correspondent à des éléments précis de l'application (concepts), identifiables par l'expert qui peut en donner une description -que nous noterons δ- formulée en termes d'indices visuels [Ing92].
La description d'un concept est alors considérée comme une WFF [2] composée de termes regroupant les propriétés ou les relations sur d'autres expressions. Les propriétés correspondent à une adaptation de la notion de sélecteurs décrits par Michalski [Mic84] qui définit le calcul des prédicats annotés. Elles sont composées d'un descripteur *(surface, niveau de gris, distance au bord, ...)* et d'un domaine déterminant leurs valeurs associées et constitué d'union d'intervalles semi-ouverts [x,y[:

$P = [p\ d]$: propriété (appelé sélecteur par Michalski)
p : descripteur de la propriété
d : domaine de la propriété

Le domaine de toute propriété P est inclus dans ce que nous nommerons le domaine de validité Δ_p associé au descripteur p de P. Pour donner un exemple, l'expression *((surface 100 200) (((nivg 0 128)))))* définit un ensemble de composantes connexes constituées des pixels de l'image d'intensité comprise dans l'intervalle [0,128[formant des amas dont les surfaces sont comprises dans l'intervalle [100,200[.

La fonction cache f_δ, à valeurs booléennes, va mettre en jeu de telles descriptions et va permettre de créer des caches applicables au référentiel de la donnée D traitée. Une telle fonction s'applique donc aux points π de Υ sur laquelle est définie D et détermine si la description de chacun d'eux, notée $\Lambda_\Upsilon(\pi)$, est une spécialisation -au sens de Michalski [Mic83]- de la description δ des zones d'intérêt de D.

$$f_\delta \left| \begin{array}{l} \Upsilon \longrightarrow \{\mathsf{True}, \mathsf{False}\} \\ \pi \longmapsto (\Lambda_\Upsilon(\pi) \subseteq \delta) \end{array} \right.$$

Si nous désignons par ∇_Υ l'ensemble des descriptions δ valides -sémantiquement correctes- dans Υ, nous pouvons définir l'ensemble F_Υ des fonctions caches définies sur Υ, c'est à dire l'ensemble des fonctions f_δ telles que δ soit un élément de ∇_Υ.

$$F_\Upsilon = \{\ f_\delta\ /\ \delta \in \nabla_\Upsilon\ \}$$

□*Cache*
Le cache à description δ, que nous noterons κ_δ, défini sur la donnée D, est déterminé par l'ensemble des points π de τ vérifiant la fonction cache f_δ. Il a donc la même dimension que la trame et fournit une information de type géométrique ou morphologique sur les éléments pertinents de la donnée (fig.3). Il correspond donc à un sous-ensemble -éventuellement disjoint- des points de la trame ayant tous les mêmes caractéristiques visuelles que l'on retrouve dans la description δ.

$$\kappa_\delta = \{\ \pi \in \tau\ /\ f_\delta(\pi) = \mathsf{True}\ \}$$

2. Well Form Formula

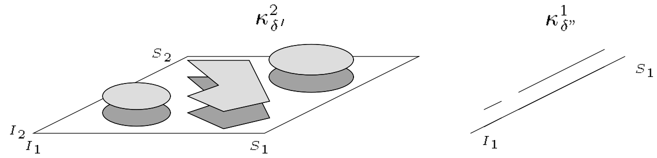

FIG. 3 - *Cache 2D à description δ', Cache 1D à description δ"*

Chaque descripteur de propriété p des prédicats de description possédant un domaine de validité Δ_p, il est possible de déterminer la propriété inverse $\overline{P}$ de P définie comme :

$$\overline{P} = \overline{[p\ d]} = [p\ \overline{d}] \ avec\ \overline{d} \cup d = \Delta_p$$

Par utilisation des règles du calcul des prédicats, il nous est donc possible de construire les complémentaires $\overline{\delta}$ des descriptions δ manipulées.

Nous appellerons $\overline{\kappa_\delta}$ le complémentaire du cache κ_δ défini sur la trame τ de la façon suivante (fig.4) :

$$\begin{aligned} \overline{\kappa_\delta} &= \{\ \pi \in \tau\ /\ f_\delta(\pi) = \mathsf{False}\ \} \\ &= \{\ \pi \in \tau\ /\ f_{\overline{\delta}}(\pi) = \mathsf{True}\ \} \\ &= \kappa_{\overline{\delta}} \end{aligned}$$

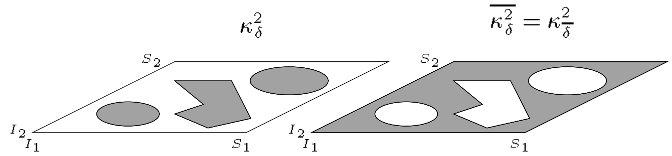

FIG. 4 - *Caches κ^2_δ et $\overline{\kappa^2_\delta}$*

Remarque : Il faut bien considérer le fait qu'un cache κ est la restriction à sa trame associée τ de l'application de sa fonction cache f_δ à tous les points π de l'entité géométrique sur laquelle est déterminée τ. La description δ peut donc très bien correspondre à des points n'appartenant pas à la trame et donc au cache (fig.5).

