Adéquation algorithme - architecture pour le traitement multimédia embarqué

S. Roux

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

N° attribué par la bibliothèque

|__|__|__|__|__|__|__|__|__|__|

THESE

pour obtenir le grade de

DOCTEUR DE L'INPG

Spécialité : Microélectronique

préparé aux laboratoires **France Télécom R&D DIH/OCF** et **TIMA**

dans le cadre de l'**Ecole Doctorale Electronique, Electrotechnique, Automatique,Télécommunications, signal**

présentée et soutenue publiquement

par

Sébastien ROUX

le 22 Janvier 2002

Titre :

ADEQUATION ALGORITHME - ARCHITECTURE POUR LE TRAITEMENT MULTIMEDIA EMBARQUE

Directeur de thèse :

Alain GUYOT

JURY

M. Marc RENAUDIN,	Président
M. Habib MEHREZ,	Rapporteur
M. Michel PAINDAVOINE,	Rapporteur
M. Gérard BOUVIER,	Examinateur
M. Alain GUYOT,	Directeur de thèse
M. Eric PETIT,	Co-encadrant

ISBN : 2-913329-83-7
ISBNE : 2-913329-84-5

Remerciements

Cette thèse a été réalisée dans le cadre d'un partenariat entre le groupe Concurrent Integrated Systems(CIS) du laboratoire Techniques de l'Informatique et de la Micro-électronique pour l'Architecture de l'ordinateur (TIMA) et France Télécom R&D.
Mes travaux de recherche ont été menés au sein du groupe Plate-forme d'Intégration de traitement Multimedia (PIM) du laboratoire Objets Communicants et Faisabilité de système (OCF).
Je tiens tout d'abord à remercier M. Patrice Senn pour m'avoir acceuilli, M. Alain Guyot pour avoir accepté la direction de cette thèse et M. Dominique Barthel pour m'avoir proposé le sujet de cette thèse.

J'adresse mes remerciements au Professeurs Habib Merhez et au Professeur Michel Paindavoine pour avoir accepté d'être rapporteur de ma thèse, le Professeur Gérard Bouvier pour sa participation à mon jury.
Je remercie le Professeur Marc Renaudin pour avoir bien voulu me faire l'honneur de présider le jury de cette thèse
Je remercie M. Elbay Bourennane pour m'avoir donné un avis éclairé sur mon mémoire.

Par ailleurs, je remercie la société Arexsys (désormais Valiosys) pour avoir acceptée une collaboration sur les aspects architecturaux étudiés dans le cadre de ma thèse, et plus particulièrement, M. Jerome Bortolami pour son aide et son amabilité.
Je remercie M. Michel Harrand de STMicroelectronics pour m'avoir permit d'utiliser une plateforme matérielle réelle (IVT) afin de tester les algorithmes développés dans cette thèse.

Je tiens à exprimer ma plus grande gratitude à Eric Petit qui a largement contribué à l'aboutissement de cette thèse. Je lui suis très reconnaissant pour son aide, ses conseils, ses critiques, ses qualités humaines et ses encouragements. J'ai été très heureux de travailler sous sa responsabilité, et encore plus de pouvoir grimper avec lui ! Espérons que ca dure !

Je tiens également à remercier Franck Mamalet, pour sa disponibilité et son agenda des concerts, Sylvie Vidal, Bruno Galilée, Fano Ramparani, Jean Paul Laval, Pascal Perret et Patrice Le Scan pour l'ambiance qu'ils ont su créer.
Je remercie toutes les personnes du site de Meylan que j'ai eu la joie de connaître, et plus particulièrement, Martine et Cathy pour leur bonne humeur permanente et leur assistance dans les tâches administratives.

J'aimerais également remercier tous ceux qui m'ont soutenu et avec qui j'ai passé de bons moments de détentes, ils sont : Mathias, Anna, Didier, Lise, Mathilde, Reza, Olivier, François, Nadège, Sébastien, Stéphane, Marine, Ketltoum, Laurent, Xavier, Jana, Tarek, et bien d'autres (pardon pour l'oubli !), mes parents, ma soeur et Thierry.
Enfin, je souhaite remercier Pascale qui a su me supporter au quotidien, surtout ces derniers temps. Et, merci a vous qui lisez ces lignes !

Table des matières

INTRODUCTION

Les débits annoncés pour les futures normes de télécommunication, ainsi que le degré d'intégration atteint avec les dernières technologies CMOS, ou encore les performances des derniers DSP du commerce, sont autant de facteurs déterminant pour l'implantation de nouvelles applications dans les terminaux mobiles (téléphone portable, assistant personnel,...). Durant ces dernières années de nouveaux modes d'utilisation des téléphones portables sont également apparus. Ces terminaux ne servent plus seulement à communiquer par la voix mais permettent des échanges de données. Les messages courts (SMS) en sont un bon exemple. La connexion des terminaux à Internet est déjà une réalité, en Europe avec le WAP (Wireless Application Protocol), et au Japon avec le "i-mode" qui connaît un franc succès (60 millions d'abonnés en 2001). Avec des débits atteignant 384 kbit/s en voix descendante, et 64 kbit/s en voix montante, pour les premiers déploiements du réseau UMTS. Un service de visiophonie mobile est désormais envisageable, d'où l'intérêt d'en étudier la faisabilité, en particulier au niveau de l'intégration du codage vidéo. C'est dans ce contexte que se situe cette thèse dont les objectifs sont d'étudier la faisabilité d'un service de visiophonie mobile et, parallèlement, de proposer une méthodologie de conception de systèmes multimedia embarqués.

Cette thèse s'intègre dans les travaux de recherche du laboratoire DIH/OCF (Direction des Interactions Humaines / Objets Communicants et Faisabilité de systèmes) de France Télécom R&D sur la faisabilité de systèmes pour les terminaux mobiles de prochaine génération. Par ailleurs, ce travail a contribué à divers projets tels que :

- le projet Européen MEDEA A222, COMPORT (COMPonents for PORTable multimedia systems[1]) dont l'objectif était la réalisation de composants ou sous-systèmes intégrés pour les terminaux multimedia portables. Le but était donc de développer une architecture flexible basse consommation pour l'intégration de nouveaux services. Dans ce contexte, nous avons démontré l'intérêt de l'utilisation de processeurs asynchrones pour l'implantation d'applications multimedia dans des systèmes embarqués. Un processeur asynchrone, du fait qu'il n'utilise pas d'horloge, conduit à une implémentation très basse consommation et sans rayonnement électro-magnétique. La démonstration a donc été faite pour l'algorithme de détection de visage proposé dans cette thèse, qui a été implanté sur le processeur ASPRO (micro-processeur asynchrone 16 bits co-développé avec le groupe CIS, 'Concurrent Integrated Systems', du laboratoire TIMA[2], STMicroelectronics et France Télécom R&D).

- le projet RNRT TEMPO VALSE (Terminal Expérimental MPEG-4 POrtable pour la Visiophonie et l'Animation Labiale ScalablE[3]) dont l'objectif est de réaliser une maquette de terminal multimedia portable basé MPEG-4, dédié à la consultation et à la communication interactive multimodale (son + image). L'application démontrée est de qualité échelonnable ("scalable"), allant de la vidéo à l'animation de visage parlant. Ma contribution à ce projet se situe au niveau de la méthodologie de conception de systèmes embarqués appliquée au codage vidéo pour la visiophonie mobile.

1. MEDEA A.222 : http://medea.org/proj/comm_txt.htm
2. Processeur ASPRO : http://tima.imag.fr/cis/
3. TEMPO VALSE : http://www.telecom.gouv.fr/rnrt/res_d17_ap99.htm

- Le projet RNRT MILPAT (Méthodologie et développement pour les InteLlectuals Properties (IPs) pour Applications Télécom[1]) dont l'objectif était de formaliser la démarche de conception de composants virtuels matériels (IPs de niveau comportemental). Ces composants correspondent à une description générique, cohérente et interfaçable d'une famille d'opérateurs matériels permettant de réaliser une application donnée. A l'aide de ces composants virtuels et grâce aux outils de synthèse d'architectures du commerce, la conception de nouvelles applications est alors fortement accélérée. La méthodologie ainsi proposée a été testée sur une application de codage de type MPEG-4 pour les télécommunications, pour laquelle nous avons fourni la spécification d'un IP de calcul de distorsion générique et paramétrable.

De plus, cette thèse participe aux travaux de recherche du groupe CIS du laboratoire TIMA sur le développement de solutions innovantes et efficaces pour la conception de systèmes intégrés complexes (SoCs).

Plus précisément, ces travaux sur la méthodologie de conception de systèmes embarqués sont motivés par l'évolution technologique et la généralisation du multimedia. L'arrivée de la vidéo sur les terminaux mobiles ne relève plus désormais de la fiction et passera certainement dans un premier temps par l'intégration de la fonction de "streaming vidéo". En effet, ce dernier n'effectuant que le décodage de la vidéo, ne pose pas de contraintes majeures au regard de la puissance de traitement aujourd'hui disponible sur les processeurs multimedia embarqués et du débit descendant offert par les réseaux 3G. En revanche, l'application de visiophonie implique l'utilisation d'un codeur vidéo dont l'implantation sur un terminal mobile reste un challenge du point de vue des contraintes de consommation (autonomie du terminal), de qualité vidéo à très bas débit (acceptabilité du service), de réseau (résistance aux erreurs, QoS), …
Face à cette problématique, nous proposons une méthodologie basée sur l'adéquation algorithme–architecture.
Au niveau algorithmie, l'utilisation d'un codeur de type MPEG-4 semble la plus attrayante. En effet, ce codeur basé objet autorise le codage efficace des scènes vidéo en rehaussant la qualité sur les objets d'intérêt. De plus, il offre des outils de résistance aux erreurs adaptés au réseau à qualité de service non garantie. Toutefois, le profil pour les applications mobiles spécifié par la norme MPEG-4 n'exploite pas la notion d'objet et reste donc basé bloc comme ces prédécesseurs, H.263 et MPEG-2. Aussi, nous proposons un nouveau schéma de codage basé bloc, de type H.263, qui met en oeuvre la notion d'objet dans l'esprit de la norme MPEG-4.
Au niveau architectural, l'implémentation d'un tel système de codage peut revêtir plusieurs aspects, et soulève de nombreuses questions comme par exemple : l'avenir des processeurs multimedia embarqués est-il dans les DSPs, les ASICs ou les processeurs RISCs ? ou encore dans un système multi-processeurs de type SoC ?, et dans le cas des SoCs, quelle méthodologie de conception faut-il adopter ?
Aujourd'hui, les réalisations de systèmes de codage vidéo les plus abouties sont principalement basées sur des ASIPs comme le circuit IVT[2] de STMicroelectronics ou le circuit TC35273XB[3] de Toshiba, qui correspondent à des architectures multi-processeurs programmables mais spé-

1. MILPAT : http://www.telecom.gouv.fr/rnrt/projets/pmilpat.htm
2. M. Harrand et al., 'A single chip CIF 30 Hz, H.261, H.263 and H.263+ video encoder / decoder with embedded display controller', IEEE Journal of Solid State Circuits, Vol. 34, No. 11, pp 1627-1633, Novembre 1999.
3. T. Nishikawa et al., 'A 60 MHz, 240 mW MPEG-4 videophone LSI with 16 Mb embedded DRAM', Int. Solid State Circuits Conf. (ISSCC), San Fransisco, 2000.

cifiques à un domaine d'application. D'un autre côté, on voit apparaître des solutions plus flexibles de type DSP-MCU (DSP ayant des capacités de contrôle) ou RISC-DSP (RISC avec des unités de calcul de type DSP). On peut citer, par exemple, le coeur de DSP SC140[1] de Motorola-Lucent pour les DSP-MCU, et l'ARM920Ex[2] pour les RISC-DSP. Une autre tendance est celle des SoCs intégrant un DSP et un RISC comme l'architecture OMAP[3] (C5510 + ARM9TDMI) de Texas Instruments. Enfin, une approche moins "classique" également envisagée exploite les performances des circuits FPGA reconfigurables dynamiquement comme l'architecture Chameleon[4]. On voit qu'aujourd'hui le choix d'une architecture pour le traitement multimedia embarqué n'est pas trivial et qu'un compromis doit certainement être trouvé entre flexibilité, consommation, performance, coût, rapidité de conception ("time-to-market"),...

Dans le premier chapitre, nous introduisons le contexte de l'étude. La première partie aborde l'environnement particulier offert par la norme de télécommunication de 3ième génération UMTS (Universal Mobile Telecomunication System). Puis, nous présentons une description fonctionnelle du terminal mobile, ainsi que les premières réalisations commerciales de terminaux de visiophonie mobile. Nous discutons aussi de l'implémentation du codeur / décodeur vidéo sur les circuits dédiés, les circuits de traitement du signal et les systèmes intégrés (SoC : System on a Chip).
Dans la deuxième partie, nous détaillons les contraintes de réalisation du service de visiophonie (acceptabilité, faisabilité) sur les terminaux mobiles.
Enfin, la dernière partie traite des normes de codage vidéo standardisées par les organismes ISO/MPEG (International Standard Organization / Motion Picture Expert Group) et UIT-T (Union Internationale des Télécommunications), en mettant en évidence leurs points communs et leurs applications cibles.

Le deuxième chapitre traite de l'adéquation algorithme architecture pour les systèmes multimedia embarqués. L'étude de faisabilité du service de visiophonie mobile a nécessité la mise en place d'une méthodologie de conception au niveau système prenant en compte les contraintes de l'embarqué, en particulier, l'autonomie du terminal.
Tout d'abord, nous définissons le codeur vidéo retenu pour cette étude. Il s'agit d'un codeur basé bloc respectant la norme H.263 (recommandation UIT-T), et pour lequel nous envisageons différentes options de codage.
Puis, dans la deuxième partie, une étude au niveau algorithmique des différentes configurations du codeur vidéo est réalisée afin de fournir des éléments de complexité et de performance. Ces éléments sont ensuite discutés et nous fixons les configurations les mieux adaptées à la visiophonie mobile.
Enfin, la question de l'architecture du système de codage vidéo est introduite. Tout d'abord, nous présentons notre méthodologie qui s'articule autour de deux niveaux d'analyse : algorithmique (spécification système et analyse de la complexité) et architecturale (partitionnement matériel / logiciel et analyse de l'efficacité de l'implémentation). Puis, nous présentons un langage de spécification système (SDL : Specification and Description Language) utilisé pour décrire l'implémentation du codeur. Cette spécification est ensuite exploitée grâce à l'outil ArchiMate d'exploration d'architectures de la société ArexSys (Valiosys). Enfin, nous introduisons une nou-

1. StarCore SC140 : http://www.starcore-dsp.com
2. Arm9E : http:\\www.arm.com
3. J. Chaoui et al., 'OMAP : enabling multimedia applications in third generation (3G) wireless terminals', technical white paper, SWPA001, Décembre 2000, http://www.ti.com
4. Architecture Chameleon : http://www.chameleonsystems.com

velle fonction (intégrée à l'outil d'exploration) pour estimer la dissipation d'énergie d'un système embarqué basée sur une mesure de taux d'activité des bus de données.
Le troisième chapitre traite de l'amélioration de la visiophonie pour le réseau mobile. Pour répondre aux contraintes de la visiophonie embarquée, nous proposons un nouveau schéma de compression vidéo exploitant l'information sémantique de l'image vidéo, dans l'esprit de la norme MPEG-4 (codage objet). Ce schéma repose à la fois sur l'extraction du visage du locuteur, et sur une nouvelle stratégie de codage vidéo compatible avec la norme H.263.
Dans la première partie, nous commençons par un tour d'horizon des techniques de détection du visage proposées dans la littérature. Puis, nous décrivons notre approche de détection / localisation du visage. Cette dernière repose sur l'hypothèse que la couleur "peau" est localisée dans une région étroite de l’espace des chrominances. Nous proposons ainsi un algorithme adaptatif de détection / localisation du visage qui satisfait aux contraintes de robustesse et de faible complexité.
Dans la seconde partie, nous effectuons l'intégration de la détection du visage dans un schéma de codage vidéo normalisé où le contrôle du codage (contrôle de l'estimation du mouvement, et régulation du débit) utilise l'information de position et de forme du visage extrait.
La troisième partie montre les apports du codage orienté visage en termes de complexité, de qualité et d'ergonomie.
Enfin, la quatrième partie conclut sur l'implémentation matérielle de la détection du visage et du codeur orienté visage.

Chapitre A : Contexte de l’étude

Nous allons avant de présenter les travaux de recherche sur l'adéquation algorithme / architecture des traitements multimedia embarqués, faire une présentation du contexte de l'étude. La premier partie de ce chapitre traite des terminaux mobiles et de l'opportunité offerte par les nouveaux réseaux mobiles (GPRS, UMTS) pour les applications multimedia. Les réalisations matérielles et/ou logicielles existantes ou à venir sont également évoquées. Dans une deuxième partie, les différentes contraintes intrinsèques à toute application vidéo sur un terminal mobile sont posées. Enfin, dans une troisième partie, un tour d'horizon des codeurs vidéo normalisés est établi afin de définir les différentes possibilités de codage qui s'offrent aujourd'hui à nous.

I. Les terminaux mobiles

Le marché des terminaux mobiles est en train d'évoluer d'un environnement dominé par la voix à un environnement où tous les media sont susceptibles de co-exister. L'apparition des services Internet, d'échange de données, d'écoute de musique, de consultation de clip vidéo, ... ne fait désormais plus aucun doute [1]. Cette transition requière cependant une modification profonde des réseaux de transmission et nécessite de repenser l'ergonomie des terminaux.

I.1 Contexte UMTS

Avec l'accroissement du nombre d'utilisateurs d'Internet et des réseaux mobiles [2], ces deux environnements se devaient de converger et d'offrir des services basés contenu sur le réseau mobile. L'UMTS est le réseau supportant cette évolution [3].
Les principaux objectifs de l'UMTS sont :

- un débit de 144 kbit/s (forte / moyenne mobilité) à 2 Mbit/s (mobilité restreinte),
- un accès aux mêmes services à partir de différents types de terminaux (agenda électronique, ordinateur portable, téléphone mobile, ...),
- une meilleure efficacité spectrale par rapport aux anciennes technologies,
- une qualité de service au moins égale à celle des réseaux précédents, voir égale à celle des réseaux filaires,
- une flexibilité accrue pour l'introduction de nouveaux services,
- la flexibilité des ressources radio: réseaux multiples (compatibilité avec les réseaux de 2^{ième} génération) et types de trafic (voix, données, vidéo, ...).

L'UMTS offre donc la possibilité de construire de multiples applications autour de l'audio, de la vidéo, du texte, du graphisme, mais quelles seront réellement les applications supportées ?, et quelles ressources sont nécessaires à la mise en oeuvre de ces services ?
Voici quelques exemples de services pour les terminaux de 3^{ième} génération :

- services multimedia,
- commerce sur mobile,
- messagerie,
- voix sur IP,
- informations (journaux, TV, ...),
- localisation / orientation,
- professionnel (télé-travail, ...).

L'évolution des réseaux mobiles et des applications associées du GSM à l'UMTS et au-delà peut être vue selon la figure 1.