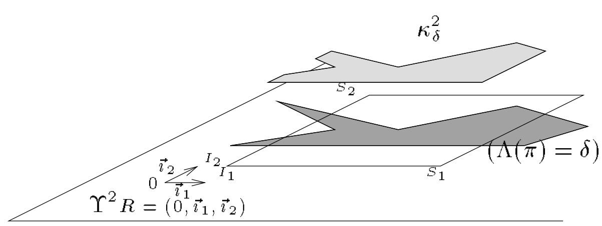

FIG. 5 - *Cache et fonction cache*

C'est ainsi qu'un utilisateur pourra décrire des entités d'une image en omettant (volontairement ou non) que celle-ci est une réduction spatiale de la scène observée. Ainsi dans la figure 6, une expression déterminant des objets sombres sera suffisante pour décrire les noyaux des cellules de l'image de gauche alors qu'elle sera trop générale pour les discriminer dans celle de droite, car elle inclura alors les artéfacts (petites tâches sombres externes aux cellules).
Ceci peut alors poser des problèmes lors de l'utilisation d'une même description sur un ensemble d'images extraites d'une même scène.

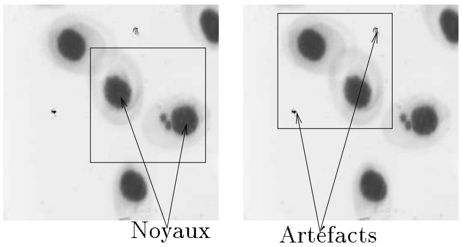

FIG. 6 - *Scène observée et images extraites*

Nous désignerons enfin par K_n l'ensemble des caches définis sur la trame τ, c'est à dire l'ensemble des κ_δ générés par les fonctions caches de F_{Υ^n}.

$$K_n = \{ \ \kappa_\delta \ / \ \delta \in \nabla_{\Upsilon^n} \quad \}$$

3.2.3 L'Index

La composante index ι d'une donnée D permet de regrouper les zones d'intérêt représentant un même concept par étiquetage des composantes connexes de son cache κ associé. Elle peut être représentée par une table d'index dont chaque élément fait référence à une composante connexe constituant le cache de la donnée dans un balayage déterminé (ligne, colonne, plan, séquence).
Les valeurs de cette table d'index déterminent le mode de considération de ces composantes connexes. En effet, ces dernières, à même description δ, peuvent être considérées dans leur globalité, indépendamment les unes des autres ou répondant à un même critère χ. Pour traduire ce dernier point, nous associons à ι ce critère et pouvons ainsi définir trois ensembles de tables d'index I_χ avec :

– $\chi \equiv \mathcal{G}l$ dans le cas d'un index de globalité (fig.7).

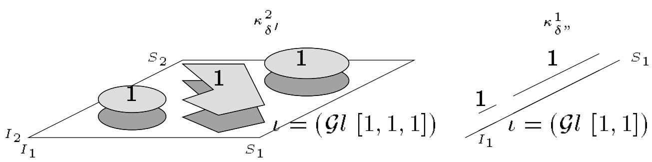

FIG. 7 - *Index de globalité*

– $\chi \equiv \mathcal{I}n$ dans le cas d'un index d'indépendance (fig.8).

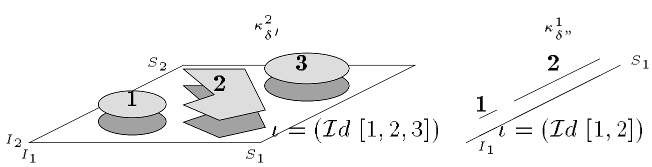

FIG. 8 - *Index d'indépendance*

– $\chi =< \textit{critère} >$ dans le cas d'un index spécifique (fig.9).

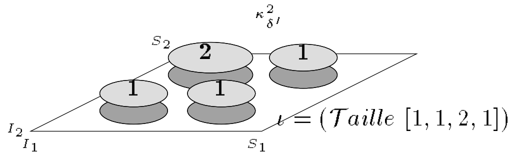

FIG. 9 - *Index spécifique χ="taille"*

Ainsi, l'index ι d'une donnée D possédant un cache κ est modélisée sous la forme du doublet ($\chi \ < table >$).

En complément à cette table d'index ι, nous construisons la table $\overline{\iota}$ contenant uniquement l'index nul, que nous noterons 0, permettant de désigner de façon globale le complémentaire $\overline{\kappa_\delta}$ de κ_δ.

$$\overline{\iota} = (\mathcal{G}l \ [0])$$

3.2.4 Le Signal

Les composantes de la donnée D vues au travers d'un cache κ indexé par une table ι, possèdent une caractéristique de représentation correspondant aux valeurs numériques associées aux intensités de leurs points.
Prenons une donnée image $\mathcal{D}$ sur laquelle nous définissons un cache κ dont la description δ associée correspond aux objets de l'image (fig.10). Les points de cette

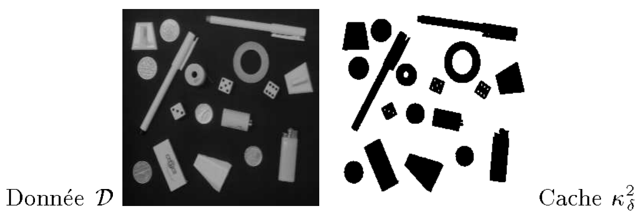

FIG. 10 - *Donnée et cache associé*

donnée, vus au travers de son cache, renferment des informations d'ordre sémantique que l'on doit représenter (fig.11). Nous appellerons ainsi signal -noté σ- la nature de l'intensité des points du cache reportés sur la donnée. Ainsi défini, ce signal devient un des éléments de construction des prédicats manipulés lors de la description des caches et possède donc un domaine de validité Δ_σ comme défini plus haut.

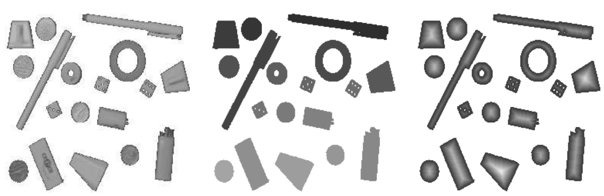

FIG. 11 - *Signaux "niveau de gris","label" et "distance aux bords"*

Comme précédemment, afin de compléter cette analyse, il faut considérer le non-cache $\overline{\kappa}$ de la donnée pour lui attribuer un signal $\overline{\sigma}$.

Nous désignerons par S_n l'ensemble des signaux des données pouvant être définies sur un référentiel Υ^n.

3.2.5 Le Codage

Le signal caractérisant la sémantique de l'information renfermée dans l'intensité du point de la donnée, il nous reste maintenant à considérer la valeur numérique de celle-ci. Considérant une donnée D sur laquelle est décrit un cache κ, nous appellerons $\mathcal{C}_\sigma$ la fonction de codage appliquée à tous les points π de κ sur Δ_σ, qui déterminera les valeurs de représentation de ce signal (fig.12).

$$\mathcal{C}_\sigma \left| \begin{array}{l} \kappa \longrightarrow \Delta_\sigma \\ \pi \longmapsto v \end{array} \right.$$

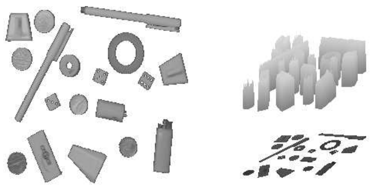

FIG. 12 - *Codage du signal "niveau de gris"*

Pour une donné D déterminée possédant le signal σ sur un cache κ, nous définirons le codage de σ sur D, que nous noterons ω, comme étant le sous-ensemble de Δ_σ correspondant aux valeurs numériques que *peuvent* prendre les intensités des points π de ce cache. Ainsi nous aurons les inclusions suivantes :

$$\{v = \mathcal{C}_\sigma(\pi), \forall \pi \in \kappa\} \subseteq \omega \subseteq \Delta_\sigma$$

Par analogie nous définirons $\overline{\omega}$ comme étant le codage des points appartenant à $\overline{\kappa}$ et dont le signal est $\overline{\sigma}$. Ainsi, quelque soit le couple $(\sigma,\overline{\sigma})$ défini sur D nous devrons avoir :

$$\omega \cup \overline{\omega} = \Delta_\sigma$$

3.3 Donnée de traitement d'images

Une donnée symbolique de traitement d'image est définie par une structure (fig.13) comportant toutes les informations développées précédemment : la trame τ, la description du cache d'interprétation δ, l'index ι, le signal σ et l'ensemble des valeurs de codage de ce signal ω.

De plus, à tous les niveaux de l'étude des données de traitement d'images, nous pouvons considérer qu'une donnée D est liée à une donnée complémentaire que nous noterons $\overline{D}$ de même trame τ et composée des éléments $\overline{\kappa}$, $\overline{\iota}$, $\overline{\sigma}$ et $\overline{\omega}$.

Si l'on considère maintenant qu'une session de traitement d'image débute par une transformation d'information, il est nécessaire de définir la structure de la donnée λ que nous appellerons donnée vide et que nous pouvons représenter par :

$$\lambda = \left\{ \begin{array}{l} \tau = \cdot \\ \kappa = \cdot \\ \iota = \cdot \\ \sigma = \cdot \\ \omega = \cdot \end{array} \right\} \text{ avec } \overline{\lambda} = \left\{ \begin{array}{l} \overline{\kappa} = \cdot \\ \overline{\iota} = \cdot \\ \overline{\sigma} = \cdot \\ \overline{\omega} = \cdot \end{array} \right\}$$

<u>*Remarque :*</u> Afin de désigner une composante χ d'une donnée D et pour éviter des confusions dans les écritures nous utiliserons la notation $D.\chi$. Pour étendre cette notation, nous noterons $\overline{D}.\overline{\chi}$ (ou plus simplement $D.\overline{\chi}$) la composante $\overline{\chi}$ du complément de la donnée D.

$$D = \left\{ \begin{array}{l} \tau = ((0,512),(0,512)) \\ \kappa = t \\ \iota = (\mathcal{G}l\ [1]) \\ \sigma = nivg \\ \omega = [0, 256[\end{array} \right\} \quad \overline{D} = \left\{ \begin{array}{l} \overline{\kappa} = () \\ \overline{\iota} = \cdot \\ \overline{\sigma} = \cdot \\ \overline{\omega} = \emptyset \end{array} \right\}$$

FIG. 13 - *Structure d'une donnée image*

4 Validation et exploitation

Le système ExTI s'inscrit dans un processus général de conception de programmes de traitement d'images.