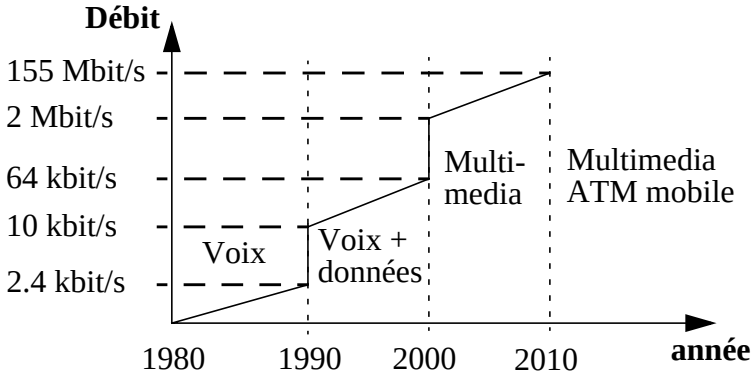
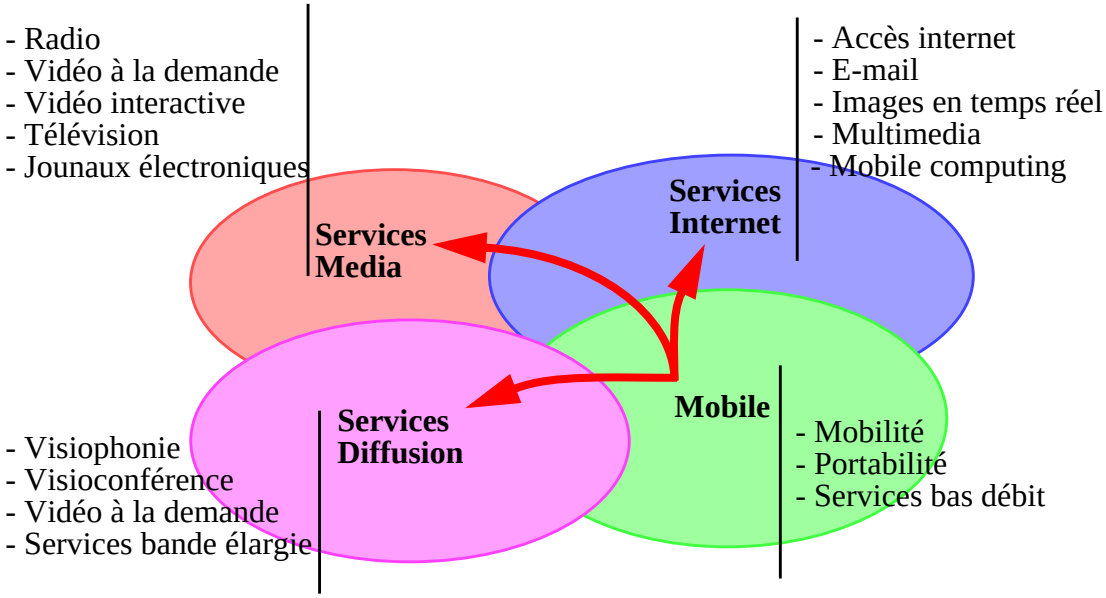


Figure 1: évolution des applications mobiles

La visiophonie fait bien entendu partie des futures applications sur réseau UMTS. Jusqu'à présent les tentatives de mise en oeuvre de ce service ont plutôt été infructueuses du fait du manque de bande passante, du prix élevé et du nombre peu élevé de terminaux équipés. Désormais, les débits disponibles à la fois sur les réseaux mobiles et les réseaux fixes permettent le déploiement d'un service de vidéo-communication accessible à tous.
L'UMTS permettra également, à terme, de faire converger le monde de la diffusion, celui d'Internet et celui du mobile (figure 2).



Source : European Broadcasting Union, Mars 2000

Figure 2: universalité de l'UMTS

Les objectifs de l'UMTS en terme de débits sont très prometteurs, ils sont présentés dans le tableau 1 en fonction de l'environnement et de la mobilité.

Tableau 1: débits proposés par l'UMTS

Environnement	Services temps réel		Autres	
	Débit max.	Taux erreur binaire / Délai	Débit max.	Taux erreur binaire / Délai
Rural (vitesse < 500 km/h)	144 kbits/s	10^{-3} à 10^{-7} / 20 à 300 ms	144 kbits/s	10^{-5} à 10^{-8} / 150 ms dans 95 % des cas
Urbain (vitesse < 120 km/h)	384 kbits/s		384 kbits/s	
Intérieur et extérieur courte portée vitesse < 10 km/h	2 Mbits/s		2 Mbits/s	

Avant l'arrivée de l'UMTS, un premier pas sera franchi avec le réseau GPRS [4] adapté à la transmission de données en mode paquet. Le GPRS est structuré en 3 classes selon les capacités du terminal à supporter simultanément ou non les services en mode commuté et en mode paquet.

Ces 3 classes de terminaux se décomposent chacune en 29 classes dites "multislot" définissant le nombre d'intervalles de temps ("Time Slot") utilisé dans les deux sens de transmission (voie montante, TX, et voie descendante, RX) par trame radio.
Les 3 classes de réseaux GPRS sont :
classe A : supporte simultanément les services commuté et paquet,
classe B : supporte séquentiellement les deux services,
classe C : supporte exclusivement l'un des deux services.

Les 29 classes de multislot sont définies dans le tableau 2.

Tableau 2: classes multislot des terminaux GPRS

Classe multislot	Slots RX	Slots TX	Max.(*)	émission et réception simultanées
1	1	1	2	non
2	2	1	3	non
3	2	2	3	non
4	3	1	4	non
5	2	2	4	non
6	3	2	4	non
7	3	3	4	non
8	4	1	5	non
9	3	2	5	non
10	4	2	5	non
11	4	3	5	non
12	4	4	5	non
13	3	3	8 ou 16	oui
14	4	4	8 ou 16	oui
15	5	5	8 ou 16	oui
16	6	6	8 ou 16	oui
17	7	7	8 ou 16	oui
18	8	8	8 ou 16	oui
19	6	2	8 ou 16	non
20	6	3	8 ou 16	non
21	6	4	8 ou 16	non
22	6	4	8 ou 16	non

Tableau 2: classes multislot des terminaux GPRS

Classe multislot	Slots RX	Slots TX	Max.(*)	émission et réception simultanées
23	6	6	8 ou 16	non
24	8	2	8 ou 16	non
25	8	3	8 ou 16	non
26	8	4	8 ou 16	non
27	8	4	8 ou 16	non
28	8	6	8 ou 16	non
29	8	8	8 ou 16	non

(*) Max. définit le nombre maximal de paquets RX ou TX dans une trame.

- Le débit brut offert par un slot est de 21.4 kbits/s (13.4 kbits/s utile avec un taux d'erreur binaire maximum de 10^{-3}). Les classes supportant l'émission et la réception simultanées correspondent au mode FDD ("Frequency Division Duplex), les autres classes utilisent le mode TDD (Time Division Duplex)

- Le taux d'erreur toléré pour la visiophonie est de 10^{-3} (spécification MPEG-4 vidéo pour les réseaux mobiles [5], le taux d'erreur toléré peut descendre jusqu'à 10^{-6} pour des applications telles que la diffusion à qualité TV). Pour construire une application de visiophonie les débits montant et descendant doivent être symétriques, et au minimum entre 30 et 40 kbit/s. Cette application temps réel suppose aussi une émission et une réception simultanées (mode FDD). Les classes GPRS correspondant à ces critères sont donc comprises entre la classe 13 et la classe 18. La classe 13 correspond à un débit utile de 40.2 kbits/s dans les deux sens pour l'audio et la vidéo.
Pour la visiophonie, on choisit une résolution minimale égale au QCIF (144*176 pixels) et une fréquence image minimale de 15 Hz ce qui correspond à qualité de service acceptable.
De plus, afin de ne pas dégrader la vidéo-communication, il faut que le retard du son vis à vis de l'image soit inférieur à 300 ms et le retard de l'image vis à vis du son soit inférieur à 150 ms [6]. Enfin, pour un bon niveau d'interactivité la latence maximale acceptable est de 100 ms.

I.2 Les terminaux visiophoniques

Avec les débits promis par les futurs réseaux mobiles, différents concepts de terminaux mobiles ainsi que des premières réalisations ont vues le jour ces dernières années.
Ces terminaux apparaissent sous deux formes :
- monolithiques comme leurs prédécesseurs (exemples : Kyocera [7], Orange [8]),
- éclatés, mettant en oeuvre la nouvelle technologie de communication radio haut débit, "bluetooth" [9] (exemple : NEC [10]).

Ils intègrent les mêmes principales fonctionnalités présentées dans le schéma fonctionnel d'un terminal mobile visiophonique de la figure 3.
Le tableau 3 donne un descriptif des premières réalisations proposées sur le marché.

Tableau 3: exemples de terminaux visiophoniques

Fabriquant	Caractéristiques
Kyocera VP-210 (1999)	Visiophone mobile, codage M-JPEG 32 kbit/s, 2 images/s autonomie : 1h écran : 2", caméra CMOS 110 kpxls
NEC WCDMA vidéophone (1999)	Visiophone mobile, codage MPEG-4, H.263 64 -> 128 kbits/s, 10 images/s écran : 2", caméra CCD 1/6" consommation : 0,3 W
Orange / Nokia / Microsoft (2000)	Visiophone mobile 28.8 kbits/s, 12 images/s écran : 4" autonomie : 1h30 ARM SA1110 200 MHz + 16 Mb ROM + 16 Mb RAM

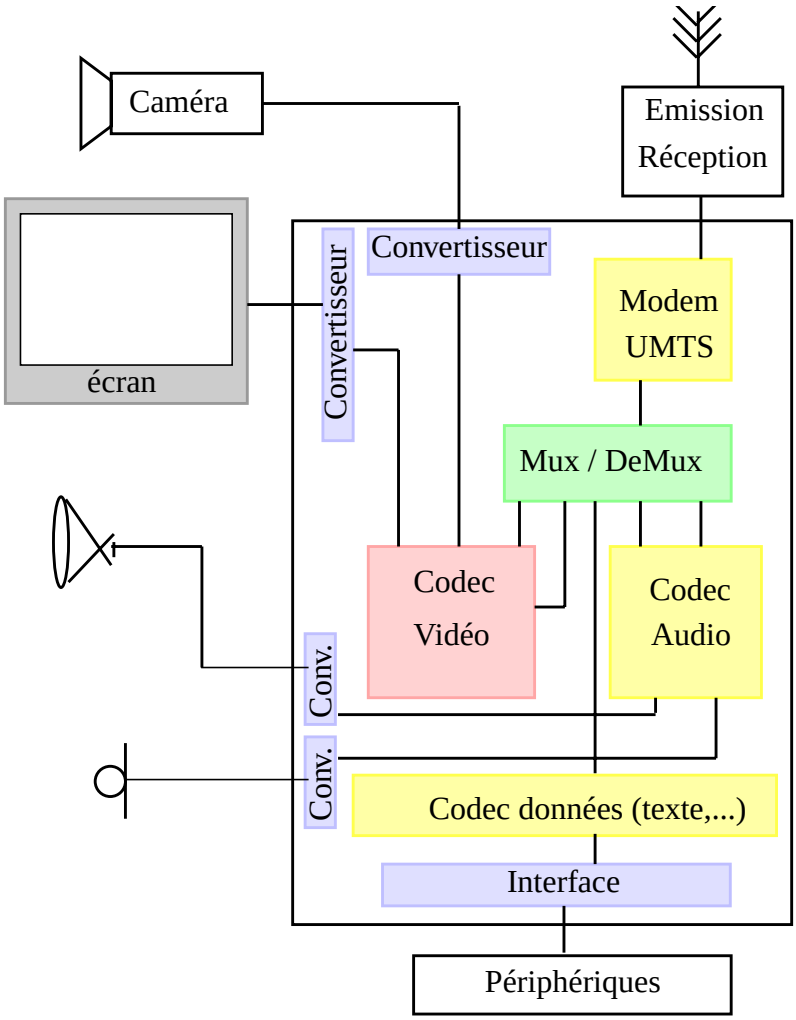


Figure 3: schéma fonctionnel d'un terminal mobile multimedia

I.3 Implantations matérielles et/ou logicielles

Aujourd'hui, différentes approches sont étudiées pour la réalisation des codeurs vidéo :

• implémentation sur du matérielle dédiée (ASIC : "Application Specific Integrated Circuit"): cette approche peut être retenue pour les applications orientées 'flot de données' fortement contraintes ou destinées au grand public. Un algorithme régulier est d'autant mieux adapté à une implémentation sur un ASIC qu'il autorise la mise en œuvre efficace d'architectures pipelines et parallèles. Cette approche conduit aux meilleures performances en terme de consommation, de vitesse et de surface. Cependant, le coût de développement d'un ASIC est très élevé car sa réalisation est spécifique à l'application et les étapes de description et d'optimisation sont longues à tous les niveaux de la conception (de l'algorithmique à l'implémentation physique). De plus, un ASIC est de fait figé et n'est donc pas réutilisable pour une autre application ce qui limite fortement son intérêt dans le domaine du multimedia embarqué où les normes de codage sont évolutives et conduisent fréquemment à des mises à jour. C'est pourquoi, les applications sont de moins en moins entièrement implémentées sous forme d'ASIC et utilisent souvent une architecture programmable à laquelle sont adjoints des blocs ASIC pour les traitements trop coûteux. Une alternative à l'ASIC est une architecture spécialisée non plus pour une application mais pour un domaine d'application particulier (comme par exemple le codage vidéo) : l'ASIP pour 'Application Specific Instruction set Processor'. Ces derniers processeurs ont une architecture mixte possédant des accélérateurs dédiés pour les traitements critiques et autorisant la programmation de nouveaux algorithmes ayant les mêmes contraintes de traitements. L'IVT de STMicroelectronics [11] est un exemple d'ASIP pour le codage vidéo.

• implémentation sur un FPGA ("Field Programmable Gate Array") :
la solution FPGA est intéressante à plus d'un titre. Elle associe les performances d'une architecture dédiée et la flexibilité de programmation. Toutefois, cette flexibilité conduit à une surface et une consommation bien supérieures à celles d'un ASIC. De plus, les cellules logiques travaillent au niveau bit ce qui n'est pas adapté à des applications du traitement du signal ou de l'image et, comme l'ASIC, le FPGA ne permet pas une implantation efficace de blocs orientés flot de contrôle. L'utilisation d'un réseau programmable de cellules plus complexes (FPFA : 'Field Programmable Function Array' [12][13]), comme l'architecture RCP de la société Chameleon [14], ou encore l'architecture IRAM de l'université de Berkeley [15], conduit à une architecture mieux adaptée au traitement du signal, et l'adjonction d'un cœur de microcontrôleur, comme dans l'architecture GARP [16], confère au circuit le contrôle nécessaire à l'implantation d'une application complexe.

• implantation sur un processeur programmable :

- DSP ('Digital Signal Processor') :
le DSP allie la flexibilité d'un processeur programmable et les performances attendues pour le traitement du signal temps réel (opérateurs arithmétiques spécialisés, bande passante importante, instructions spécifiques). L'apparition récente d'architecture VLIW (TMS320C6x) permet d'accroître fortement sa puissance de calcul au détriment de la consommation et de la facilité de programmation. Une analyse détaillée des différents types de DSP est fournie dans [17].

- microcontrôleur :
circuit le plus largement utilisé, il est destiné à des opérations de contrôle et de gestion de tâches. Bien que les dernières évolutions de microcontrôleur intègrent des unités arithmétiques plus complexes (ARM9Ex), leur performance reste limité pour le traitement du signal fortement contraint.

- processeur généraliste (GPP : 'General Purpose Processor') :
la puissance de calcul et la flexibilité sont les objectifs premiers du GPP, ils sont atteints grâce à une fréquence de fonctionnement élevée, des architectures matérielle et logicielle adaptés : superscalaire, fort pipeline, prédiction de branchement, caches multi-niveaux, instructions spécialisés (par exemple l'instruction multimedia : MMX pour le Pentium, AltiVec pour le PowerPC, VIS de Sun) et extension SIMD. Cependant, le GPP n'est pas bien adapté aux applications embarquées du traitement du signal et de l'image fortement contraint pour deux principales raisons : sa dissipation d'énergie est généralement élevée et surtout il n'est pas adapté au temps réel.

- réalisation d'un système sur puce (SoC: "System on a Chip") :
l'association de divers processeurs pour leur performances spécifiques (DSP, RISC, GPP) ne se fait plus sur une carte PCB, elle est désormais réalisée sur une seule puce : le SoC. Deux types de SoC sont à distinguer : ceux basés ASIC et ceux basés coeurs de processeurs (DSP, µC, GPP). La figure 4.a illustre le premier cas où certaines applications sont entièrement réalisées en matériel, un µC étant généralement adjoint pour la gestion des tâches, le support de l'OS,... Ces SoCs ciblent généralement une application et ne permettent pas de modifier profondément les algorithmes implantés. Dans le deuxième cas (figure 4.b), seules les blocs de traitements particulièrement coûteux sont réalisés en matériel, l'essentiel de l'application étant implanté en logiciel sur les coeurs de processeurs. Ceci conduit à une plus grande flexibilité sur les choix algorithmiques et permet une mise à jour moins contraignante des applications embarquées.

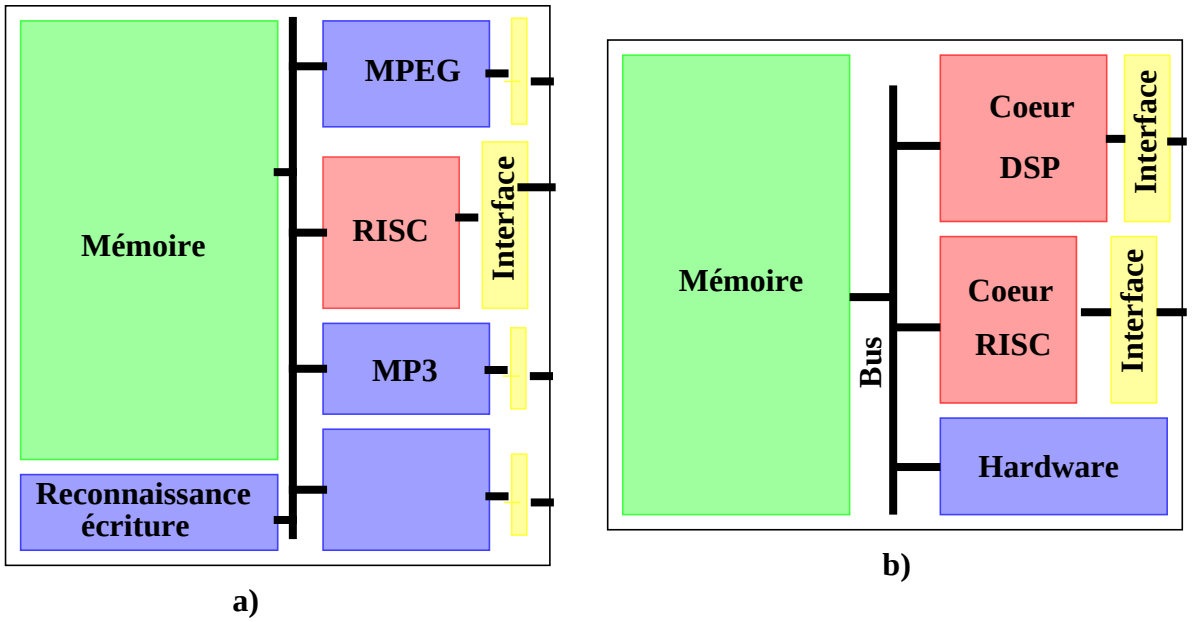


Figure 4: SoC a) basé ASIC, b) basé coeurs de processeurs

Par ailleurs, la conception de systèmes sur une puce, au vu de sa complexité, intègre la notion d'IP (Intellectual Property [41]). Un IP est un composant virtuel dont on peut distinguer deux types : le premier type correspond à un coeur de processeur (RISC, DSP, ...), le deuxième correspond à une fonction ou un opérateur spécifique. Il peut apparaître sous différentes formes :
- IP matériel (Hard core) : composant virtuel peu flexible entièrement caractérisé pour une technologie particulière.
- IP firme (Firm core) : composant virtuel optimisé décrit en HDL structurel à l'aide des composants élémentaires d'une bibliothèque générique. Plus flexible que l'IP matériel, il est portable

sur plusieurs technologies.
- IP logiciel (Soft core) : composant virtuel indépendant de la technologie décrit en HDL synthétisable (le plus souvent au niveau RTL).

Le tableau 4 présente différentes implantations de codeur et/ou de décodeur vidéo sur DSP. le tableau 5 donne, quant à lui quelques exemples de réalisations de SoCs plus ou moins spécifiques au codage vidéo.