L'idée maîtresse est que lors de la manipulation de tels logiciels par un utilisateur non spécialiste du traitement d'images mais expert dans son domaine d'application, il existe deux types de connaissances bien distinctes qu'il faut pouvoir mettre en correspondance - connaissance des techniques du traitement d'images et connaissances du domaine d'application.

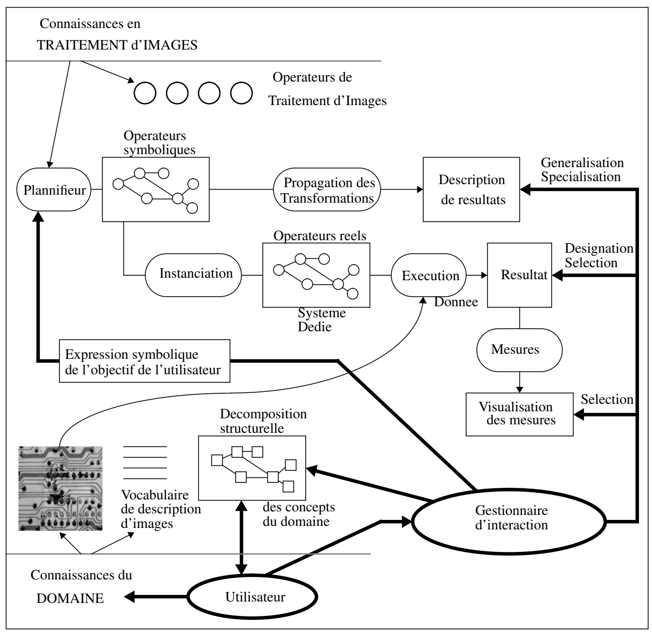

FIG. 14 - *Processus coopératif de conception de programmes*

La figure 14 montre le processus de conception de programmes de traitement d'images séparant les deux types de connaissances mises en œuvre. Ce processus est traduit par l'association des systémes ExTI, qui gére les connaissances en traitement d'images en mettant en jeu des modules de planification et d'exécution de chaînes d'opérateurs, et SATI chargé d'exploiter les connaissances de l'utilisateur dans son propre domaine.

Dans ce double système ExTI-SATI nous avons limité l'action de l'utilisateur (traits gras sur le schéma) à la formulation d'objectifs sous la forme d'expressions symboliques exprimées en termes d'indices visuels et à des interactions (désignation, généralisation et spécialisation) sur les données produites par le système.

La résolution d'un problème d'interprétation, du point de vue de l'utilisateur, consiste alors à déterminer les relations entre les objets instanciés du modèle du monde et les éléments de l'image -produits par les opérateurs- que nous appelons indices visuels ; ce sera au système de prendre en charge la reconstruction de l'expression de l'objectif par planification d'opérateurs symboliques, c'est à dire de construire la chaîne d'opérateurs résolvant ce problème.
Pour répondre correctement à ces objectifs, les modèles de données et d'opérateurs implantés dans ExTI doivent donc satisfaire à plusieurs objectifs importants concernant les transformations d'information et la planification d'opérateurs.

4.1 Les transformations d'information

Lorsqu'une chaîne de traitement est construite mais ne répond pas exactement à l'objectif souhaité (fig.6), il est nécessaire de fournir des renseignements et des outils de rectification afin de faciliter la mise au point de l'objectif formulé. L'utilisation de ce nouveau modèle de données permet, comme dans le modèle précédent, de rendre bien compte des transformations d'informations effectuées par les opérateurs tout au long de la chaîne de traitement, mais, du fait de la séparation des différents éléments constituant la donnée, il fournit aussi des informations détaillées concernant le ou les champs transformés par les opérateurs.

Pour illustrer ceci, prenons l'exemple simple suivant (fig.15), trace d'une planification d'objectif [3].
La chaîne résultat comporte une séquence d'opérateurs symboliques [fs_non, fs_distance, fs_seuiller, fs_ou] traduisant l'opérateur de dilatation binaire par élément structurant de rayon 1 qui sera considéré comme un macro-opérateur pour le système (fig.16).

Comme nous pouvons le constater, les représentations des données produites au cours de la construction de cette chaîne traduisent bien les éléments mis en jeu par ces transformations mais surtout les divers champs d'action de ces opérateurs.
Ainsi l'opérateur symbolique de mesure des distances aux bords de la composante (fs_distance correspondant à l'opérateur *vdistance* de KHOROS [Uni92]) n'effectue qu'un changement de la valeur du champ signal de la donnée entraînant évidemment un changement de la valeur du codage ω. C'est exactement ce qui est traduit dans les représentations des données $g28$ et $g30$.

$$g28 = \left\{ \begin{array}{l} \tau = ((0\ 256)\ (0\ 256)) \\ \kappa = (((nivg\ 113\ 256))) \\ \iota = (\mathcal{G}l\ :g28:) \\ \sigma = \mathbf{vab} \\ \omega = \mathbf{(1\ 2)} \end{array} \right\}$$

$$\overset{fs\_distance}{\longmapsto} \quad g30 = \left\{ \begin{array}{l} \tau = ((0\ 256)\ (0\ 256)) \\ \kappa = (((nivg\ 113\ 256))) \\ \iota = (\mathcal{G}l\ :g28:) \\ \sigma = \mathbf{dist} \\ \omega = \mathbf{(1\ 128)} \end{array} \right\}$$

$$\overline{g28} = \left\{ \begin{array}{l} \overline{\kappa} = (((nivg\ 0\ 113))) \\ \overline{\iota} = (\mathcal{G}l\ 0) \\ \overline{\sigma} = vab \\ \overline{\omega} = (0\ 1) \end{array} \right\}$$

$$\overset{fs\_distance}{\longmapsto} \quad \overline{g30} = \left\{ \begin{array}{l} \overline{\kappa} = (((nivg\ 0\ 113))) \\ \overline{\iota} = (\mathcal{G}l\ 0) \\ \overline{\sigma} = vab \\ \overline{\omega} = (0\ 1) \end{array} \right\}$$

Une telle décomposition n'apparaît que lors de la phase de planification, sans utilisation d'opérateurs comme la dilatation que nous pouvons qualifier de macro-opérateurs du point de vue des transformations d'informations effectuées.

```
OBJECTIF = #[ ((0 256) (0 256))
              (((dist 1 2) (((nivg 113 256)))) ((nivg 0 113)))
              (Gl 1)
              vab
              (1 2)]


Planification du but principal :
================================
Phase No1 : sur les plans generes
Phase No2 : sur les sous-buts etiquettes reconnus
Phase No3 : sur les autres sous-buts
Phase No4 : sur la racine

 g9 = (fs_lire g7 im)
      Trame : ((0 256) (0 256))
      Cache : t          ~Cache : ()
      Index : (Gl 1)     ~Index : ()
      Signal : nivg      ~Signal : ()
      Codage : (0 256)  ~Codage : ()
g11 = (fs_seuiller g9 f0 f113)
      Trame : ((0 256) (0 256))
      Cache : (((nivg 0 113))) ~Cache : (((nivg 113 256)))
      Index : (Gl :g11:)        ~Index : (Gl 0)
      Signal : vab              ~Signal : vab
      Codage : (1 2)            ~Codage : (0 1)

...

g28 = (fs_non g11)
      Trame : ((0 256) (0 256))
      Cache : (((nivg 113 256))) ~Cache : (((nivg 0 113)))
      Index : (Gl :g28:)          ~Index : (Gl 0)
      Signal : vab                ~Signal : vab
      Codage : (1 2)              ~Codage : (0 1)
g30 = (fs_distance g28)
      Trame : ((0 256) (0 256))
      Cache : (((nivg 113 256))) ~Cache : (((nivg 0 113)))
      Index : (Gl :g28:)          ~Index : (Gl 0)
      Signal : dist               ~Signal : vab
      Codage : (1 128)            ~Codage : (0 1)
g32 = (fs_seuiller g30 f1 f2)
      Trame : ((0 256) (0 256))
      Cache : (((dist 1 2) (((nivg 113 256)))))
                         ~Cache : (((dist 2 128) (((nivg 113 256))))
                                   ((nivg 0 113)))
      Index : (Gl :g32:)   ~Index : (Gl 0)
      Signal : vab         ~Signal : vab
      Codage : (1 2)       ~Codage : (0 1)
g36 = (fs_ou g32 g11)
      Trame : ((0 256) (0 256))
      Cache : (((dist 1 2) (((nivg 0 113))))
              ((nivg 113 256)))
                         ~Cache : (((dist 2 128) (((nivg 0 113)))))
      Index : (Gl :g36:)   ~Index : (Gl 0)
      Signal : vab         ~Signal : vab
      Codage : (1 2)       ~Codage : (0 1)
Solutions : (g36)
```

FIG. 15 - *Trace de la planification*

3. Notations dans la trace : ~x traduit $\overline{x}$, :gx: représente la table d'index à instancier lors de l'exécution et un doublet de valeurs $x\ y$ traduit l'intervalle semi-ouvert $[x, y[$.

La séquence rencontrée devra être alors reconnue au moment de l'écriture du script instancié afin d'être remplacée par l'opérateur réel de dilatation.