Tableau 4: les DSP

Réalisations	Traitement vidéo	Spécifications
TI TMS320C54x [18]	décodage MPEG-4 : SQCIF @ 30 Hz (utilise 40 MHz du DSP)	1 MAC, 2 ALU Instruction sur 16 bits 30-532 MIPS consommation : 0,32 mW/MIPS
TI TMS320C55x [19]	codeur / décodeur MPEG-4 QCIF @ 30 Hz (utilise 200 MHz du DSP)	2 MAC, 4 ALU Instruction à longueur variable (8 à 32 bits) 288-600 MIPS consommation : 0,05 mW/MIPS
Motorola StarCore (SC140) [20]	JPEG2000 Image couleur 128*128 2 images/s	4 MAC 1200 MMACS (3000 MIPS RISC) @ 300 MHz 480 MMACS (1200 MIPS RISC) @ 120 MHz consommation : 0,2 mW/MMAC (1,5 V), 0,07 mW/MMAC (0,9 V)
Intel-Analog Devices MSA (Blackfin ADSP) [21]		2 MAC, 2 ALU Instruction sur 16 bits 600 MMACS @ 300 MHz consommation totale < 42 mW (0,9 V)

Tableau 5: les systèmes intégrés sur une puce (SoC)

Réalisations	Fonctionnalités	Spécifications
STMicroelectronics IVT [11]	codeur / décodeur H.263+ : CIF @ 30Hz	ASIP : architecture multiprocesseurs dédiés (cf. C.IV.2)
Toshiba TC35273XB [22]	Codec MPEG-4 SPL1 : QCIF @ 15 Hz Mux / Demux codec AMR	processeur RISC 16 bits + accélérateurs matériel 70 MHz 12 Mb de DRAM embarquée consommation 80 mW technologie 0,18 µm

Tableau 5: les systèmes intégrés sur une puce (SoC)

Réalisations	Fonctionnalités	Spécifications
Matsushita MN1959041 [23]	codeur / décodeur multi-canaux. codeur MPEG-4 SP : 2 canaux QCIF @ 15 Hz décodeur MPEG-4 SP : 1 CIF ou 4 QCIF @ 30Hz, ou MPEG-4 CP : 4 objets QCIF @ 15 Hz	technologie 0,18 µm consommation 50 mW (codec QCIF 15 Hz) 20 Mb de DRAM embarquée 54 Mhz
KAIST [24]	Décodeur MPEG-4 SP@L1 (QCIF@15Hz) graphique 3D : 2,2 Mpolygones/s	ARM9 + unité arithmétique MAC + matériel dédié (rendu 3D, compensation mouvement) technologie 0.18 µm consommation 160 mW @ 80 MHz
OMAP [25]	Codeur / décodeur MPEG-4 : QCIF @ 15 Hz (utilise 15% des capacités)	ARM9TDMI + C5510
Geo Emblaze A3 [26]	Décodeur MPEG-4 SP : QCIF @ 15 Hz <=> 35 MHz + 93 kB RAM + 35 kB ROM Audio+Mux+Video (décode) : <=> 80 MHz + 233 kB RAM + 130 kB ROM Voie+Mux+Video (codec) : <=> 157 MHz + 640 kB RAM + 328 kB ROM	ARM920 + matériel dédié (capture vidéo, estimation de mouvement, affichage vidéo) technologie 0.18 µm consommation 1.8 mW/MHz @2.5V@200Mhz

Outre les réalisations purement matérielles déjà commercialisées, on peut constater que la tendance est à la réalisation de systèmes intégrés sur une puce ou SoC. L'architecture des systèmes multimedia peut donc être vue comme un SoC construit autour d'un coeur de micro-contrôleur (pour la gestion du système et le support d'un éventuel OS), d'un DSP pour le traitement du signal (voix, texte, une partie de la vidéo,...), de blocs "matériels" dédiés pour les traitements fortement contraints et d'une quantité de mémoire embarquée importante (en général de type DRAM pour leur grande densité) afin de satisfaire la contrainte de faible consommation. Les circuits OMAP, Emblaze, VCPex, ..., sont les premières concrétisations de l'émergence des SoCs (programmables). Cependant, comme nous le verrons dans le chapitre sur les architectures, la mise en place d'une méthodologie de conception des SoCs n'est pas triviale et les outils de conceptions au niveau système ne sont pas toujours matures.

II. Contraintes de la visiophonie mobile

II.1 Qualité de service

Nous allons, dans ce paragraphe, lister les différents paramètres influant sur la qualité de service pour la visiophonie.

• Avant de pouvoir transmettre une image, il faut la capturer. Les premiers éléments à spécifier sont donc la caméra et les convertisseurs associés qui définissent différentes caractéristiques de la vidéo : la résolution, la fréquence, l'échantillonnage des pixels (luminance, chrominances).
La résolution minimale acceptable sur un terminal mobile est le QCIF (144*176 pixels) en 4:2:0 (luminance en pleine résolution et les chrominances en quart de résolution). Nous fixons également une fréquence image minimale égale à 15 Hz afin de conserver une bonne fluidité de la vidéo mais surtout afin de maintenir une bonne synchronisation du son et des lèvres du locuteur. En effet, un mouvement saccadé des lèvres conduit à une perte d'intelligibilité lors d'une vidéocommunication. L'échantillonnage de la luminance et des chrominances des pixels est fait sur 8 bits, ce qui offre une précision suffisante et fournit une dynamique compatible avec l'architecture des processeurs programmables (multiple d'un octet).

• Un deuxième élément jouant sur la qualité de service est l'écran. Les contraintes sont similaires à celles de la caméra (résolution, fréquence image, nombre de bits par pixel) avec en supplément la taille de l'écran et sa fréquence de rafraîchissement.
L'écran est un matériel critique du mobile du fait de sa forte consommation. De plus, si on veut intégrer de la vidéo, il doit être en couleur et d'une taille suffisante pour une bonne visibilité de la vidéo. Pour une application de visiophonie, il n'est pas acceptable de descendre en dessous d'une taille de 2". Le choix de la technologie adaptée est également un point important, dans le cas de la visiophonie mobile, le rendu doit être aussi visible en environnement intérieur qu'extérieur.

• Enfin, il faut tenir compte des contraintes induites par l'ensemble du système (saisie, codage, transmission, décodage, affichage) telles que le retard entre l'image et le son, la latence et les défauts du rendu visuel dus aux codages source et canal.
Afin de ne pas nuire à la conversation le retard et l'avance du son sur l'image ne doit pas excéder 300 ms et 150 ms, respectivement [6]. La latence quant à elle ne doit pas dépasser 100 ms si on souhaite maintenir un bon niveau d'interactivité.

II.2 Contraintes du réseau

La nature du réseau supportant la transmission de la vidéo influe sur les choix et techniques de codage et par conséquent sur la qualité de la vidéo. L'influence du réseau est liée à ses caractéristiques en débit, taux d'erreur et mode de transmission (mode circuit, mode paquet, protocoles,...).
Pour notre étude on se place dans le cas de la visiophonie sur un terminal mobile en situation de mobilité restreinte. Le réseaux UMTS devrait donc à terme nous offrir un débit pouvant aller jusqu'à 384 kbit/s, ce qui est un débit largement suffisant aujourd'hui pour des services de visiophonie. Cependant, les premiers déploiements des réseaux UMTS n'offriront certainement pas des débits aussi élevés. Pour qu'un service de visiophonie soit possible sur réseau mobile, il faut par conséquent palier à ce manque de débit par des techniques de codage performantes. De plus, la disponibilité de l'UMTS n'est pas encore clairement établie et son déploiement est précédé

par le réseau GPRS qui offre différentes configurations structurées en classes. Ces classes définissent entre autre les débits montant et descendant (paragraphe I.1). Pour qu'une vidéo-communication soit possible avec une qualité acceptable, nous estimons qu'un débit de 30 à 40 kbit/s est au moins nécessaire (son non compris). Il faut donc atteindre la classe 13 (débit utile de 40.2 kbit/s). Dans les études réalisées dans les parties B et C, nous avons retenu un débit cible de 40 kbit/s ce qui correspond à la classe 14 (débit utile : 53.6 kbit/s).

Nous choisissons un taux d'erreur de 10^{-3} pour l'application de visiophonie mobile. Cet environnement très "bruité" correspond au pire cas spécifié par la norme MPEG-4 dans ses profils visant les applications sur réseau mobile (le taux d'erreur typique retenu étant 10^{-4}). Cette contrainte nécessite la mise en oeuvre d'outils spécifiques pour la résistance aux erreurs. Cependant, le taux d'erreur dépend de l'application et si, par exemple, on envisage une application de diffusion TV un taux d'erreur de 10^{-6} doit être atteint sur le réseau.

Le dernier aspect d'un réseau à prendre en compte est le mode de transmission utilisé. Celui-ci conditionne le choix des outils de résistance aux erreurs et des techniques de régulation du débit comme nous le verrons par la suite.

L'ensemble des stratégies de codage mis en oeuvre ici répond donc à un compromis nécessaire entre la qualité de codage et le débit.

II.3 Contraintes du terminal

Une des premières contraintes qui s'impose dans le cas d'un terminal mobile est l'autonomie. Pour tout système embarqué la dissipation d'énergie est un problème majeur. Dans le cas d'un terminal UMTS, le codage canal est déjà très gourmand en puissance de calcul [27], les ressources restantes pour le codage source sont donc d'autant plus réduites. Des optimisations, tant au niveau algorithmique qu'au niveau architectural seront nécessaires afin de satisfaire à cette contrainte de faible complexité, faible consommation. Nous savons que tout traitement vidéo requiert des mémoires de grande capacité pour le stockage des images et une bande passante importante. En effet, nous verrons que les accès mémoires représentent plus du tiers des instructions d'un codage vidéo. Il semble donc évident qu'une attention toute particulière devra être donnée à l'architecture de la mémoire du système et au dimensionnement de cette mémoire [28]. D'autres sources non négligeables de dissipation d'énergie sont également présentes dans un terminal mobile. La plus coûteuse est l'écran qui représente aujourd'hui la plus grande part de la consommation d'un terminal [29]. On peut aussi citer l'émetteur / récepteur RF, les divers périphériques (micro, haut parleur, ...) [30].

II.4 Contraintes du codage

Les défauts induits par la compression peuvent être séparés en 6 catégories [31].

- effet mosquito

Il s'agit d'un défaut temporel. Il est essentiellement visible dans les régions de faible contraste par des fluctuations de la luminance / chrominance qui apparaissent au voisinage des contours de fort contraste et des objets en mouvement.

- rupture de ligne

Généralement, ce défaut apparaît sur de longues droites diagonales et se traduit par une discontinuité de la ligne au niveau de la frontière entre deux blocs (8*8 pixels) voisins. Cet effet est très peu visible. Il est du à la transformée en cosinus discrete.

- flou
Ce défaut est du à une variation rapide de luminance entre deux pixels voisins. Lors de la transformée en cosinus discrete les forts coefficients peuvent être perdus, ce qui induit un léger flou sur les contours des objets.

- bruit de quantification
Il apparaît lorsque se produit une saturation des coefficients quantifiés. Ce défaut peut être présent dans tout système faisant appel à la quantification.

- saccade
Directement lié à la technique de régulation du débit, ce défaut correspond à une chute de la fréquence image du fait d'un débit trop faible. Si la fréquence n'est ni suffisante pour avoir une vidéo fluide, ni suffisamment faible pour avoir de l'image par image alors le désagrément peut être considérable et apparaît sous forme d'un clignotement de l'image.

- effet bloc
Ce défaut est le plus remarquable sur les systèmes de codage par compensation de mouvement basée bloc. Les contours des blocs apparaissent plus marqués du fait d'une quantification trop rude des coefficients DC (réduction de la continuité entre blocs adjacents du coefficient DC) et/ou des coefficients AC (perte des hautes fréquences).

III. Les normes de codage vidéo

Deux organismes se partagent la normalisation des codeurs vidéo, l'UIT-T [32] qui s'intéresse principalement aux bas débits et aux applications de visiophonie et de visioconférence, et le groupe ISO/MPEG [33] qui dans un premier temps s'est intéressé à de plus hauts débits pour des applications d'archivage et de diffusion de la vidéo avec ses normes MPEG-1 et MPEG-2. Aujourd'hui, le groupe ISO/MPEG cherche avec la norme MPEG-4 à créer une norme "universelle" englobant les différents traitements multimedia. Les codeurs issus de ces deux groupes de normalisation reposent sur des principes de base communs, mais diffèrent de part leur configuration, choisie en fonction de l'application.

III.1 Principes

Les codeurs normalisés exploitent les redondances inhérentes aux vidéo naturelles, elles sont de deux types :
- spatiale, deux pixels voisins d'une image naturelle se ressemblent,
- et temporelle, deux images consécutives ne diffèrent que par les déplacements des objets les composants.

La dé-corrélation spatiale est effectuée intra-image par transformée dans le domaine fréquentiel (en général, il s'agit d'une transformation en cosinus discrète (DCT) sur une subdivision de l'image en blocs). La redondance temporelle est quant à elle réduite par estimation de mouvement entre l'image à coder et une image de référence. S'en suit une quantification et un codage entropique permettant la compression des données ainsi obtenues.

Trois modes de codage sont possibles et conduisent à distinguer trois types d'image (figure 5) :
- les images intra, dites de type I, codées sans référence à une autre image. Seule la redondance spatiale est réduite. Les coefficients de la transformée sont ensuite quantifiés et codés.
- les images inter prédites, dites de type P, codées en référence à une image précédemment reconstruite (estimation de mouvement, transformation en cosinus discrète de l'erreur de prédiction, quantification de cette erreur et codage entropique des valeurs quantifiées ainsi que des vecteurs de mouvement trouvés).
- les images inter prédites bidirectionnelles, dites de type B. Elles sont prédites en références à deux images reconstruites, une correspondant à une image précédente et l'autre à une image suivante.

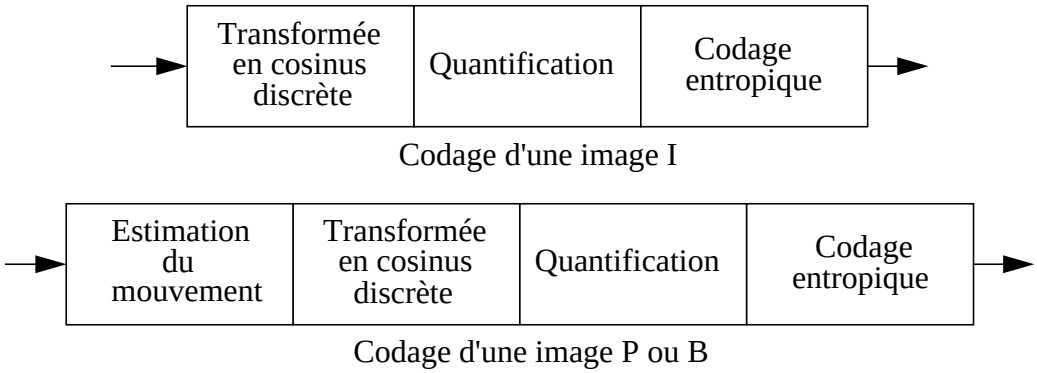


Figure 5: principes de codage des images I, P et B.

Les images de type I et P servent de références aux images P et B, les images de type B ne servent jamais de références.
Le codage d'une séquence vidéo consiste en une succession d'images intra et inter (figure 6) agencées de manière à satisfaire les contraintes de débit, de qualité, de complexité, ...

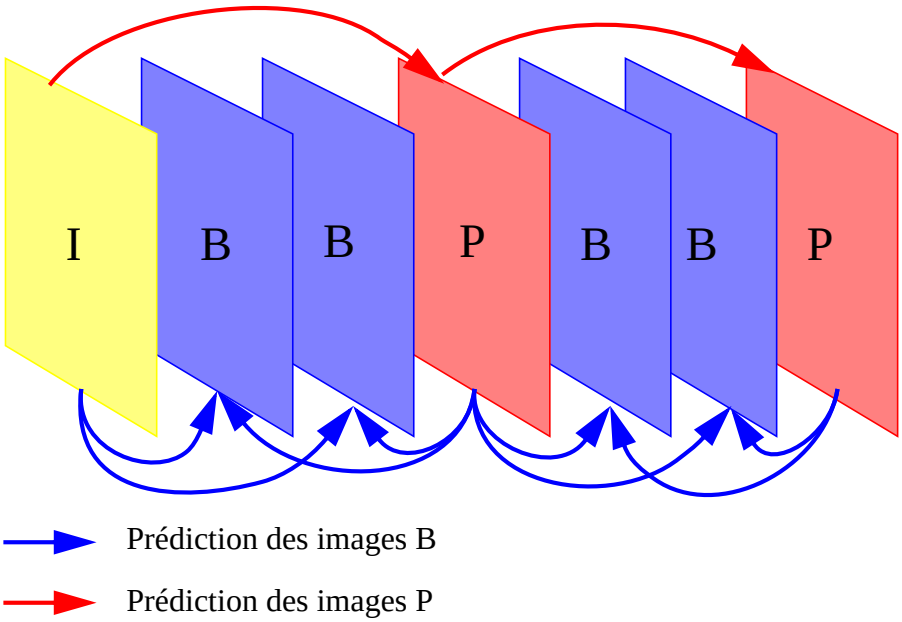
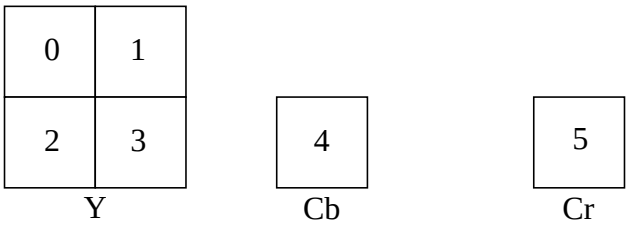
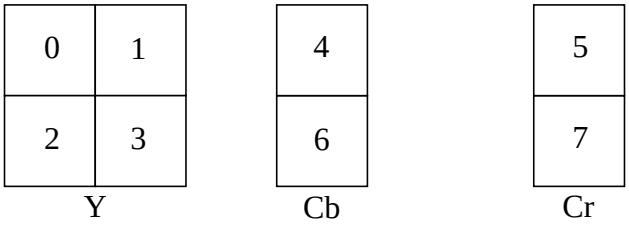


Figure 6: exemple de configuration de codage d'une séquence vidéo

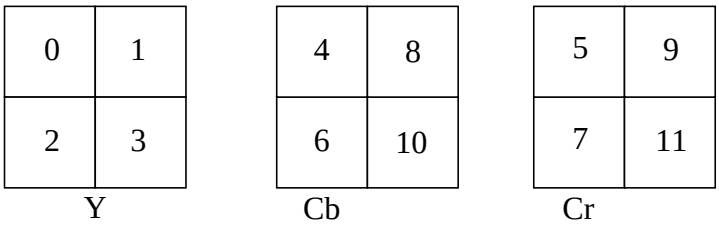
Les images I sont les plus coûteuses en débit mais servent de point d'entrée dans une séquence vidéo, leur fréquence dépend de l'application visée (accès aléatoire nécessaire ou non,...).
Les images B fournissent le meilleurs taux de compression mais au prix d'une augmentation de la complexité. De plus, il n'est pas toujours souhaitable d'utiliser des images B. En effet, l'insertion d'une image B induit un retard important au codage et au décodage du fait de l'utilisation d'une image de référence lui succédant et de la nécessité de ré-ordonner les images au décodeur, ce qui n'est pas toujours compatible avec les applications ayant des contraintes fortes de faible retard et de temps réel telles que la visiophonie. Les images B sont généralement utilisées pour des applications nécessitant un fort taux de compression comme l’archivage de séquence vidéo.
Les images P correspondent à un compromis entre les images I et les images B, et constituent en général le corps du codage en visiophonie.
Notons, de plus, que les codeurs travaillent sur des subdivisions de la luminance et des chrominances des images. Le découpage en luminance est un quadrillage régulier par des carrés de taille 16*16 pixels appelés macrobloc (MB), chacun de ces macroblocs est découpé en quatre blocs de taille 8*8 pixels. Le nombre de blocs de chrominances rouge et bleue associés à chaque macrobloc dépend du sous-échantillonnage choisi qui peut être de trois types: 4:2:0, 4:2:2 ou 4:4:4 (figure 7). Du fait du pavage de la luminance par des macroblocs les dimensions de l'image sont des multiples de 16 pixels.



- structure de macrobloc en 4:2:0



- structure de macrobloc en 4:2:2



- structure de macrobloc en 4:4:4

Figure 7: les différents types de MB

Le diagramme fonctionnel d'un codec vidéo selon les principes exposés précédemment est présenté figure 8.

DCT / IDCT: transformée / transformée inverse en cosinus discrète,
Q / IQ: quantification / quantification inverse,
EM / CM: estimation / compensation de mouvement,
VLC / VLD: codage / décodage à longueur variable.

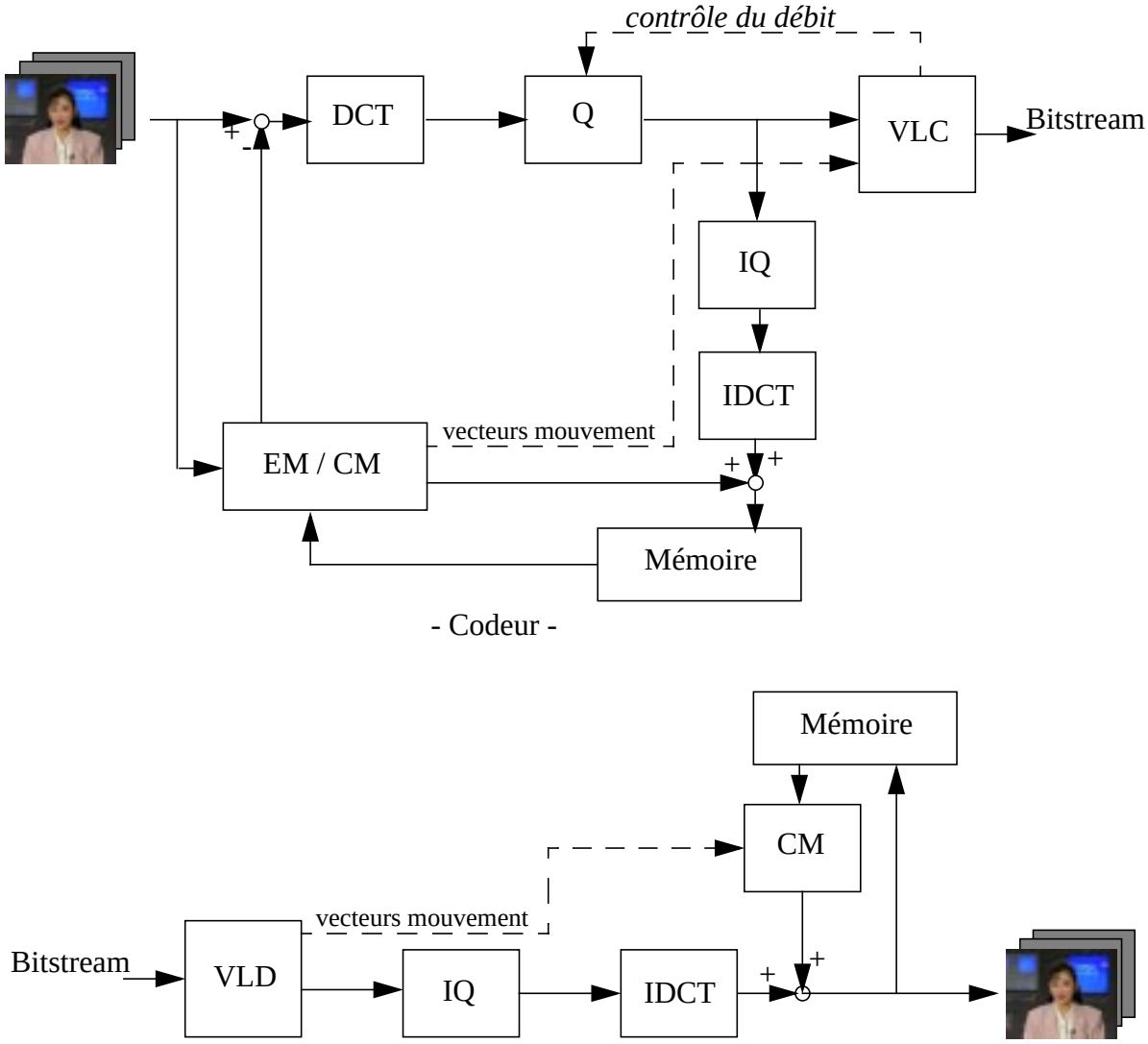


Figure 8: diagramme fonctionnel d'un codec vidéo

III.2 Normes de l'UIT-T

Le groupe de standardisation des télécommunications de l'Union Internationale des Télécommunications (UIT-T [32]) vis le jour avec la norme H.261 issue des travaux de recherche du CCITT dans le domaine de l'audiovisuel.

III.2.1 H.261

Première norme vidéo approuvée en 1993 (étude réalisée entre 1988 et 1993), H.261 [34] vise les applications de visiophonie pour le réseau RNIS à des débits de p*64 kbits/s (où p est compris entre 1 et 30).
Les formats d'image traités sont le QCIF (144*176 pixels) et le CIF (288*352 pixels). La fréquence image de base est 29.97 Hz mais peut être réduite en sautant 1, 2 ou 3 images entre chaque image transmise. Chaque MB subit une transformée en cosinus discrète de taille 8*8. Les coefficients ainsi obtenus sont alors quantifiés puis codés par un codage à longueur variable. La

prédiction se fait inter-image et elle peut être améliorer par une compensation de mouvement et un filtrage.

- La compensations de mouvement est optionnelle à l'encodeur, elle est spécifiée au niveau du décodeur comme suit :

chaque MB peut être affecté d'un vecteur de mouvement dont les dimensions horizontales et verticales ne peuvent excéder +/- 15 pixels. Le vecteur de mouvement estimé sur la luminance pointe un MB inclus dans l'image précédemment reconstruite. Les chrominances sont prédites par le même vecteur mouvement dont les coordonnées ont été divisées par 2 et arrondies à l'entier le plus proche. Cette prédiction correspond à la construction d'une image P.

- Le filtrage, optionnel, est un filtre spatial à deux dimensions non récursif travaillant au niveau pixel sur des blocs de taille 8*8 intervenant dans la boucle de codage.

H.261 spécifie également un rafraîchissement intra aléatoire au niveau MB toutes les 132 transmissions.

L'arrangement du flux vidéo est structuré en 4 niveaux:

- image,
- groupe de blocs (GOB),
- macrobloc (MB),
- bloc.

La couche image constitue l'entête d'une image dans le bitstream, elle est composée d'un code de début d'image, d'une référence temporelle et de diverses informations sur le mode de codage. Suivent les GOBs, constitués chacun d'un entête de GOB et de 33 macroblocs répartis comme indiqué figure 9. L'entête est structuré comme suit: un code de début, le numéro du GOB la valeur du pas de quantification et des indications sur le mode de codage. Chaque MB possède un code de début de MB et 6 blocs de codes à longueur variable correspondants aux valeurs quantifiées des coefficients de la transformée des 4 blocs de luminance et des 2 blocs de chrominance associés. L'entête du MB précise les choix de codage appliqués aux blocs à venir.

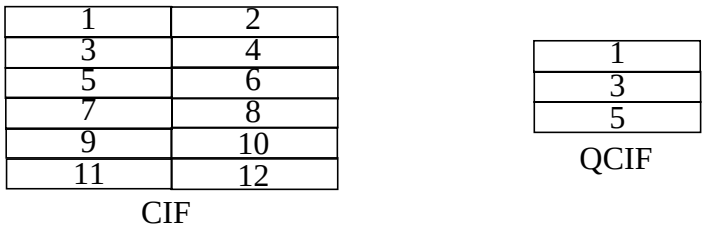


Figure 9: arrangement des GOBs dans une image

III.2.2 H.263

Il s'agit d'une norme de codage vidéo pour la vidéo-communication très bas débit dont la première version fut adoptée en 1995 [35]. H.263 vise les applications de visiophonie et de visioconférence sur RTC et RNIS. Cette norme repose sur les principes mis en place par la recommandation H.261.

Le codeur accepte cinq formats standardisés qui sont le QCIF et le SQCIF et optionnellement le CIF, le 4CIF (576*704 pixels) et le 16CIF (1152*1408 pixels). Les fréquences images restent inchangées par rapport à H.261.

Le décodeur doit avoir la possibilité d'effectuer des compensations en mouvement, le codeur pouvant exploiter ou non cette technique qui a évoluée depuis H.261. En effet, la précision des vecteurs de mouvement n'est plus en pixel entier mais en demi-pixel ce qui améliore grandement la qualité de la vidéo en réduisant fortement les effets dits "mosquito". L'utilisation d'image B est désormais possible.

Dans la syntaxe H.263, le nombre de macroblocs par GOB est de 1 ligne de macroblocs pour les résolutions SQCIF, QCIF et CIF (sauf en mode RRU: Reduced Resolution Update mode, cf. b.9, où les GOBs représentent 2 lignes de macroblocs), de 2 lignes de macroblocs pour la résolution 4CIF, et de 4 lignes de macroblocs pour la résolution 16CIF.
En plus de cette configuration de base, six options de codage ont été ajoutées afin d'augmenter les performances du codage et ses fonctionnalités.

La version 2 de la recommandation H.263 (1998), souvent appelée H.263+, met en oeuvre douze options supplémentaires et permet désormais de définir des formats et fréquences d'image personnalisés, ce qui étend encore la gamme de vidéo.

La dernière version de H.263 (2000), appelée H.263++, ajoute trois options et une spécification à la version antérieure. Outre l'amélioration en termes de qualité et de taux de compression, elle prend mieux en compte la transmission vidéo temps réel sur des réseaux à qualité de service non garantie (IP, pertes de paquets, ...).

Les différentes options de la recommandation H.263 et de ses extensions sont présentées brièvement ci-dessous.

a - Options de la version 1 (H.263)

a.1 - Mode multipoint en présence continue et multiplexage vidéo (CPMVM : Continuous Presence Multipoint and Video Multiplex), H.263 annexe C

Dans ce mode, jusqu'à quatre trains binaires de quatre vidéo indépendantes peuvent être envoyés via le même canal de transmission. Cette option est utilisée dans les applications multipoints ou quand le nombre de canaux disponibles est insuffisant.

a.2 - Vecteur mouvement non restreint (UMV : Unrestricted Motion Vector), H.263 annexe D

Ce procédé permet aux vecteurs de mouvement de pointer sur un MB extérieur à l'image de référence (D.1). Les pixels de bord d'image sont utilisés comme prédicteurs des pixels extérieurs à l'image. Ce mode induit des gains significatifs sur la qualité des contours de l'image si le mouvement est important en limites d'image (Ce gain est d'autant plus important que l'image est petite). De plus, ce mode permet d'augmenter la dynamique des vecteurs de mouvement (D.2), ce qui est utile dans le cas d'un déplacement de la caméra ou pour les images à grand format.

a.3 - Syntaxe basé sur un codage arithmétique (SAC: Syntax-based Arithmetic Coding), H.263 annexe E

Un codage arithmétique est utilisé en remplacement du codage à longueur variable et permet de réduire de façon significative le nombre de bits produits, à qualité constante.

a.4 - Mode de prédiction avancé (AP: Advanced Prediction), H.263 annexe F

Cette option met en oeuvre la "compensation de mouvement par bloc recouvrant" (OBMC: "Overlapped Block Motion Compensation") utilisée dans le codage de la luminance des images P conjointement à la prédiction par quatre vecteurs mouvement 8*8 sur certains MBs de l'image au lieu du vecteur mouvement 16*16 employé par défaut. Ce mode permet d'obtenir une amélioration de la qualité subjective de l'image par une réduction des effets blocs (elle donne des résultats équivalents au filtrage dans la boucle de codage prévue par la norme h263+). Lorsque ce mode est actif le mode UMV (unrestricted motion vector) dans son plus bas niveau est auto-

matiquement activé (D.1).

a.5 - Mode image PB, H.263 annexe G

Une image PB correspond au codage conjoint de deux images, une de type P et l'autre de type B. Une image PB est donc composée d'une image P prédite à partir d'une image P précédemment décodée et d'une image B prédite à partir de l'image P précédemment reconstruite et de l'image P actuellement codée. Grâce à cette option, la fréquence image peut être doublée au prix d'un faible surcoût en débit.

a.6 - Correction d'erreur par anticipation (FEC: Forward Error Correction), H.263 annexe H

Ce mode est utilisé lorsqu'aucune correction anticipée d'erreur n'est fournie par des moyens extérieurs tels que la couche multiplex et la couche système. La méthode de codage BCH est celle spécifiée dans la recommandation H.261.

b - Options ajoutées par la version 2 (H.263+)

b.1 - Codage Intra avancé (AIC: Advanced Intra Coding), H.263+ annexe I

Dans ce mode, les coefficients DCT des blocs Intra sont prédits des blocs voisins. Une table de codes à longueur variable spécifique aux blocs Intra est définie. Cette technique est appliquée à tous les blocs Intra, qu'ils appartiennent à une image Intra ou Inter. Cette option augmente fortement le taux de compression des blocs Intra.

b.2 - Filtrage de réduction des effets blocs (DF: Deblocking Filter), H.263+ annexe J

Dans ce cas, un filtrage est appliqué aux contours des blocs des images I et P afin de réduire les effets de blocs. Ce filtre affecte l'image reconstruite et influence les estimations du mouvement, il est situé dans la boucle de prédiction du mouvement.

b.3 - Structure en "slice" (SS: Slice Structured mode), H.263+ annexe K

La structure en "slice" est une extension de la structure en GOB du train binaire dans laquelle le découpage est flexible. Les objectifs de cette option sont de fournir une meilleure résistance aux erreurs, de rendre le train binaire plus adapté à un mode de transmission par paquets, et de minimiser les délais vidéo. Un "slice" est similaire à un GOB dans le sens où il apparaît dans la syntaxe intercalé entre la couche image et la couche MB. Cependant, son utilisation permet un découpage flexible de l'image en opposition avec le découpage pré-établi et l'ordre de transmission figé de la structure en GOB.

b.4 - Information complémentaire (SEI: Supplemental Enhancement Information), H.263+ annexe L

Une information complémentaire peut être adjointe au train binaire afin d'offrir de nouvelles fonctionnalités d'affichage des images ou des informations utiles pour la manipulation de la vidéo. Cette information peut être utilisée pour signaler qu'une image doit être totalement ou partiellement figée, pour des requêtes de blocage / déblocage de tout ou partie d'image avec ou non re-dimensionnement, pour marquer certaines images d'une séquence en vue d'une manipulation externe,... Cette information peut être présente même si le décodeur ne possède pas les fonctionnalités nécessaire à son utilisation, dans ce cas elle est tout simplement ignorée.

b.5 - Mode image PB amélioré (Improved PB-frames), H.263+ annexe M

Ce mode raffine le mode de codage en image PB déjà présent dans H.263 Version 1. Dans ce cas, les blocs B de l'image PB peuvent être prédits de l'image précédente par un vecteur mouvement indépendant ou prédit du bloc correspondant de l'image suivante, ce qui permet d'augmenter l'efficacité de codage dans le cas où les vecteurs P ne sont pas de bons candidats pour la prédictions des blocs B. La prédiction par rapport à l'image suivante seule est très efficace lorsqu'il y a un changement de scène entre une image P et une image PB.

b.6 - Sélection de l'image de référence (RPS: Reference Picture Selection), H.263+ annexe N

Cette option permet d'augmenter les performances pour la vidéo-communication en temps réel sur un canal fortement bruité. L'image de référence servant à la prédiction n'est pas obligatoirement la dernière image P (ou I) reconstruite. Ce mode est utilisé avec des messages de retour qui servent à synchroniser le codeur avec le décodeur (le codeur peut modifier son schéma de codage pour ne plus faire référence aux images perdues).

b.7 - Mode de décomposition graduelle temporelle, spatiale et de SNR, H.263+ annexe O

La décomposition graduelle, souvent appelée "scalabilité" en référence à l'anglais "scalability", correspond à la division du train binaire en plusieurs niveaux : un niveau de base et une ou plusieurs couches d'amélioration. Le flux de base est décodable indépendamment des couches supérieures qui, elles, sont décodées conjointement à la couche de base. L'amélioration affecte soit la fréquence image, soit la qualité d'image, soit la résolution de l'image, soit une combinaison des trois. La scalabilité temporelle est réalisée par l'ajout d'images B et conduit à une meilleure qualité perçue, due à une augmentation de la fréquence image de la vidéo ainsi décodée. La scalabilité spatiale améliore la qualité de l'image en jouant sur la résolution horizontale ou verticale ou les deux. La scalabilité au niveau SNR améliore la qualité de l'image sans augmenter sa résolution mais en transmettant plusieurs niveaux d'erreur de prédiction. Ce mode peut être très utile dans le cas de réseaux hétérogènes avec des capacités de bande passante différentes. Il procure également une aide aux techniques de correction d'erreurs.

b.8 - Ré-échantillonnage de l'image de référence (RPR: Reference Picture Resampling), H.263+ annexe P

Cette option induit un mode syntaxique particulier offrant la possibilité de ré-échantillonner l'image de référence avant son utilisation pour la prédiction. Ceci permet de choisir dynamiquement une résolution appropriée à la prédiction. Elle autorise également des déformations de l'image de référence utiles pour une estimation de mouvement globale ou pour la génération d'effets spéciaux.

b.9 - Codage en résolution réduite (RRU: Reduced-Resolution Update mode), H.263+ annexe Q

Ce mode est utilisé afin d'améliorer la qualité subjective des scènes vidéo contenant un fort mouvement, en augmentant la fréquence image. Le codeur a la possibilité d'intercaler des images de résolution réduite, le décodeur effectuant l'interpolation nécessaire à la reconstruction de l'image à son format original.

b.10 - Décodage par segments indépendants (ISD: Idependent Segment Decoding), H.263+ annexe R

Ce mode donne la possibilité de construire un flux binaire sans dépendance entre les données des différents GOBs ou slice constituant les images. Cette technique fournit une bonne résistance aux erreurs en évitant la propagation des erreurs inter segments (GOBs ou slice).

b.11 - Codage à longueur variable Inter alterné (AIVLC: Alternative Inter VLC), H.263+ annexe S

Ce mode est utilisé pour l'amélioration du codage des images Inter lorsqu'elles subissent des changements significatifs en luminance ou chrominance. Cette option offre la possibilité de compresser certains MBs Inter en utilisant les tables habituellement réservées aux MBs Intra.

b.12 - Mode de quantification modifiée (MQ: Modified Quantization), H.263+ annexe T

Cette option offre plus de possibilités pour la régulation du débit. Elle permet également de réduire les erreurs de quantification sur la chrominance, d'étendre la dynamique des coefficients de la DCT tout en mettant des restrictions sur leur valeur.
Ce mode autorise une dynamique accrue du paramètre donnant le différentiel du pas de quantification inter macroblocs. Il utilise également un pas de quantification réduit pour les chrominances par rapport à la luminance. Le nombre de niveaux représentables des coefficients de la transformée est ainsi augmenté pour permettre d'exploiter au mieux la précision fournie par le pas de quantification. Certaines restrictions sont enfin ajoutées aux coefficients afin d'améliorer les performances de détection des erreurs et de minimiser la complexité au décodage.

c - Options ajoutées par la version 3

c.1 - Sélection d'image de référence améliorée (ERPS: Enhanced Reference Picture Selection), H.263++ annexe U

Ce mode reprend les principes du mode RPS ("reference picture selection") pour une meilleure résistance à la perte de paquets, il y ajoute la possibilité de choisir les images de références pour les images B, et permet d'associer chaque vecteur de mouvement à une image de référence qui peut être différente suivant les MBs, et d'utiliser un canal de retour, et il met en oeuvre des techniques efficaces de gestion de la mémoire des images de références.

c.2 - Slice partionné en groupe de données (DPS: Data Partitioned Slice), H.263++ annexe V

Cette option partitionne un "slice" en groupes indépendants, séparant ainsi, deux à deux, les entêtes, les vecteurs de mouvement et les coefficients de la transformée. De plus, la protection des vecteurs mouvement est renforcée par une représentation en codes réversibles. Ces techniques améliorent la résistance aux erreurs et plus particulièrement aux corruptions locales du train binaire lors de la transmission. Cette stratégie est calquée sur la norme MPEG-4 "Simple Profile".

c.3 - Information complémentaire additionnelle (ASEI: Additional Supplemental Enhancement Information), H.263++ annexe W

Cette information ajoutée au train binaire inclut des indications sur l'utilisation d'une transformation inverse en cosinus discrète en virgule fixe, et aussi des messages ayant trait à l'image: données binaires arbitraires, texte (propriété intellectuelle, description de la vidéo,...), répétition des entêtes d'image (image courante, précédente, suivante avec référence temporelle,...), indications pour la vidéo entrelacée, identificateur d'images de référence,...

c.4 - Définition des profils et niveaux, annexe X

Au vu de la diversification des options de codage et de leur nombre, l'UIT-T a dernièrement décidé d'adopter la structure en profils et niveaux déjà adoptée par le groupe MPEG pour la spécification des codeurs. A savoir, un codeur H.263, selon cette annexe, peut désormais être défini par un profil et un niveau. La norme définit 9 profils et 7 niveaux :
0 - Le profil de base,
1 - le profil d'efficacité de codage H.320 avec une compatibilité ascendante, version 2,
2 - le profil de compatibilité ascendante, version 1,
3 - le profil interactivité et streaming sur mobile, version 2,
4 - le profil interactivité et streaming sur mobile, version 3,
5 - le profil conversationnel à haut taux de compression,
6 - le profil conversationnel sur internet,
7 - le profil conversationnel pour format entrelacé,
8 - le profil à forte latence.

Le tableau 6 fait correspondre les options aux profils.

Tableau 6: profils H.263

Annexe	Profil								
	0	1	2	3	4	5	6	7	8
Format image personnalisé	L	L	L	L	L	L	L	L	L
Fréquence image personnalisée	L	L	L	L	L	L	L	L	L
C. Mux video et présence multipoint continue									
D.1 Vecteur mouvement au delà des bordures de l'image			X	X	X	X	X	X	X
D.2 Extension de la dynamique des vecteurs mouvement						X	X	X	X
D.2 Vecteurs mouvement illimités									
E. Codage arithmétique									
F.2 Quatre vecteurs mouvement			X			X	X	X	X
F.3 Compensation en mouvement par bloc recouvrant			X			X	X	X	X
G. Image PB									
H. Correction d'erreur avant									
I. Codage intra avancé		X		X	X	X	X	X	X
J. Filtre anti-bloc		X		X	X	X	X	X	X

Tableau 6: profils H.263

Annexe	Profil								
	0	1	2	3	4	5	6	7	8
K. Codage structuré en slice, sans les sous-modes.				X	X		X		X
K. Avec l'ordonnancement aléatoire							X		X
K. Avec slices rectangulaires									
L.4 Information supplémentaire :gèle de l'image		X				X	X	X	X
L. Autres informations supplémentaires									
M. Image PB amélioré									
N. Sélection de l'image de référence									
O.1.1 Scalabilité temporelle (image B)									X
O. Scalabilité temporelle, spaciale et de SNR.									
P.5 Ré-échantillonnage de l'image de référence (facteur 4)									X
P. Ré-échantillonnage de l'image de référence									
Q. Mise à jour en résolution réduite									
R. Décodage segment indépendant									
S. VLC inter alternatif									
T. Quantification modifiée		X		X	X	X	X	X	X
U. Sélection d'une image de référence améliorée						X	X	X	X
U. Sélection d'une image de référence améliorée (mémoire)									
U. Sélection d'une image de référence améliorée (image B)									
V. Partitionnement des données en slices					X				

Tableau 6: profils H.263

Annexe	Profil								
	0	1	2	3	4	5	6	7	8
W. Information supplémentaire, répétition de l'entête d'image					X				
W. Information supplémentaire, indications pour l'entrelacé								X	
W. Information supplémentaire, autre									

L : dépend du niveau (tableau 7),
X : option retenue pour le profil.