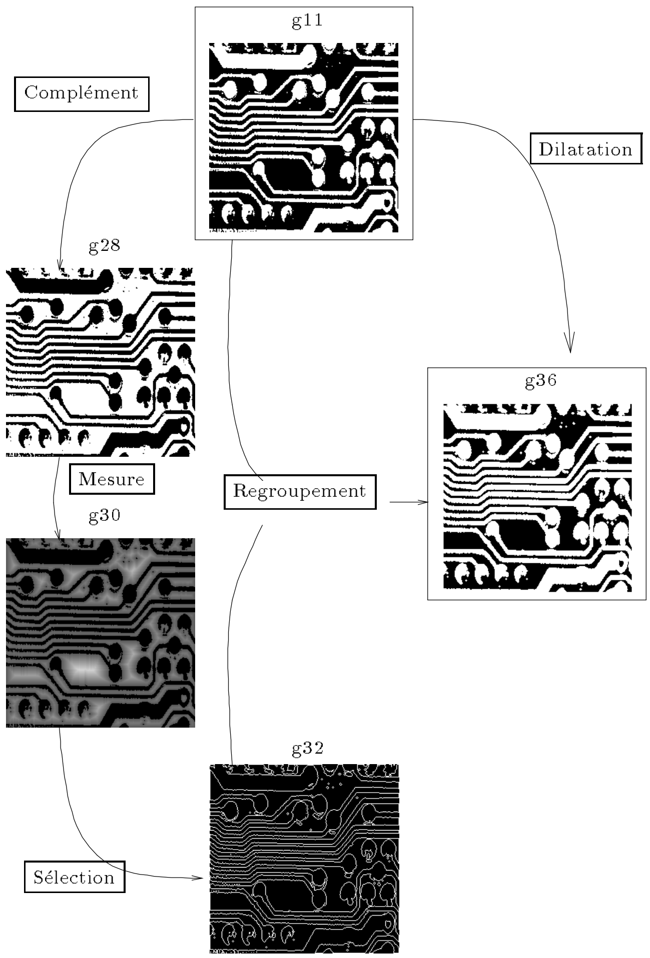

FIG. 16 - *Décomposition de l'opérateur de dilatation*

La base d'opérateurs symboliques étant ainsi réduite, la chaîne résultat contient de nombreux opérateurs élémentaires, permettant d'avoir accès à des informations fines sur les modifications des champs des données manipulées. Ceci facilite en particulier l'implantation de techniques de fusion de plans déjà établis.

Ce modèle, offrant la possibilité d'accéder aux caractéristiques visuelles des données manipulées, permet de représenter des notions particulières, comme par exemple les voisinages en utilisant correctement les champs trame et codage. Prenons cet exemple et considérons deux modules d'étiquetage de composantes connexes opérant avec les voisinages simples 4-voisins et 8-voisins sur une même donnée -reprenons g28. Les trois champs index, signal et codage sont modifié : [4]

$$g28 = \left\{ \begin{array}{l} \tau = ((0\ 256)\ (0\ 256)) \\ \kappa = (((nivg\ 113\ 256))) \\ \iota = (\mathbf{Gl}\ \mathbf{:}g28\mathbf{:}) \\ \sigma = \mathbf{vab} \\ \omega = \mathbf{(1\ 2)} \end{array} \right\}$$

$$\overset{fs\_label4}{\longmapsto} \quad I_4 = \left\{ \begin{array}{l} \tau = ((0\ 256)\ (0\ 256)) \\ \kappa = (((nivg\ 113\ 256))) \\ \iota = (\mathbf{In}\ \mathbf{:}I_4\mathbf{:}) \\ \sigma = \mathbf{etiq4} \\ \omega = \mathbf{(1\ 32769=}\ \lceil \frac{(256\times 256)+1}{2} \rceil + 1) \end{array} \right\}$$

$$g28 = \left\{ \begin{array}{l} \tau = ((0\ 256)\ (0\ 256)) \\ \kappa = (((nivg\ 113\ 256))) \\ \iota = (\mathbf{Gl}\ \mathbf{:}g28\mathbf{:}) \\ \sigma = \mathbf{vab} \\ \omega = \mathbf{(1\ 2)} \end{array} \right\}$$

$$\overset{fs\_label8}{\longmapsto} \quad I_8 = \left\{ \begin{array}{l} \tau = ((0\ 256)\ (0\ 256)) \\ \kappa = (((nivg\ 113\ 256))) \\ \iota = (\mathbf{In}\ \mathbf{:}I_8\mathbf{:}) \\ \sigma = \mathbf{etiq8} \\ \omega = \mathbf{(1\ 16385=}\ (\lceil \frac{256+1}{2} \rceil \times \lceil \frac{256+1}{2} \rceil) + 1) \end{array} \right\}$$

Avec de tels opérateurs, les champs $I_4.\iota$ et $I_8.\iota$ ne peuvent être instanciés correctement (table d'index) que lors de l'exécution de mesures effectives sur la donnée réelle ; ils ne sont donc pas utilisables pour discriminer les voisinages utilisés.
Les couples de champs $(I_x.\sigma, I_x.\omega)_{x=4,8}$ permettent par contre cette discrimination. Ainsi, lors d'une opération $\mathcal{M}$ de mesures (surface, moments, ...) sur les composantes étiquetées de I_x, la trame de la donnée résultat $J_x = mesure(\mathcal{M}, I_x)$ de dimension $n = 1$ dépendra du champ codage $I_x.\omega$:

$$J_4 = mesure(\mathcal{M}, I_4) \Longrightarrow J_4.\tau = (I_4.\omega) = ((1\ 32769))$$

$$J_8 = mesure(\mathcal{M}, I_8) \Longrightarrow J_8.\tau = (I_8.\omega) = ((1\ 16385))$$

La transformation d'information issue de l'étiquetage effectué est donc correctement conservée.