Tableau 7: niveaux

	Niveaux						
	01	02	03	04	05	06	07
Format image max.	QCIF	CIF	CIF	CIF	CIF (format personnalisé)	720*288 (format personnalisé)	720*576 (format personnalisé)
Fréquence image max.	15/1,001 Hz	15/1,001 Hz (CIF) 30/1,001 Hz (QCIF)	30/1,001 Hz	30/1,001 Hz	50 Hz 60/1,001 Hz (352*240) Fréquence personnalisée	50 Hz 60/1,001 Hz (720*240) Fréquence personnalisée	50 Hz 60/1,001 Hz (720*480) Fréquence personnalisée
Débit max. en kbits/s	64	128	384	2048	4096	8192	16384

Les profils 3 et 4 sont les profils définis pour les applications mobiles. Ils mettent en oeuvre des outils de codage efficaces et de faible complexité ainsi que des outils de résistance aux erreurs essentiels pour une transmission de vidéo sur le réseau mobile. Le profil 4 est le plus intéressant pour les applications de visiophonie mobile du fait de l'implantation de l'annexe V ("data partitioned slice mode") augmentant la résistance aux erreurs.

III.3 Normes MPEG

III.3.1 MPEG-1

MPEG-1 [36] est devenu un standard international de l'ISO/IEC en 1992, dédié principalement à l'archivage sur CD-ROM avec une qualité VHS, et pour un débit d'environ 1,5 Mbits/s.
MPEG-1 possède une certaine flexibilité et l'utilisateur est à même de choisir les outils nécessaires à la construction de son application. Cependant, le groupe MPEG recommande les paramètres minimaux suivant afin d'assurer une bonne compatibilité des décodeurs. Le décodeur doit être capable de décoder de la vidéo à 30 Hz, avec une résolution de 720*576 pixels, et à un débit pouvant aller jusqu'à 1.86 Mbits/s.
MPEG-1 a été développée sur les bases des codages JPEG et H.261 afin que les principes restent communs et qu'une réalisation supportant plusieurs normes de codage soit possible. Cependant, MPEG-1 ayant été construite pour les applications multimedia sur CD-ROM, elle requière davantage de fonctionnalités, les plus importantes étant l'accès aléatoire à la vidéo, l'avance et le retour rapides, la lecture en sens inverse et l'éditabilité de la vidéo compressée.
MPEG-1 supporte les trois types d'image: I, P et B, sont donc supportés. Les images I, codées indépendamment des autres apportent les fonctionnalités d'accès aléatoire et d'avance et retour rapides, les images P et B permettent quant à elles d'avoir un taux de compression important.
Un codage uniquement à base d'images I fournit donc le meilleur degré de fonctionnalités (au sens accès aléatoire, avance / retour rapide, éditabilité) mais avec un faible taux de compression.
L'intercalage régulier d'image I dans un flux d'images P (figure 10) permet d'obtenir un bon taux de compression tout en gardant un degré d'accès aléatoire et d'avance / retour rapide acceptable.
L'utilisation d'images B permet d'atteindre un fort taux de compression tout en maintenant un degré de fonctionnalité suffisant mais au prix d'un délai de codage et de décodage accru. Un exemple de séquencement est présenté figure 10.

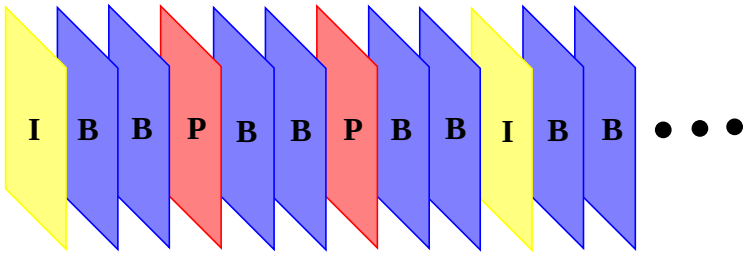


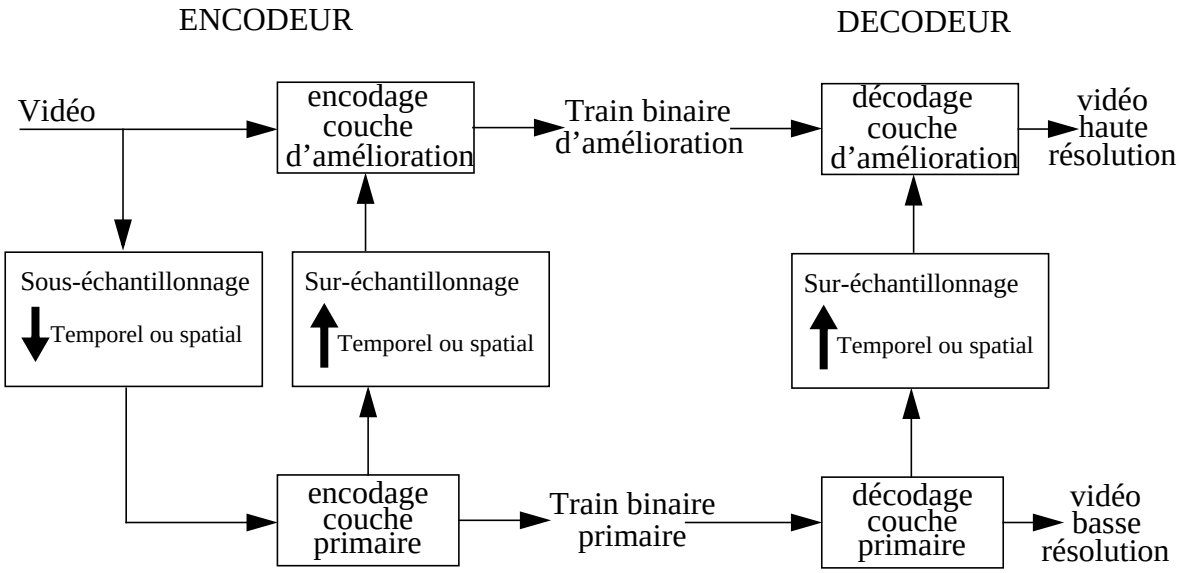
Figure 10: exemple de séquencement d'images I, P et B

Deux fonctionnalités supplémentaires sont à ajouter. La première a pour but de réduire le nombre de bits nécessaires au codage du coefficient DC (intensité moyenne du bloc) de la transformée par une prédiction inter-bloc de ces coefficients. La deuxième correspond à la possibilité d'ajuster le débit aux besoins de l'application en jouant sur le pas de quantification au niveau de chaque macrobloc ce qui permet de mettre en oeuvre des stratégies d'allocation du débit mieux adaptées aux contraintes des applications.
Pour la norme MPEG-1, huit fréquences d'image ont été retenues: 23.976, 24, 25, 29.97, 30, 50, 59.94 et 60 Hz, ainsi que tous les formats multiples de 16*16 pixels, pour un ratio de sous-échantillonnage Y:U:V égal à 4:2:0.

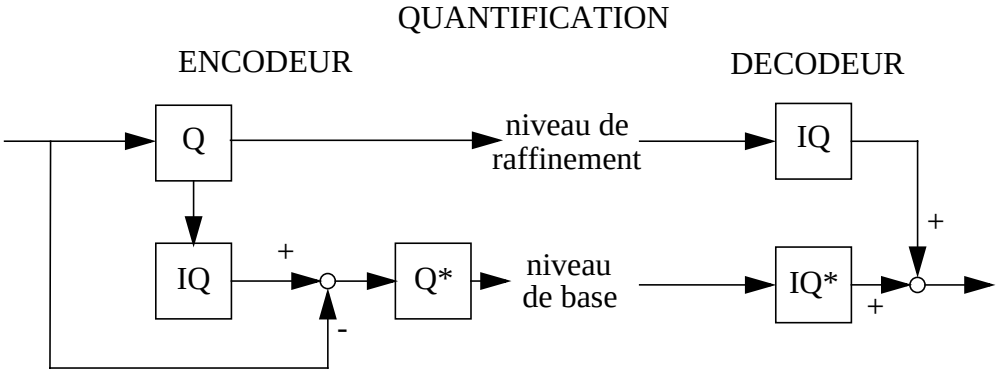
III.3.2 MPEG-2

Les études sur cette norme commencées en 1991 [37] dans le but de définir un standard pour la compression d'images TV allant de la qualité NTSC/PAL à la qualité CCIR Rec.601 pour des

débits inférieurs à 15 Mbits/s (MP@ML), furent finalisées en 1994. En 1992, les applications ciblées par la norme ont été étendues à la TVHD rendant obsolète la norme MPEG-3 vidéo initialement prévue pour ces applications. Les travaux de normalisation de MPEG-2 menés en collaboration avec le groupe d'experts de l'UIT-T SG-15 pour le codage vidéo ATM visent donc des applications aussi variées que la distribution de la TV sur le câble, les services de bases de données via ATM, les applications VTR numériques et la diffusion vidéo numérique terrestre et satellite.
MPEG-2 peut être vu comme un sur-ensemble de MPEG-1. Les fonctionnalités et la qualité du codage sont améliorées par l'ajout de modes de prédiction permettant de supporter de la vidéo entrelacée, de la scalabilité,...
MPEG-2 offre aussi une large gamme de fréquences, et reprend les huit possibilités de base offertes par MPEG-1, où les fréquences sont pondèrées par un coefficient dont le numérateur peut prendre une valeur entière comprise entre 1 et 4, et le dénominateur une valeur entre 1 et 32. Tous les formats d'image multiples de 16*16 pixels sont également supportés.
Afin de réduire la complexité des applications ne nécessitant pas toute la généricité du standard, MPEG-2 introduit une structure en "profil" pour les fonctionnalités et "niveaux" pour les résolutions. Les différentes configurations sont présentées dans les tableaux 8 et 9.
Outre les nouveaux modes de prédictions permettant de supporter de la vidéo entrelacée ou non (prédictions inter-champs et inter-trames sur des macroblocs de taille 16*8 pixels ou prédictions inter-images sur des macroblocs de taille 16*16 pixels), MPEG-2 spécifie un format de sous-échantillonnage Y:U:V supplémentaire (4:2:2) pour la production de vidéo à la qualité studio.
Les outils de décomposition graduelle (scalabilité) introduit par MPEG-2 profil "main" permettent l'inter-opérabilité entre différents services et permettent à des décodeurs de complexité et/ou de résolutions différentes de pouvoir décoder le même train binaire.
Le support de résolutions multiples (scalabilités spatiale et temporelle) est particulièrement intéressant pour le codage conjoint de vidéo de qualité HDTV et SDTV et la consultation rapide d'archives vidéo. La décomposition graduelle en qualité (scalabilité SNR) est, quant à elle, intéressante pour transmettre un flux de base conjointement à une couche d’amélioration et ainsi partitionner le train binaire en différents flux indépendants permettant de mettre oeuvre des systèmes plus résistants aux erreurs de transmission. MPEG-2 supporte jusqu'à trois niveaux différents de scalabilité. Les trois types de scalabilité sont illustrés figure 11 par des exemples de réalisations.



a) exemple de scalabilité spatiale ou temporelle



b) exemple de scalabilité SNR

Figure 11: les différentes scalabilités

Tableau 8: profils définis par MPEG-2

Profil	Algorithmes
Simple	fonctionnalités de MPEG-1 représentation YUV en 4:2:0 + codage vidéo entrelacée + accès aléatoire
Main	fonctionnalités du profil simple + images B (scalabilité temporelle)
SNR scalable	fonctionnalités du profil main + codage en décomposition graduelle de qualité (2 niveaux permis)

Tableau 8: profils définis par MPEG-2

Profil	Algorithmes
Spatial scalable	fonctionnalités du profil SNR scalable + codage en décomposition graduelle spatiale (2 niveaux permis)
High	fonctionnalités du profil spatial scalable 3 niveaux de scalabilité permis + représentation YUV en 4:2:2

Tableau 9: niveaux définis par MPEG-2

Level	Paramètres
Low	résolution max: 352*288 pixels à 30 Hz (CIF) débit max: 4 Mbits/s
Main	résolution max: 720*576 pixels à 30 Hz (4CIF) débit max: 15 Mbits/s
High 1440	résolution max: 1440*1152 pixels à 60 Hz (16CIF) débit max: 60 Mbits/s
High	résolution max: 1920*1152 pixels à 60 Hz (HDTV) débit max: 80 Mbits/s

Le tableau 10 fait le résumé des spécifications des normes MPEG-1 et MPEG-2.

Tableau 10: MPEG-1 vs MPEG-2

	MPEG-1	MPEG-2
Standardisation	1992	1994
Application cible	Archivage vidéo sur CD-ROM	Télévision numérique et HDTV
Résolutions spatiales	typiquement CIF	TV -> HDTV
Résolution fréquencielle	25 - 30 images/s	50 - 60 trames/s, 100 -120 trames/s
Débits	typiquement 1,5 Mbit/s	de 4 Mbit/s à 80 Mbit/s
Qualité	comparable au VHS	comparable à de la TV PAL/NTSC (pour le profil main)
Taux de compression	20 - 30	30 - 40

III.3.3 MPEG-4

Les standards précédents, MPEG-1 et MPEG-2, bien que parfaitement adaptés aux applications vidéo qu'elles ciblent, n'offrent pas la flexibilité nécessaire aux applications multimedia. C'est pourquoi le groupe MPEG a débuté en 1993 des travaux sur la norme MPEG-4 [38] dans le but de construire un standard englobant un grand nombre d'applications multimedia. Nous nous intéresserons, ici, uniquement à la partie vidéo de la norme [39], le lecteur peut consulter le site officiel du groupe MPEG pour les parties système et audio [33].

L'objectif de cette norme ISO/IEC est la production et la manipulation de contenus audio-visuels avec une plus grande ré-utilisation, flexibilité et interactivité. Ces données audio-visuelles peuvent être réparties suivant quatre points:

- la nature des données (images fixes, vidéo,...),
- la nature de la source (images naturelles ou synthétiques, texte ou graphique, images médicales, images de surveillance vidéo,...),
- la nature de la communication (pouvant aller du point à point au multi-point vers multi-point),
- la nature des fonctionnalités désirées (manipulation d'objets, transmission progressive, résistance aux erreurs,...).

MPEG-4 regroupe différentes technologies jusqu'à présent indépendantes (figure 12), et se veut générique afin de pouvoir satisfaire le plus grand nombre d'applications multimedia. Les différentes applications visées par MPEG-4 peuvent être classées en cinq catégories: la télévision numérique, la production vidéo, le multimedia mobile, le jeux et le streaming vidéo. La visiophonie rentre dans la catégorie multimedia mobile.

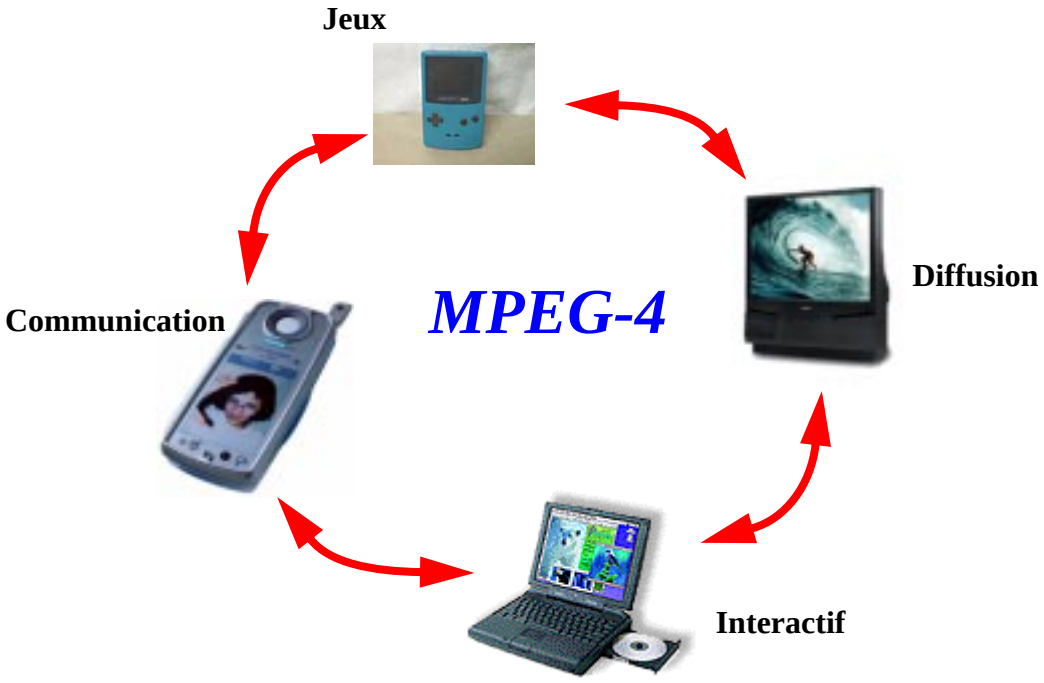


Figure 12: domaines d'applications de la norme MPEG-4

III.4.3.a) La télévision numérique

Avec la croissance de l'internet, l'intérêt de l'interactivité avec les contenus de la télévision devient manifeste. L'augmentation du contrôle par l'utilisateur de contenus textuels, audio, graphiques ou vidéo ajoute une plus-value au contenu du programme TV, ou au service associé à ce programme. La sur-impression d'informations, le multi-fenêtrage pour, par exemple, visualiser

simultanément les statistiques associées au sport regardé, l'édition de liens entre certains évènements et des documents vidéo leur faisant référence , sont des exemples d'exploitation des fonctionnalités de MPEG-4.

III.4.3.b) La production de vidéo

L'utilisation de techniques de production virtuelle en complément des techniques de clé chromatique permet une production disjointe de scènes et de composantes vidéo pouvant être mixer avec des effets spéciaux synthétiques. Une meilleure qualité et flexibilité est obtenue par une manipulation d'objets et non plus d'images, offrant des moyens plus sophistiqués pour le contrôle de la vidéo. Cette technologie permet, par exemple dans le cas de la diffusion d'émission TV nationale, d'ajouter une information régionale au programme.

III.4.3.c) Le multimedia embarqué

L'engouement pour les téléphones cellulaires et les assistants personnels offre un nouveau champ d'application pour le monde du multimedia. Pour s'adapter à ces nouvelles contraintes, MPEG-4 améliore le taux de compression à très bas débit. De plus, cette norme offre des outils pour améliorer la résistance aux erreurs (perte de paquets) en particulier pour les réseaux à qualité de service non garantie.

III.4.3.d) Les jeux

La popularité des plate-formes de jeux vidéo interactifs telles la PlayStation de Sony fournit un autre secteur d'application pour la norme. La plupart des jeux actuels sont basés sur la composition de scènes et d'objets 3D. Ainsi, MPEG-4 permet l'ajout de vidéo naturelle, ce qui rend les jeux plus réalistes. De plus, grâce à une manipulation par objet, il est désormais possible de personnaliser les jeux et d'introduire de nouveaux objets à l'intérieur de ceux-ci.

III.4.3.e) Le "streaming" vidéo

La consultation d'informations vidéo en direct ou en différé sur internet n'est qu'un exemple de streaming vidéo. MPEG-4, par l'utilisation de techniques adaptées de résistance aux erreurs et de codage satisfait aux contraintes de transmissions (perte de paquets, bande passante limitée,...), et offre également à l'utilisateur le contrôle de la vidéo grâce à une décomposition graduelle ("scalability") temporelle, spatiale et au niveau de la qualité, de chaque objet vidéo composant le train binaire.

Bien sûr, ces applications nécessitent la mise en oeuvre d'outils supportant les fonctionnalités annoncées. Le groupe de normalisation les classe en trois catégories:

1. l'interactivité basée contenu

Différentes fonctionnalités sont ici regroupées. Tout d'abord, la manipulation de contenus et l'édition du train binaire doivent être possible sans nécessiter de décodage préalable. Ensuite, le codage conjoint de la vidéo naturelle et synthétique doit être possible. Enfin, un accès aléatoire précis aux trames vidéo ou aux objets vidéo doit être garanti.

2. la compression

Deux points essentiels : MPEG-4 doit fournir une qualité de codage à débit équivalent supérieure aux standards existants, et également, un codage efficace de scène en vues multiples exploitant la redondance inter-vidéo.

3. l'accès universel

MPEG-4 doit inclure des techniques de codage spécifiques aux environnements très bruités autorisant ainsi la diffusion d'applications sur des réseaux divers (sans fils, câblés, de stocka-

ge,...). La norme doit enfin fournir les outils nécessaires à une décomposition graduelle au niveau objet tant temporelle que spatiale ou de qualité.

Le concept fondamental introduit par MPEG-4 est l'objet audio-visuel. Un objet vidéo consiste en une succession de couche de description permettant un codage "scalable". La nature "scalable" de la syntaxe permet une reconstruction de la vidéo à une qualité optimale par rapport aux contraintes de l'application et du réseau, et une adaptation de la complexité aux ressources disponibles sur le terminal cible. Une scène visuelle MPEG-4 est composée d'un ou plusieurs objets vidéo caractérisés temporellement et spatialement par le biais de leur forme, leur mouvement et leur texture. Dans certain cas, une description par objets vidéo de la scène n'est pas souhaitable du fait du surcoût que cela représente (en débit et en complexité), le codage utilise alors une décomposition en objet rectangulaire (comme pour les normes MPEG-1, MPEG-2 et H.26x).
Afin de satisfaire aux fonctionnalités promues par MPEG-4 vidéo, le train binaire est hiérarchisé en niveaux par l'intermédiaire de codes spécifiques appelés "start codes", ce qui permet un accès indépendant à chaque niveau. Les niveaux sont au nombre de cinq et agencés comme décrit figure 13.
Le niveau correspondant à l'agencement des objets visuels (VS: Visual Object Sequence) décrit l'ensemble de la scène MPEG-4 potentiellement composée d'objets naturels et/ou synthétiques, 2D et/ou 3D et le niveau de qualité associé.
Un objet vidéo (VO: Video Object), quant à lui, correspond à la description d'un objet (2D) particulier de la scène.
La couche objet vidéo (VOL: Video Object Layer) permet la scalabilité au niveau des objets vidéo. A noter que deux types de VOL existent, le premier limitant les fonctionnalités de MPEG-4 et introduisant le codage avec des entêtes courts (compatibilité avec les trains binaires H.263), et le second offrant toutes les fonctionnalités de MPEG-4.
Chaque VOL est ensuite décomposé en occurrences d'objet vidéo (VOP: Vidéo Object Planes). Un VOP est un VO à un instant t, il peut être encodé indépendamment des autres VOP (VOP de type I) ou en référence à des VOPs précédents et/ou suivants (VOP de type P ou B). Un VOP peut également décrire un "sprite" qui est un VO généralement plus large que l'image affichée, construit tout au long de la vidéo par des ajouts d'informations, des modifications du contraste et des déformations.
Ces VOPs peuvent être optionnellement regroupés en GOVs (Group of Video Object Planes) indépendants, ce qui fournit des points d'entrée pour un accès aléatoire.

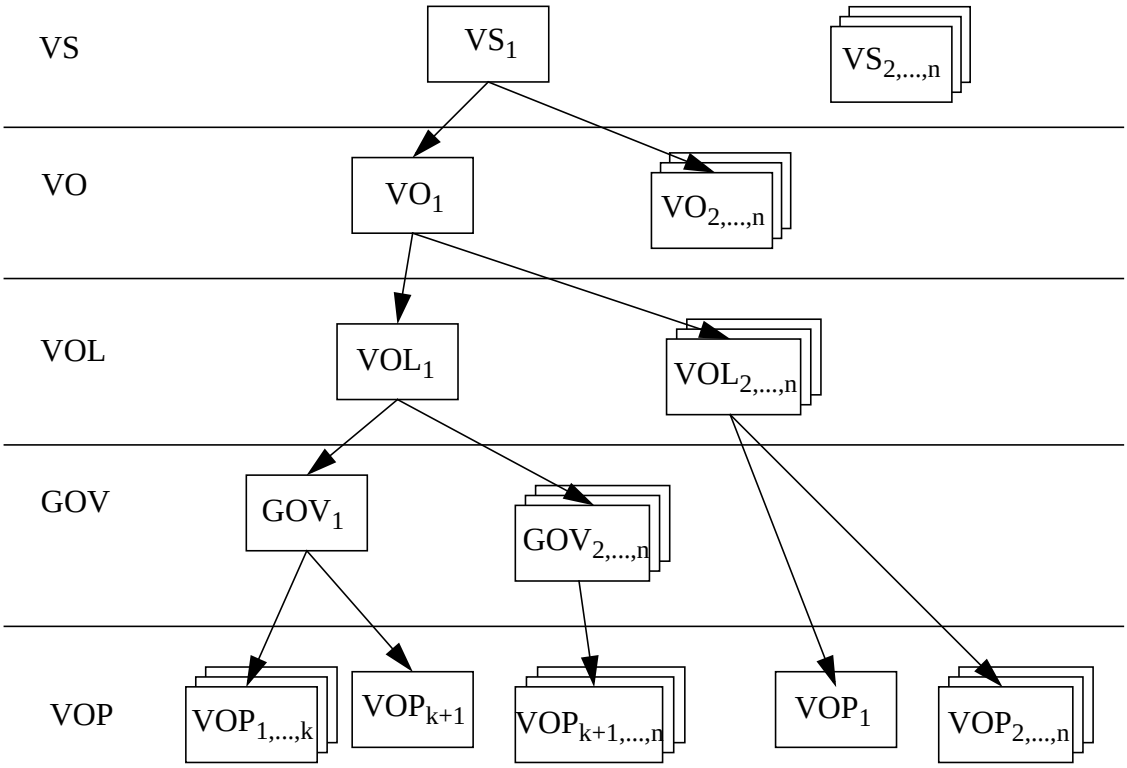


Figure 13: Exemple de structure d'un train binaire MPEG-4 vidéo.

Le standard MPEG-4 visuel est optimisé pour trois gammes de débit:
- inférieur à 64 kbit/s,
- de 64 à 384 kbit/s,
- de 384 kbit/s à 4 Mbit/s,

mais reste valide pour des débits supérieurs nécessaires au codage de haute qualité. MPEG-4 est, dans le même esprit que pour MPEG-2 vis à vis de MPEG-1, un sur-ensemble de MPEG-2. Toutes les fonctionnalités de MPEG-2 sont donc supportées avec l'ajout de la notion d'objets vidéo, de vidéo synthétique 2D ou 3D,... L'ensemble de ces fonctionnalités ne sont pas systématiquement nécessaires à la construction d'une application donnée, le groupe de normalisation reprend donc, pour MPEG-4, la structure en "profils et niveaux" mise en place lors des travaux sur MPEG-2.
Les profils définissent les outils nécessaires à la réalisation d'une application donnée, les niveaux imposent les contraintes de formats d'image, fréquences image (trame), débits, niveaux de scalabilité limitant la complexité du décodeur [40]. Ces profils et niveaux sont ensuite conjugués pour former des points de conformité. Un point de conformité (nommé profil@niveau) spécifie à un niveau particulier, un profil audio, vidéo, graphique, descripteur de scène ou d'objet permettant de tester la validité d'un codec MPEG-4. Nous nous intéressons, ci-après, aux profils vidéo. Le tableau 11 présente les différents profils de la version 1 de MPEG-4 vidéo, standard international depuis Avril 1999.

Tableau 11: Profils et outils associés

Profils Outils	Simple	Simple scalable	Core	Main	N-bit	Scalable texture	Simple face animation	Basic animated texture	Hybrid
Outils de base: - I-VOP, P-VOP[1] - prédiction AC/DC[2] - UMV, 4-MV[1]	X	X	X	X	X				X
Résistance aux erreurs[3]: - resynchronisation des slices - partitionnement des données - VLC réversible	X	X	X	X	X				X
Entête court[4]	X	X	X	X	X				X
B-VOP[1]		X	X	X	X				X
Quantification 1/2[5]			X	X	X				X
Scalabilité temporelle (P-VOP): - rectangulaire - forme quelconque			X	X	X				X
Masque binaire[6]			X	X	X			X	X
Plan Alpha[6]				X					
Vidéo entrlacée[7]				X					
Sprite[8]				X					
Scalabilité temporelle[1]		X							
Scalabilité spaciale[1]		X							
N-bit (4 à 12 bits)[9]					X				
Texture image fixe scalable[10]				X		X		X	X
Maillage 2D (topologie régulière)[11]								X	X
Maillage 2D (topologie de Delaunay)[11]									X
Paramètres d'animation faciaux[12]							X	X	X

Commentaires :
1. Outils identiques aux options de codage des normes H.263 V1, V2 et V3.
2. La prédiction des coefficients AC/DC permet de réduire l'énergie moyenne des valeurs quantifiées. Cette prédiction est faite à partir des macroblocs voisins du macrobloc à coder (Annexe A.1)
3. MPEG-4 propose quatre outils de résistance aux erreurs (Annexe A.2):
- l'insertion de marqueurs dans le train binaire comme aide à la resynchronisation,
- le partitionnement des données : cela consiste à coder séparément l'information de mouvement et l'information de texture,
- l'extension du code d'entête : on ajoute de la redondance à l'information d'entête,
- les codes à longueur variable réversibles : ils permettent de décoder le train binaire dans les deux sens.
4. Cet outil permet de supprimer l'information d'entête inutile pour les profils les plus bas (codage rectangulaire).
5. Deux méthodes de quantification sont possibles, l'une en utilisant les tables de quantification de MPEG-2, l'autre avec un pas de quantification uniforme sur chaque bloc comme pour H.263.
6. Le masque binaire définit la forme d'un VOP, le plan Alpha ajoute une information sur la transparence de l'objet (Annexe A.3)
7. Identique à MPEG-2.
8. Un sprite est un VO persistant de la scène sur une portion de vidéo et globalement statique. Un exemple de sprite est l'arrière plan.
9. Cet outil permet de choisir la dynamique des valeurs de luminance et de chrominances des pixels. Ces pixels peuvent être codés en utilisant 4 à 12 bits. Une faible précision est bien adaptée aux applications très bas débit sans contrainte de qualité telle que la vidéo-surveillance. En revanche, les applications nécessitant une qualité d'image parfaite telle l'imagerie médicale requiert une grande précision sur les valeurs des pixels.
10. MPEG-4 met en place des outils spécifiques au codage des images fixes (codage par ondelettes, nombre de niveaux de scalabilité spatiale augmenté,...).
11. Maillage 2D régulier ou de Delaunay pour le codage des VOPs utilisé, par exemple, pour le plaquage de texture sur un VOP naturel ou synthétique.
12. Cet outil spécifie des paramètres de description et d'animation de visages synthétiques.

La version 2, standard international en Février 2000, ajoute les profils suivants:

pour la vidéo naturelle

- le profil Advanced Real-Time Simple (ARTS) fournit les outils permettant une meilleure résistance aux erreurs pour le codage des VOs rectangulaires par l'utilisation du canal de retour et une stabilité de la fréquence image accrue grâce à une file d'attente de faible délai en sortie. Les applications visées sont le vidéophone, la télé-conférence et le contrôle à distance.
- le profil Core Scalable ajoute au profil core la scalabilité temporelle et spatiale au niveau VO. Le principal apport est de pouvoir exploiter la scalabilité sur les objets d'intérêt de la scène, très utile pour les applications pour l'Internet, le mobile et la diffusion.
- le profil Advanced Coding Efficiency (ACE) a pour but l'amélioration du codage des VOs rectangulaires et de forme quelconque par l'ajout des outils suivants:
  - la compensation de mouvement au quart de pixel permettant de réduire davantage les effets blocs,
  - la compensation de mouvement globale (GMC: Global Motion Compensation) basée sur la notion de "sprite" permet de ne transmettre que 8 paramètres de mouvement (correspondant à une transformation affine) pour décrire la déformation du "sprite".
  - la compensation de mouvement par blocs recouvrants (OBMC) décrite dans les options de

la norme H.263,

- la transformée en cosinus discrète adaptée aux objets de forme quelconque (SA-DCT: Shape Adaptive DCT) est une décomposition sur une base orthonormée de transformées en cosinus discrètes à une dimension.

L'objectif étant de satisfaire les besoins d'efficacité de codage requis pour des applications telles que la réception d'émissions TV sur un terminal mobile, l'enregistrement vidéo avec un caméscope.

pour la vidéo naturelle et/ou de synthèse

- le profil Advanced Scalable Texture supporte le décodage d'images fixes et de textures de forme quelconque, ce qui inclus le codage graduel de forme, la décomposition par ondelettes et la résistance aux erreurs. Des exemples d'utilisation sont la consultation rapide sur Internet, les PDA avec des fonctionnalités multimedia et les appareils photos numériques.
- le profil Advanced Core regroupe le profil core et le profil advanced scalable texture. Il vise essentiellement les applications de streaming interactif sur Internet.
- le profil Simple Face and Body Animation est un sur-ensemble du profil "Simple Face" auquel il ajoute les outils nécessaires à l'animation de corps de clone.

Les versions les plus récentes (Mars 2001) ajoutent quatre autres profils :

- le profil Advanced Simple englobe le profil simple (VOP rectangulaire) en y ajoutant des outils augmentant l'efficacité du codage: images B, compensation de mouvement au quart de pixel, compensation de mouvement globale et OBMC.
- le profil Fine Granular Scalability autorise jusqu'à 8 niveaux de scalabilité permettant une meilleure adaptation de la qualité aux conditions de transmission et de décodage. Le profil sous-jacent peut être le profil "Simple" ou le profil "Advanced Simple".
- le profil Simple Studio cible les applications de production vidéo de très haute qualité. Ce profil ne supporte que les VOPs de type I qui peuvent être de forme quelconque et multi-canaux. Le débit de la vidéo peut atteindre 2 Gbits/s.
- le profil Core Studio ajoute au profil "Simple Studio" les VOPs de type P.

Les niveaux associés aux profils de vidéo naturelle et synthétique de la version 1 sont présentés tableau 12.

Tableau 12: profils et niveaux associés

Profils	Niveaux	Résolution	Débit maxi (bits/s)	Nombre de VOP	Mémoire(Nb MBs)	Niveaux de scalabilité
Simple	L1	QCIF	64 k	4	198	-
	L2	CIF	128 k	4	792	-
	L3	CIF	384 k	4	792	-
Simple scalable	L1	CIF	128 k	4	495	1 spatiale ou temporelle
	L2	CIF	256 k	4	792	1 spatiale ou temporelle

Tableau 12: profils et niveaux associés

Profils	Niveaux	Résolution	Débit maxi (bits/s)	Nombre de VOP	Mémoire(Nb MBs)	Niveaux de scalabilité
Core	L1	QCIF	384 k	4	594	1 temporelle
	L2	CIF	2 M	16	2376	1 temporelle
Main	L2	CIF	2 M	16	2376	1 temporelle
	L3	ITU-R 601	15 M	32	9720	1 temporelle
	L4	1920x1088	38.4 M	32	48960	1 temporelle 2 spatiales
N-bit	L2	CIF	2 M	16	792	1 temporelle
Scalable texture	L1	4CIF (8 bits)	-	-	-	5 spatiales
	L2	2048x2048 (10 bits)	-	-	-	8 spatiales
	L3	8192x8192 (12 bits)	-	-	-	10 spatiales
Simple face animation	L1	QCIF	16 k	1	-	-
	L2	CIF	32 k	4	-	-
Basic animated texture	L1	QCIF, 4CIF fixe (8 bits)	128 k	4	-	5 spatiales
	L2	CIF, 2048x2048 fixe (10 bits)	128 k	8	-	8 spatiales
Hybrid	L1	QCIF, 4CIF fixe (8 bits)	64 k	4	-	1 temporelle 5 spatiales
	L2	CIF, 2048x2048 fixe (10 bits)	128 k	8	-	1 temporelle 8 spatiales

Les profils de la vidéo naturelle spécifiés pour MPEG-4 permettent de différencier deux catégories de codeur. Ceux incluant des VOPs de forme quelconque et ceux travaillant sur un seul VOP de forme rectangulaire. La figure 14 illustre ces deux configurations. La forme la plus générique, introduisant le codage de forme (codage arithmétique du masque binaire ou du plan Alpha), nécessite une phase de pré-traitement afin d'extraire les VOPs de la scène vidéo. Cependant, les moyens de segmentation et d’étiquetage des VOPs ne sont pas spécifiés par la norme. La deuxième configuration, correspondant aux profils "Simple", "Simple Scalable" et "Advanced Real Time Simple", est identique, dans ses principes, à un codeur H.263 (ou MPEG-2), mais diffère par la syntaxe du train binaire produit, par les techniques de résistance aux erreurs, et par des techniques de codage permettant d'augmenter le taux de compression.

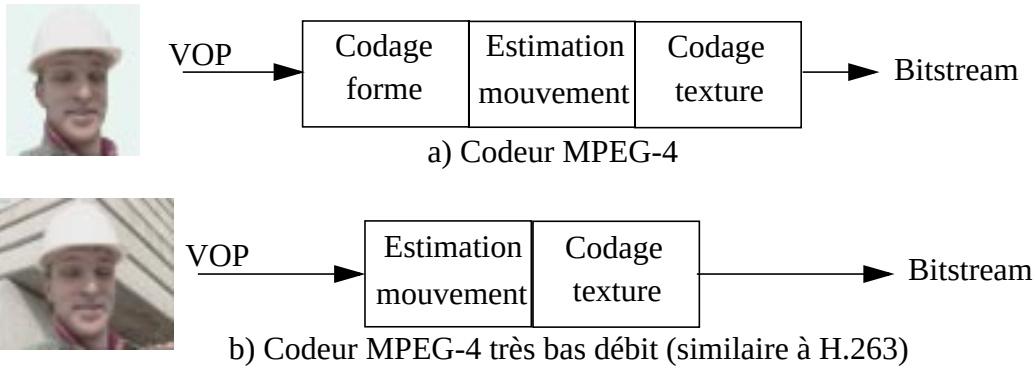


Figure 14: a) codage par objets, b) codage rectangulaire

Les profils définis pour l'application visée dans le cadre de cette étude sont:
- "Simple",
- "Simple Scalable",

pour la version 1 de MPEG-4,
- "Advanced Real Time Simple",
- "Advanced Simple",

pour les versions suivantes.
Ces profils sont adaptés aux très bas débits, aux environnements bruités et à la faible complexité. Les figures 15 et 16 situent cette catégorie de codeur en fonction de ses fonctionnalités et des débits visés par rapport aux autres profils de MPEG-4.

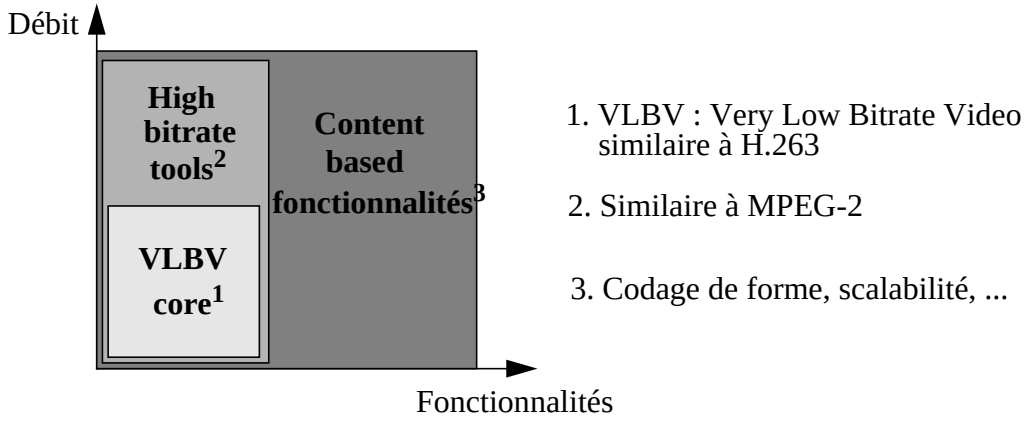


Figure 15: classification des outils et algorithmes de codage MPEG-4

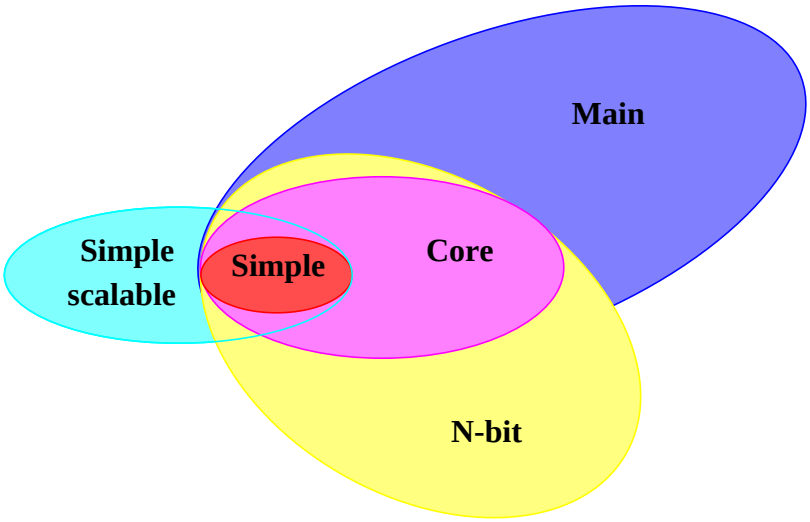


Figure 16: profils visuels de MPEG-4 V1

Comme nous venons de le voir, le profil correspondant à une application de visiophonie mobile est le profil "Simple" ("Advanced Real Time Simple" pour la version 2) semblable à la norme H.263 de l'UIT-T. Le tableau 13 résume les fonctionnalités des deux normes (MPEG-4 SP et H263) et met en avant leurs ressemblances et différences.