4.2 Exploitation du modèle

Planification :
La principale utilisation de ce modèle réside dans la possibilité de planifier des chaînes d'opérateurs. Ce système de planification, ExTI, met en œuvre un module de filtrage permettant de constater la distance qui sépare la donnée produite à l'objectif. Du fait de la séparation des cinq éléments de la donnée, le planifieur va devoir fusionner les cinq réponses du nouveau filtrage de façon à évoluer selon une stratégie qui, entre autre, implémente pour le traitement d'images des notions adaptées de *General Problem Solver* [Win77].

Une autre possibilité d'exploitation consiste à effectuer des planifications d'opérateurs en tenant compte d'objectifs imprécis ne donnant aucun renseignement sur certains champs de la donnée à produire. Ceci a pour conséquences directe, lors de la phase de filtrage, de ne pas considérer ces champs sur les données créées, et donc de forcer l'utilisation d'opérateurs semblables à des transformations près -exemple : fs_label4 et fs_label8.

4. $\lceil x \rceil$: Partie entière de x.

A l'utilisateur de choisir alors la ou les chaînes de traitement fournissant des données résultats voisines de l'objectif à atteindre. Ces dernières étant décrites pleinement dans leur représentation symbolique, l'objectif peut être facilement ajusté ou complété.

Raisonnement :
Parce qu'il peut fournir un mode de structuration des opérateurs autre que fonctionnel, algorithmique ou basé sur les types de données, ce modèle offre plusieurs axes de mise au point de stratégies du raisonnement en traitement d'images en étudiant les différentes familles d'opérateurs travaillant sur ces champs de la donnée et plus généralement sur les ensembles T_n, K_n, I_χ et S_n définis plus haut.

Validation de la chaîne :
Grâce à sa possibilité de rendre compte finement des transformations d'informations et à les propager, ce modèle permet de construire des outils de validation d'une chaîne de traitement, qu'elle ait été obtenue par planification ou de façon interactive.

Contrôle d'exécution :
Enfin, ce modèle permet de donner les moyens de contrôler l'exécution de ces chaînes d'opérateurs en facilitant l'introduction de connaissances sur leurs enchaînements ou en montrant les incidences des modifications d'opérateurs -ajustement de paramètres, substitution d'opérateurs d'une même famille,... - sur toute la longueur de la chaîne.

5 Conclusion

Pour permettre la conception d'un graphe d'opérateurs, un logiciel de traitement d'images met à la disposition de ses utilisateurs un ensemble d'outils de manipulation des données images. C'est le cas pour KHOROS [Uni92] ou encore VISILOG [Noe92] dans lesquels les opérateurs sont regroupés selon leurs fonctionnalités ou selon les types des données utilisées ou produites. Pour exploiter de tels systèmes, l'utilisateur doit nécessairement appréhender le traitement d'images, même si des aides -contextuelle ou en ligne- lui sont proposées. Qu'il construise sa chaîne d'opérateurs selon une approche descendante -la plus naturelle-, ascendante ou mixte (fig.17), cet utilisateur reste confronté à trois types de difficultés : raisonner en termes d'opérateurs de traitement d'images, configurer la chaîne et justifier les résultats -c'est à dire garantir que la chaîne fournit le bon résultat dans d'autres cas que celui qui a servi à la générer.
Parallèlement, pour développer un système de conception automatique de chaînes d'opérateurs, il est nécessaire de pouvoir modéliser les connaissances du traitement d'images et de représenter les entités manipulées, selon un formalisme identique.

ExTI, associé à une interface utilisateur du type SATI, rassemble les connaissances d'un expert en traitement d'images et les restitue dans des termes accessibles aux divers utilisateurs. Le modèle que nous avons décrit lui fournit une description fine des données en termes d'in-

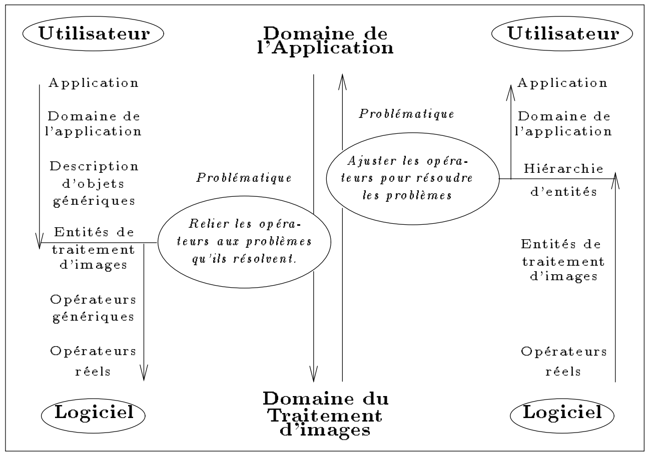

FIG. 17 - *Approches Descendante / Ascendante*

dices visuels permettant de montrer de façon précise les transformations effectués par les opérateurs. Nous distinguons ainsi les cinq éléments de la données qui correspondent à une représentation visuelle de l'image : la trame -ensemble de points de la données-, le cache -dont la description permet de regrouper ces points en zones d'intérêt, l'index -indiquant le mode de considération de chaque amas de points du cache, le signal -représentant la nature de l'information liée à l'intensité de ces mêmes points- et le codage -correspondant aux valeurs de ces intensités.

Cette décomposition des données entraîne une modélisation particulière des opérateurs qui les manipulent. L'intégration de nouveaux opérateurs dans le système ExTI passera par leur représentation sous la forme d'un ensemble de cinq fonctions de transformations affectant les champs du modèle de la donnée. Un tel modéle permet donc de représenter aisément les opérateurs du type morphologiques, les opérateurs locaux de différentiation ainsi que ceux effectuant des mesures sur les données car les transformations qu'ils opérent sont facilement interprétable en termes d'indices visuels. Les opérateurs plus complexes sont considérés comme des macro-opérateurs dans ExTI (voir 4.1 pour le principe). Ils seront effectivement regénérés par le module de planification. Tous ces opérateurs pourront être alors structurés selon des critères particuliers relatifs aux champs de la donnée atteints par ces transformations. De là, ce modèle peut être exploité pour faciliter l'aide à la mise au point de programme, pour implémenter des stratégies de planification spécifiques ou pour développer des outils d'apprentissage du traitement d'images.

Références

[Bel85] A. Belaïd. – Un langage de traitement d'images fondé sur des concepts de types abstraits. *Techniques et Sciences Informatiques*, vol. 4, n° 3, 1985, pp. 309–323.

[Cap94] O. Capdevielle et P. Dalle. – Image Processing chain construction by interactive goal specification. *In : ICIP, Austin Texas (U.S.A.)*, 1994.

[Cle93] V. Clement et M. Thonnat. – A knowledge-based approach to integration of image processing procedures. *CVGIP : Image Understanding*, vol. 57, n° 2, Mars 1993, pp. 166–183.

[Dal91] P. Dalle et J.-M. Inglebert. – Modèle de description d'opérateurs de traitement d'images. *In : 8e RFIA, Lyon-Villeurbanne (France)*, 1991, pp. 257–266.

[Dav93] J.L. Davidson. – Classification of Lattice Transformations in Image Processing. *CVGIP : Image Understanding*, vol. 57, n° 3, Mai 1993, pp. 283–306.

[Ebe91] D. Eberly et D. Wenzel. – Adaptation of Group Algebras to Signal and Image Processing. *CVGIP : Graphical Models and Image Processing*, vol. 53, n° 4, Juillet 1991, pp. 340–348.

[Ing92] J. M. Inglebert. – *Planification automatique de chaînes de traitement d'images.* – Thèse de 3e cycle, Université P. Sabatier de Toulouse (France), 1992.

[IUE94] *Image Understanding Environment, Program – IUE Overview Document.* – ftp.aai.com (192.190.241.40), 1994.

[Jun93] E. Jungert et S.K. Chang. – An Image Algebra for Pictorial Data Manipulation. *CVGIP : Image Understanding*, vol. 58, n° 2, Septembre 1993, pp. 147–160.

[Lie92] C. E. Liedtke et A. Blömer. – Architecture of the knowledge based configuration system for image analysis CONNY. *In : 11e ICPR, The Hague (The Netherlands)*, 1992, pp. 375–378.

[Mat90] T. Matsuyama et V. Hwang. – *SIGMA : a knowledge-based aerial image understanding system.* – Plenum Press, 1990.

[Mic83] R.S. Michalski. – A theory and methodology of inductive learning. *Artificial Intelligence*, vol. 20, n° 2, 1983, pp. 111–161.

[Mic84] R.S. Michalski, J.G. Carbonell et T.M. Mitchell. – A theory and methodology of inductive learning. *In : Machine learning : an artificial intelligent approach*, pp. 83–134. – Berlin : Springer-Verlag, 1984.

[Noe92] Noesis, 13 Av M Saulnier, 78140 Vélisy. – *Visilog 4 Reference Guide*, 1992.

[Rit90] G.X. Ritter, J.N. Wilson et J.L. Davidson. – Image Algebra : An Overview. *CVGIP*, vol. 49, n° 3, Septembre 1990, pp. 297–331.

[Sch89] M. Schmitt. – Mathematical Morphology and Artificial Intelligence : An automatic programming system. *Signal Processing*, vol. 16, n° 4, 1989, pp. 389–401.

[Sch92] M. Schmitt. – *La morphologie mathématique en analyse d'images.* – Rapport technique n° 4, Thomson-CSF, Décembre 1992.

[Ser89] V. Serfaty–Dutron, B. Hoeltzener–Douarin et B. Zavidovique. – Exemple de conception mécanique d'Opérateurs de T.I. *In : 7e RFIA, Paris (France)*, 1989, pp. 75–88.

[Tho93] M. Thonnat, V. Clement et J. van den Elst. – *Supervision of Perception Tasks for Autonomous Systems : the OCAPI Approach.* – Rapport de recherche n° 2000, INRIA, Juin 1993.

[Uni92] University of New Mexico. – *Khoros Reference Manual*, 1992.

[vdB92] R. van denBoomgaard et R van Balen. – Methods for Fast Morphological Image Transforms Using Bitmapped Binary Images. *CVGIP : Graphical Models and Image Processing*, vol. 54, n° 3, Mai 1992, pp. 252–258.

[Win77] P.H. Winston. – *Artificial Intelligence.* – Addison-Wesley Publishing Compagny, 1977, pp. 130–143.