Tableau 13: H.263 vs MPEG-4

Outils/Options	H.263			MPEG-4			
	V1	V2	V3	SP	SSP	ARTSP	ASP
UMV / 4-MV	X	X	X	X	X	X	X
Prédiction AC/DC		I	I	I/P	I/P	I/P	I/P
Resynchronisation des "slice"		X	X	X	X	X	X
Partition des données		X	X	X	X	X	X
VLC réversible				X	X	X	X
Entête court(*)				X	X	X	X
Codage arithmétique		X	X				
OBMC (**)	X	X	X				X
image PB	X	X	X				
Information de retour		X	X			X	
Délai tampon court						X	
Filtre dans la boucle de codage		X	X				

Tableau 13: H.263 vs MPEG-4

Outils/Options	H.263			MPEG-4			
	V1	V2	V3	SP	SSP	ARTSP	ASP
Scalabilité temporelle, spatiale, SNR		X	X		X		
VLC Inter alternatif		X	X				
Quantification flexible		X	X				
GMC							X
1/4 de pixel							X
image B	X	X	X		X		X
Sélection de l'image de référence Ré-échantillonnage de la référence Codage en résolution réduite		X	X				

(*) Cet outil est utilisé dans MPEG-4 pour une compatibilité avec les codeurs H.263.
(**) OBMC : "Overlapped Block Motion Compensation".
NB : la précision de la compensation en mouvement est au moins du demi-pixel pour tous les profils ou versions.
SP : "Simple Profile"
SSP : "Simple Scalable Profile"
ARTSP : "Advanced Real-Time Simple Profile"
ASP : "Advanced Simple Profile"

IV. Bibliographie

[1] K. Kobayashi, 'MobileTerminals and Devices Technology for the 21st Century' Special review paper – A new start in the 21st Century, NEC Res. & Develop., Vol. 42, No.1, Janvier 2001.

[2] 3GPP : http://www.3gpp.org

[3] Report on UMTS, 'Final Draft 2.5 enabling UMTS (3G) services and applications', No. 11, Report from the UMTS Forum, Juillet 2000.

[4] Digital cellular telecommunication system (phase 2+) (GSM), 'Multiplexing and multiple access on the radio path (GSM 05.02 version 6.4.1, release 1997)', document ETSI EN 300 908 V6.4.1.

[5] 'MPEG-4 video verification model version 16.0', ISO/IEC JTC1/SC29/WG11 N3312, Mars 2000.

[6] J. G. Beerends et F. E. De Caluwe, 'The influence of video quality on perceived audio quality and vice versa', Journal Audio Eng. Soc., Vol. 47, No. 5, Mai 1999.

[7] Kyocera VP-210 : http://www.kyocera.co.jp/news/1999/9905/003-e.asp

[8] Orange videophone : http://www.vnunet.com/Products/Hardware/1123396, http://www.orange.co.uk

[9] Bluetooth : http://www.bluetooth.com

[10] NEC WCDMA vidéophone : http://www.nec.co.jp/press/en/9909/2201.html

[11] M. Harrand et al., 'A single chip CIF 30 Hz, H.261, H.263 and H.263+ video encoder / decoder with embedded display controller', IEEE Journal of Solid State Circuits, Vol. 34, No. 11, pp 1627-1633, Novembre 1999.

[12] J. Smit, 'Energy complexity and architecture', 7th International workshop on Power, Modelling, Optimisation, Simulation (PATMOS), Louvain La Neuve, Septembre 1997.

[13] H. Bouma, 'Design and implementation of an FPFA', M. Sc. Thesis, University of Twente, department of electrical engineering, laboratory of signals and systems, Juillet 2001.

[14] Architecture Chameleon : http://www.chameleonsystems.com

[15] D. Patterson et al. 'Intelligent RAM (IRAM): chips that remember and compute', 1997 IEEE International Solid State Circuits Conference, san Francisco, CA, 6-8 Février 1997.

[16] T. Callahan, J. Hauser, J. Wawrzynek, 'The GARP architecture and C compiler', IEEE computer, Avril 2000.

[17] Berkeley Design Technology Inc., 'The evolution of DSP processors', lecture presented to U.C., Berkeley CS 152, Avril 2000, http://www.bdti.com

[18] M. Budagavi et al., 'The wireless MPEG-4 on Texas Instruments DSP Chips', DSP Solutions R&D Center, Texas Instruments Inc., http://www.ti.com

[19] 'TMS320C55x DSP functional overview', Texas Instruments, No. SPRU312, Juin 2000, http://www.ti.com

[20] StarCore SC140 : http://www.starcore-dsp.com

[21] Analog Devices, Blackfin DSP, Preliminary technical data : ADSP-21532, Septembre 2001, http://www.analog.com

[22] T. Nishikawa et al., 'A 60 MHz, 240 mW MPEG-4 videophone LSI with 16 Mb embedded DRAM', Int. Solid State Circuits Conf. (ISSCC), San Fransisco, 2000.

[23] T. Hashimoto et al., 'A 90 mW MPEG-4 video codec LSI with the capability for core profile', Int. Solid State Circuits Conf. (ISSCC), San Fransisco, 2001.

[24] C. Yoon et al., 'A 80 / 20 MHz, 160 mW multimedia processor integrated with embedded DRAM, MPEG-4 accelerator, and 3D rendering engine for mobile applications', Int. Solid State Circuits Conf. (ISSCC), San Fransisco, 2001.

[25] J. Chaoui et al., 'OMAP : enabling multimedia applications in third generation (3G) wireless terminals', technical white paper, SWPA001, Décembre 2000, http://www.ti.com

[26] Geo Emblaze A3 : http://www.emblaze.com/serve/products/a3.asp

[27] E. Batut, 'A low complexity channel estimation scheme for WB-CDMA terminal', 3rd Workshop on signal Processing Advances in Wireless Communications (SPAWC), Taiwan, 20-23 Mai 2001.

[28] F. Catthoor, F. Fransenn, S. Wuytack, L. Nachtergaele et H. De Man, 'Global communication and memory optimizing transformations for low power signal processing systems', IEEE workshop on VLSI signal processing, La Jolla, Octobre 1994.

[29] W. Mangione-Smith et al., 'A low power architecture for wireless multimedia systems: lessons learned from building a power hog', Int. Symp. on Low Power Electronics and Design (ISLPED), pp 23-28, Monterey, 1996.

[30] T.H. Meng, B.M. Gordon, E.K. Tsern et A.C. Hung, 'Portable video on demand in wireless communication', Proceedings of the IEEE, Vol. 83, No. 4, pp 659-680, Avril 1995.

[31] T. Ebrahimi et M. Kunt, 'Visual data compression for multimedia applications', Proceedings of the IEEE, Vol. 86, No. 6, pp 1109-1125, Juin 1998.

[32] UIT-T : Union Internationale de Télécommunications – Télécommunications, http://www.itu.ch

[33] MPEG : Motion Picture Expert Group, http://www.mpeg.telecomitalialab.com, http://www.mpeg.org

[34] 'line transmission of non telephone signals : Video codec for audiovisual services at p x 64 kbit/s', ITU-T Recommendation H.261 (03/93).

[35] 'Video coding for low bit rate communication', Serie H : Audiovisual and multimedia systems, infrastructure of audiovisual services – coding of moving video, ITU-T, recommendation H.263, Février 1998.

[36] 'MPEG-1 video group – coding of moving pictures and associated audio for digital storage media at up to 1.5 Mbit/s – video', ISO/IEC JTC1/SC29/WG11 11172-2, International standard, Geneva, 1993.

[37] 'MPEG-2 video group – generic coding of moving pictures and associated audio', ISO/IEC JTC1/SC29/WG11 13818-2, International standard, 1995.

[38] 'MPEG-4 video group – generic coding of audio-visual objects', ISO/IEC JTC1/SC29/WG11 14496-2, International standard, Atlantic City, 1998.

[39] T. Ebrahimi, C. Horne, 'MPEG-4 natural video coding – An overview’, Signal processing: image communication 15, pp365-385, 2000.

[40] R. Koenen, ‘Profiles and levels in MPEG-4: Approach and overview’, Signal processing: image communication 15, pp 463-478, 2000.

[41] VSIA : Virtual Socket Interface Alliance, http://www.vsi.org

Chapitre B: Faisabilité d'intégration de la visiophonie mobile

Afin d'étudier les performances de codage nous allons, dans la première partie de ce chapitre, préciser les différentes configurations étudiées. Nous distinguons ici deux aspects. Le premier concerne les outils (au sens MPEG-4) mis en oeuvre pour le codage. Le deuxième, directement lié à la contrainte de faible complexité de l'application, concerne l'algorithme de compression vidéo et en particulier l'estimation de mouvement.
Ensuite, dans la deuxième partie, nous définirons les outils informatiques utilisés pour les mesures de performances. Nous interpréterons les résultats obtenus sur différentes séquences vidéo typiques de scènes de visiophonie.
Enfin, dans la troisième partie, nous aborderons les aspects architecturaux du système de codage vidéo. Nous définirons une méthodologie de conception autour des analyses algorithmiques et architecturales et nous l'illustrerons sur l'exemple du codeur vidéo.

I. Les configurations étudiées

Le codeur étudié est basé sur un source C développé par l'université de British Columbia, Canada [1] satisfaisant à la norme H.263+ [2]. Le source C utilisé pour réaliser notre étude n'est pas exactement le source originel (18 000 lignes de C). En effet, une version optimisée a été développée afin de prendre en compte les contraintes de l'embarqué (accès mémoire, architecture en virgule fixe du processeur).
Par ailleurs, les performances de codage sont liées aux techniques de prédiction employées qui sont fixées à la fois par la norme et par le concepteur de l'application.

• La norme H.263 spécifie des options de codage, notre codeur met en oeuvre les options suivantes :
- vecteur mouvement non restreint (H.263 Annexe D),
- codage arithmétique (H.263 Annexe E),
- prédiction avancée (H.263 Annexe F),
- codage d'images PB et PB améliorées (H.263 Annexes G et M),
- mode intra avancé (H.263 Annexe I),
- filtrage anti-blocs (H.263 Annexe J),
- sélection de l'image de référence (H.263 Annexe N),
- codage à longueur variable inter alternatif (H.263 Annexe S),
- quantification modifiée (H.263 Annexe T),
- décomposition graduelle aux niveaux SNR, spatiale et temporelle (H.263 Annexe O),
- découpage en "slice" (Annexe K),
- rafraîchissement intra au niveau image et/ou macrobloc.

Ce large éventail d'options nous permet de réaliser une première étude des performances de codage pour différentes configurations du codeur qui sont précisées dans le paragraphe I.1.

• En revanche, la norme H.263 ne spécifie pas la technique d'estimation de mouvement qui peut donc être choisie par le concepteur. Les algorithmes de codage basés sur la compensation de mouvements par blocs font appel comme nous l'avons vu à un estimateur de mouvement des macroblocs composant l'image. Cet estimateur recherche le bloc le plus ressemblant dans une fenêtre de recherche d'une image de référence par calculs de distorsions. La mesure de distorsion est généralement la somme des valeurs absolues des différences entre les pixels du macrobloc de luminance courant et ceux du macrobloc de référence. La fenêtre de recherche est, le plus souvent, de +/- 15 pixels autour de la position du macrobloc courant.
Pour une image QCIF (176*144 pixels), une recherche exhaustive des vecteurs mouvements nécessite donc le calcul de 77439 distorsions. Un calcul de distorsion faisant appel à 768 opérations élémentaires (additions, soustractions, valeurs absolues), la recherche des vecteurs mouvements d'une image nécessite environ 60 millions d'opérations (équivalent à 900 MOPS pour le codage d'une vidéo en QCIF @ 15 Hz). Heureusement, il existe plusieurs stratégies de recherche des vecteurs mouvement conduisant à des complexités plus faibles, comme nous le verrons au paragraphe I.3.

• De plus, l'application visée, à savoir la visiophonie mobile, conduit à fixer certains paramètres :
- la résolution choisie est le QCIF (144*176 pixels), elle correspond à un bon compromis entre la quantité d'information à compresser et la résolution d'affichage acceptable sur un mobile.
- le débit cible est fixé à 40 kbits/s ce qui correspond, comme nous l'avons vu dans le chapitre

précédent, à une valeur réaliste pour les prochains déploiements des réseaux GPRS [3] et UMTS [4].
- la fréquence image cible est fixée à 15 Hz afin de maintenir une bonne fluidité de l'image tout en restant à un taux de compression permettant une qualité acceptable de la vidéo (taux de compression voisin de 100).
- la régulation du débit est conforme à la recommandation TMN8 [5] (Test Model Near term 8), elle est détaillée dans le paragraphe I.2.3.

I.1 Les options de codage normalisées

Nous avons choisis d'étudier différentes configurations du codeur H.263 ainsi qu'une version approchée du profil Simple de la norme visuelle de MPEG-4 [6].

I.1.1 Recommandations H.263

Le comité de normalisation de l'UIT-T propose différentes configurations de codage associant diverses options complémentaires [2]. Elles correspondent à des recommandations spécifiées ci-après. Dans le cadre de cette thèse, nous n'avons pas cherché à respecter scrupuleusement ces recommandations qui auraient nécessitées de développer d'autres options. Toutefois, nous avons essayé de satisfaire au mieux ces configurations. Nous n'avons pas choisi pour nos tests d'utiliser la notion de profils et niveaux introduite trop récemment par l'annexe X de la norme H.263++.

I.1.1.a) L1

Cette première configuration met en oeuvre le codage intra avancé (H.263+ Annexe I), le filtre de la boucle de codage (H.263+ Annexe J), le mode de quantification modifiée (H.263+ Annexe T) et le mode d'ajout d'informations (H.263+ Annexe L). Cette recommandation correspond au profil 1 défini dans H.263 annexe X.
Pour l'étude de ce cas, seul l'annexe L n'est pas implanté.

I.1.1.b) L2

La deuxième recommandation propose d'utiliser conjointement le mode du vecteur mouvement non restreint (H.263+ Annexe D), le mode de structure en slice (H.263+ Annexe K) et le mode de ré-échantillonnage de l'image de référence (H.263+ Annexe P).
L'annexe P n'est pas implanté.

I.1.1.c) L3

L'UIT-T recommande pour cette configuration le mode de prédiction avancé (H.263+ Annexe F), le codage en images PB amélioré (H.263+ Annexe M), le mode permettant un décodage par segments indépendants (H.263+ Annexe R) et le codage à longueur variable inter alternatif (H.263+ Annexe S).
Les tests sur cette recommandation ne prennent pas en compte l'annexe R.

I.1.2 MPEG-4 Profil Simple

Ce profil correspond, comme nous l'avons vu dans la partie précédente, aux applications vidéo sur les terminaux mobiles. Il s'agit d'un codage basé bloc mettant en oeuvre les outils suivants :
- la prédiction avancée (AP),
- la prédiction de mouvement non restreinte (UMV),
- la prédiction des coefficients AC/DC,
- l'utilisation d'entêtes courts,
- les outils de résistances aux erreurs : partitionnement des données, resynchronisation des slices

et codage à longueur variable réversible.
Aux vues des similitudes entre ce profil et la norme H.263, nous avons choisi comme approximation du codeur MPEG-4 "Simple Profil", la norme H.263 incluant les options suivantes :
- la prédiction avancée,
- la prédiction de mouvement non restreinte,
- le codage intra avancé,
- le codage inter alternatif.
De plus, le niveau choisi est le niveau 1 correspondant à la résolution QCIF.

I.2 La régulation du débit

Le comité de normalisation de l'UIT-T sur le codage vidéo recommande des techniques de régulation du débit sous la forme de "Test Model, Near term" [5].

I.2.1 Codage à débit variable

Ce mode utilise des pas de quantification constants sur l'image et entre les images afin de fournir un niveau de qualité constant de la vidéo. Cette approche n'est intéressante que pour le développement et la validation d'options de codage. En effet, une fois le pas de quantification fixé, on dispose d'un environnement de test valable pour la comparaison des qualités subjective et objective du codage.

I.2.2 Codage à débit fixe

Les recommandations suivantes sont proposées pour un codage à débit fixe. La régulation est réalisée par un saut d'image(s) et/ou par une adaptation de la valeur du pas de quantification. La méthode retenue pour l'adaptation du pas de quantification dépend entre autre du mode de codage adopté. En effet, le macrobloc courant ne peut pas toujours être codé avec le pas le mieux adapté. La valeur du pas de quantification (QP) est indiquée à chaque nouvelle image (toutes les valeurs sont permises entre 1 et 31). Il est également possible faire varier ce pas de +/- 2 à chaque nouveau GOB et optionnellement à chaque nouveau macrobloc. Une version étendue du codeur permet des variations non contraintes du pas de quantification (dans la mesure où il reste compris entre 1 et 31).

I.2.2.a) Recommandation TMN5

Cette régulation effectue des sauts d'images si le nombre de bits disponible est jugé insuffisant et ré-actualise le pas de quantification à chaque nouveau GOB. L'allocation tient compte du nombre de bits alloués à l'image précédente et définit un modèle de variation.
La première image est codée en mode intra avec un pas de quantification fixe généralement égal à 16. Les images P sont codées avec un pas de quantification calculé selon le modèle décrit dans la recommandation TMN5 de la norme H.263.

I.2.2.b) Recommandation TMN8

Pour cette méthode de contrôle deux niveaux de régulation sont prévus. Le premier se situe au niveau trame et définit le nombre de bits disponibles par trame, le second au niveau macrobloc adapte la valeur du pas de quantification afin de satisfaire la contrainte de débit [6].

A l'étape d'initialisation, le nombre de bits contenus dans le buffer de sortie est nul (W=0). Deux paramètres K et C servent à ajuster le modèle statistique à chaque nouveau MB. K permet d'ajuster le pas de quantification en fonction de la valeur de la variance sur les pixels du MB courant par rapport à la variance sur le reste de l'image. C permet de prendre en compte le surcoût du au codage de la chrominance. La première image n'est pas affectée par la régulation,

elle est codée en mode Intra avec un pas de quantification fixe (généralement égal à 15) afin de fournir une bonne référence pour la prédiction des images suivantes.

i. Régulation niveau trame

Le but à ce niveau est de connaître le niveau de remplissage du buffer de sortie afin de rectifier l'allocation des bits si l'objectif de débit n'est pas satisfait. On cherche à allouer R/F bits par image.
A chaque nouvelle image, on regarde donc si le nombre de bits alloué à l'image précédente dépasse ou non la valeur idéal pour le débit ciblé (R) à la fréquence fixée (F). Si l'objectif d'allocation est dépassé, on ajoute cet écart à la valeur du buffer de sortie pour le codage de l'image courante (W), sinon on considère l'objectif atteint.
Si plus d'une image est stockée dans le buffer de sortie c'est à dire W > R/F, alors on saute des images. Le nombre d'images sautées est calculé de manière à ramener le niveau de remplissage du buffer de sortie à une valeur inférieure à ce seuil.
Ensuite, le nombre de bits disponibles (B) pour le codage de l'image courante est calculé en fonction de la valeur idéale (R/F) et du niveau de remplissage du buffer de sortie souhaité.

ii. *Régulation niveau macrobloc*

Pour chaque image, on commence par calculer la variance de la luminance sur toute l'image (cette variance étant pondérée par le nombre de bits disponibles par pixel).
A chaque nouveau macrobloc, on vérifie si le nombre de bits jusqu'à présent alloué satisfait toujours au débit cible. Si c'est le cas, alors, le nouveau pas de quantification est calculé en fonction du modèle de variation de la luminance et du nombre de bits restants pour coder l'image courante (ce calcul prend en compte les restrictions sur la variation du pas de quantification spécifiées par la norme H.263). Si l'objectif n'est pas satisfait, alors, le pas de quantification est mis à sa valeur maximale. Les paramètres du modèle sont alors mis à jour puis le cas du macrobloc suivant est traité.

Cette méthode a été choisi pour les études de faisabilité réalisées sur le codeur H.263, car elle est très flexible du fait du calcul du pas de quantification à chaque macrobloc et prend ainsi en compte la nature de l'image. En effet, plus de bits sont généralement alloués aux macroblocs contenant de fort contraste.

Une version en virgule fixe de cet algorithme a été réalisée en vu d'une implantation matérielle future.
Une représentation en nombres fractionnaires positifs sur 32 bits a été retenue. Les 25 bits de poids fort correspondent à la partie entière, les 7 bits de poids faible restant représentent la partie fractionnaire. Un nombre fractionnaire, $S_{n,m}$, s'écrit comme suit en complément à 2 :

$$S_{n,m} = -s_n \cdot 2^{n-m} + \sum_{i=0}^{i=n-1} s_i \cdot 2^{i-m} \qquad s_i \in [0, 1] \qquad \text{(B.1)}$$

Dans notre cas, $S_{32,7}$ est positif et s'écrit donc :

$$S_{32,7} = \sum_{i=-7}^{i=24} s_i \cdot 2^{i} \qquad s_i \in [0, 1] \qquad \text{(B.2)}$$

Cette précision est suffisante pour construire le modèle de régulation. La figure 17 représente le nombre de bits alloués par image sur la séquence Carphone pour la version en virgule flot-

tante et la version en virgule fixe de la régulation de débit TMN8.

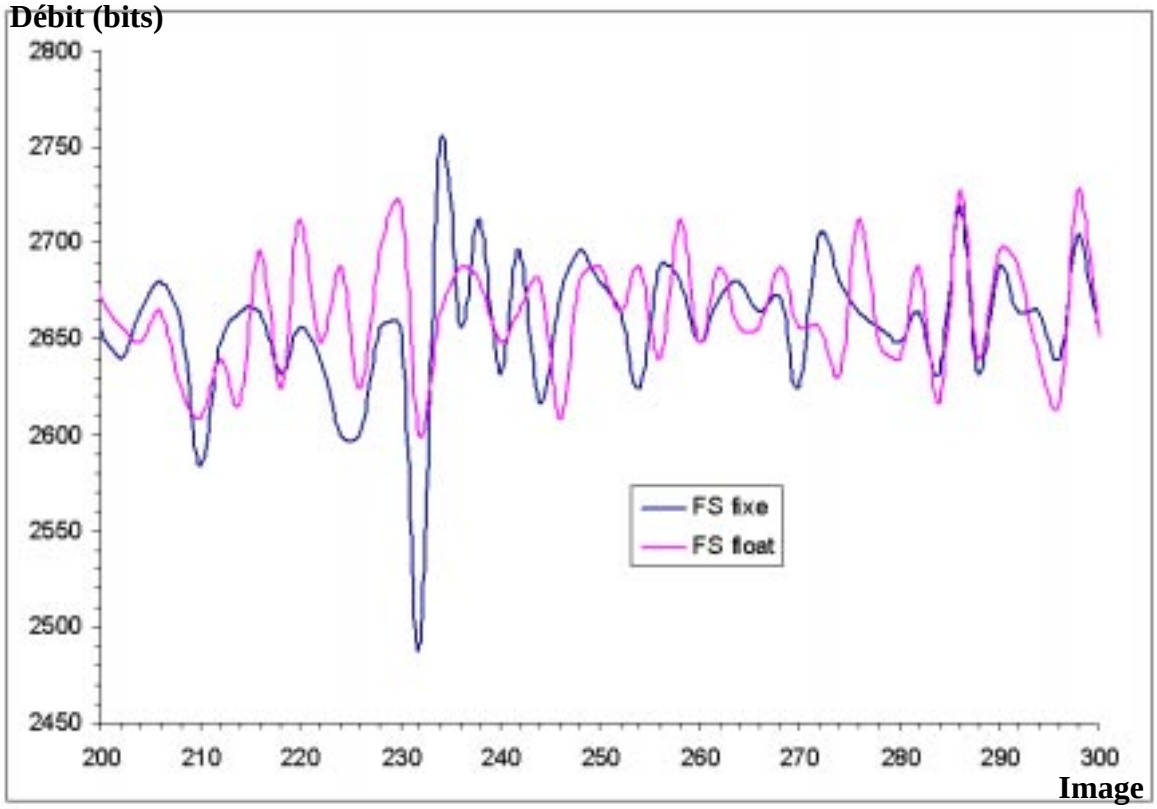


Figure 17: allocation des bits par image (séquence Carphone)

Du fait du caractère récursif des estimateurs, une erreur même faible sur la précision des calculs conduit à une régulation différente. Pour autant, l'implémentation en virgule fixe ne fournit pas de moins bonnes performances. Deux résultats de simulations sont fournis, un sur la séquence Carphone (180 images @ 15 Hz @ 40 kbit/s, figure 17) et un sur la séquence Miss America (annexe B.1). Le nombre d'images sautées ainsi que le nombre moyen de bits alloués par image sont les mêmes, des écarts négligeables sont obtenus sur l'écart type et l'étalement du nombre de bits alloués par image.

I.2.2.c) Régulation du débit "hors ligne"

Ce mode de régulation ne saute aucune image. La fréquence image, à la différence des deux modes de régulation précédents, est fixe. Le but de cette technique est de satisfaire le débit cible en moyenne sur toute la séquence. Elle est utilisée pour l'archivage vidéo, c'est pourquoi on parle de régulation hors ligne. Pour cette recommandation, le début de la régulation peut intervenir après un certain nombre d'images, fixé par l'utilisateur.
Cependant, différentes conditions de codage peuvent faire que le débit visé ne soit pas satisfait :
- une fréquence image trop élevée,
- un pas de quantification initial trop faible,
- un début de la régulation trop tardive,
- une séquence codée trop courte (la régulation n'a pas le temps d'agir).

I.3 L'estimation de mouvement

Dans les codages par estimation/compensation de mouvement, la technique d'estimation de

mouvement utilisée influe significativement sur le gain en compression et/ou sur la qualité de l'image. L'estimation de mouvement est la prédiction des pixels de l'image courante à partir des pixels de l'image précédemment reconstruite. La plupart des techniques d'estimation de mouvement reposent sur les principes suivants :
- la luminosité est uniforme le long des déplacements : la variation de luminosité d'un pixel au cour du temps est négligeables, cette variation correspond à un changement visuel mais pas à un réel déplacement.
- les problèmes d'occlusion sont négligés. Ces problèmes se produisent lorsqu'une partie cachée de l'arrière plan apparaît, dans ce cas, aucune région de l'image de référence ne lui correspond. Toutefois, ce cas se produit "rarement" dans une scène vidéo réelle.

Les différentes méthodes d'estimation de mouvement peuvent être regroupées en trois grandes catégories :

- les méthodes basées sur l'équation des flux optiques [7].
Ces méthodes effectuent une analyse des gradients spatiaux-temporels de la luminance *Y* des pixels. Elles reposent sur l'hypothèse que les changements de luminosité en chaque point de l'image sont uniquement dus à un déplacement. Soit un pixel $x(x_1,x_2)$ à l'instant *t*, l'équation des flux optiques s'écrit:

$$dY = \frac{\partial Y(x,t)}{\partial x_1} \cdot dx_1(t) + \frac{\partial Y(x,t)}{\partial x_2} \cdot dx_2(t) + \frac{\partial Y(x,t)}{\partial t} \cdot dt \qquad \text{(B.3)}$$

pour Δt fixé, on cherche $(\Delta x_1, \Delta x_2)$ tel que $\Delta Y = 0$

Cette équation est insuffisante pour calculer le mouvement des pixels, différentes hypothèses supplémentaires sont proposées dans la littérature pour résoudre ce problème, comme les solutions de Lucas-Kanade [8], de Horn-Schunck [9] et de Nagel [10].

- les méthodes pixels-récursives [11].
Le but est dans ce cas de converger vers un minimum de la différence entre l'image courante et une image compensée en mouvement. Les estimations de déplacement sont faites de façon récursive au niveau pixel et sont du type prédiction-correction de la forme:

$$v(x, t+\Delta t) = v_{prev}(x, t+\Delta t) + u(x, t+\Delta t) \qquad \text{(B.4)}$$

où
- $v(x, t+\Delta t)$, est l'estimation du déplacement
- $v_{prev}(x, t+\Delta t)$, est la prédiction du déplacement
- $u(x, t+\Delta t)$, est la correction de déplacement
- Δt, est l'indice temporel de l'image de référence

L'estimateur de mouvement du pixel *x* à l'instant *t* est ainsi ré-actualisé jusqu'à convergence. L'algorithme pixel-récursif vise à minimiser la différence entre l'image compensée en mouvement et l'image courante:

$$min_{v(x,t)}(Y(x + v(x, t+\Delta t), t+\Delta t) - Y(x,t)) \qquad \text{(B.5)}$$

Cette méthode conduit à un champ dense de vecteurs mouvement souvent incompatible avec la contrainte de compression.

- les méthodes par appariement de blocs
Elles sont largement utilisées dans les codeurs vidéo normalisés du fait de leur meilleurs compromis entre complexité et efficacité de codage, et de leur plus grande facilité d'implantation. C'est pourquoi, nous considérerons par la suite uniquement ces méthodes.

I.3.1 Recherche par appariement de blocs : généralités

Ces méthodes sont basées sur la comparaison d'un bloc de l'image courante aux blocs d'une fenêtre de recherche d'une image de référence (précédente ou suivante, figure 18). Elles visent à minimiser un certain critère de distorsion (erreur quadratique, erreur absolue, ...). Le choix de l'opérateur de distorsion n'est pas imposé par les normes des comités de standardisation (MPEG, UIT-T) mais le plus souvent la somme des valeurs absolues des différences (SAD) sur la luminance des pixels est utilisée. L'équation ci-dessous, correspond au calcul d'une distorsion pour un MB de 16*16 pixels.

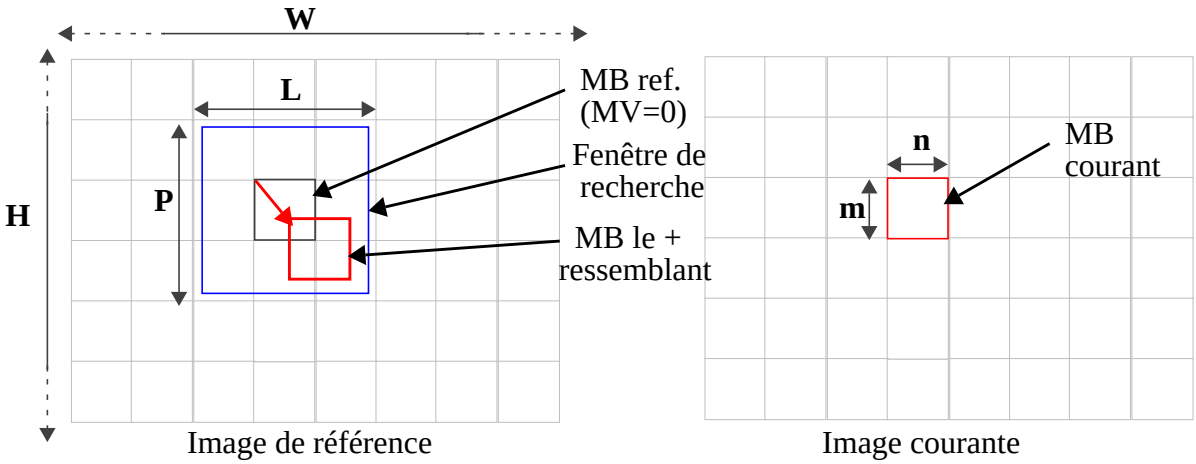
$$SAD(x,y) = \sum_{j=0}^{15}\sum_{i=0}^{15} \left| Ycur_{i,j} - Yref_{x+i,y+j} \right| \qquad (B.6)$$

$Ycur$ est la luminance du macrobloc courant

$Yref$ est la luminance du macrobloc de référence

(x, y) sont les coordonnées du MB de référence dans la fenêtre de recherche

Les algorithmes basés bloc impose l'hypothèse supplémentaire que les pixels du bloc courant à estimer ont tous le même mouvement de translation. Les dimensions du bloc sont fixées par la norme à 16*16 pixels (ou 8*8 pixels si le mode de prédiction par 4 vecteurs mouvement est sélectionné et 8*16 pour de la vidéo entrelacée) ce qui correspond à un bon compromis entre la granularité des estimations et la complexité de la recherche des vecteurs mouvement.
Le plus simple des algorithmes de recherche par appariement de blocs (BMA : "Block Matching Algorithm") est l'algorithme de recherche exhaustive. L'optimum est sélectionné parmi tous les vecteurs possibles à l'intérieur de la fenêtre de recherche ce qui correspond à $(2d+1)^2$ calculs de distorsion (vecteur mouvement $\in [-d,d]$, d étant la dynamique des vecteurs). Cependant, le nombre élevé d'opérations rend cette technique inadaptée à la plupart des applications temps réel. C'est pourquoi de nombreux algorithmes de recherche dits rapides sont proposés dans la littérature. Ces méthodes rapides peuvent être réparties dans 4 catégories :
- recherches discrètes dans la fenêtre de recherche,
- recherches hiérarchiques,
- recherches avec échappement,
- méthodes hybrides.



W,H : largeur et hauteur de l'image.
L,P : dimensions de la fenêtre de recherche. L = 2d + n, P = 2d + m.
m,n : dimensions des macroblocs (MB). Généralement, m = n = 16.
MV : vecteur mouvement.
→ vecteur mouvement trouvé.

Figure 18: recherche par appariement de blocs

I.3.2 Accélération de la recherche des vecteurs mouvement

I.3.2.a) Méthodes discrètes

Pour ces algorithmes, une hypothèse supplémentaire est faite. La variation de la distorsion dans la fenêtre de recherche est monotone, c'est à dire que la distorsion est supposée croissante dans toutes les directions à partir du minimum global. Ces algorithmes sont itératifs. A chaque itération une direction privilégiée est choisie. L'algorithme se termine lorsqu'une frontière de la fenêtre de recherche est atteinte ou lorsque le vecteur courant satisfait à un certain critère de convergence.

Quelques exemples de réalisations sont présentées : la recherche dite en "3 pas" (3SS [12]), la recherche dite en "4 pas" (4SS [13]), la recherche logarithmique à 2 dimensions (2D-LOG [14]), la recherche orthogonale (OSA [15]), la recherche en croix (CSA [16]) et la recherche selon le gradient de plus forte pente (GD [17]), mais de nombreuses variantes de ces méthodes existent dans la littérature [18][19][20][21][22].

Dans les paragraphes suivants, *d* est la dynamique des vecteurs mouvement (*d* est choisie égale à 7 pour les schémas explicatifs), $\lceil . \rceil$ est l'arrondi à l'entier supérieur, $\lfloor . \rfloor$ est la partie entière. Les algorithmes rapides testent des sous-ensembles de la fenêtre de recherche. Ces sous-ensembles peuvent être de type "carré", "fois" et "plus" (figure 19).

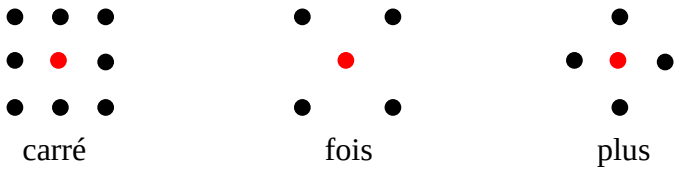


Figure 19: les sous-ensembles de recherche

iii. Recherche en trois pas

L'algorithme 3SS, proposé en 1981 par Koga [12], est basé sur une approche par raffinements successifs avec un pas à décroissance logarithmique. A chaque itération le sous-ensemble choisi est un "carré" de pas égal au pas précédent divisé par 2. Le pas initial est $\lceil d/2 \rceil$. La figure 20 illustre la recherche en 3 pas pour des vecteurs mouvements d'amplitude +/- 7 pixels.

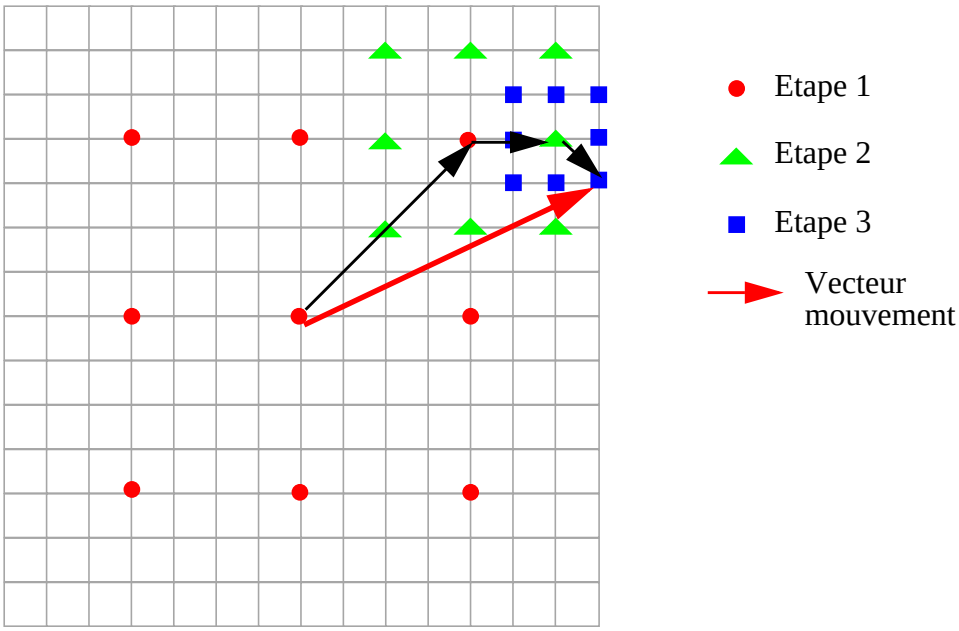


Figure 20: recherche en 3 pas

iv. Recherche en quatre pas

L. M. Po et W.C. Ma, en 1996 [13], exploitent la répartition centrale des vecteurs de mouvement en réduisant le pas initial à $\lceil d/4 \rceil$ au lieu de $\lceil d/2 \rceil$ dans le cas de la 3SS.
Le pas est divisé par 2 lorsque le minimum de distorsion se trouve au centre du sous ensemble de recherche ou lorsque le bord de la fenêtre de recherche est atteint.

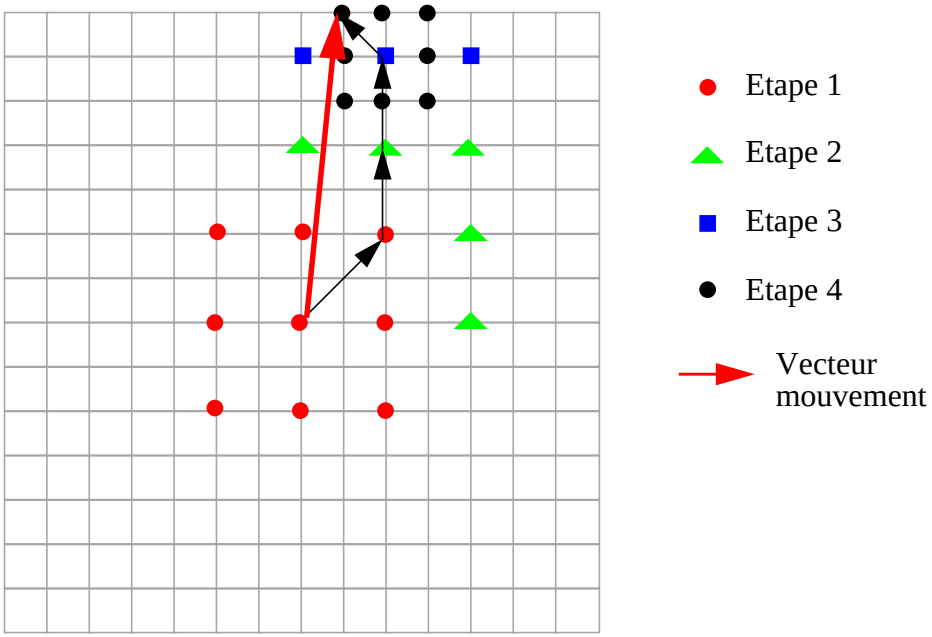


Figure 21: recherche en 4 pas

v. Recherche logarithmique à deux dimensions

Proposé par Jain en 1981 [14], cet algorithme utilise comme sous-ensemble de recherche le "plus". Le pas initial est $\lceil d/4 \rceil$. Ce pas est divisé par 2 lorsque le minimum de distorsion courant est au centre du "plus" ou lorsque le bord de la fenêtre de recherche à été atteint. Pour la dernière itération (pas égal à 1), le sous-ensemble est un "carré". La figure 22 illustre une réalisation.

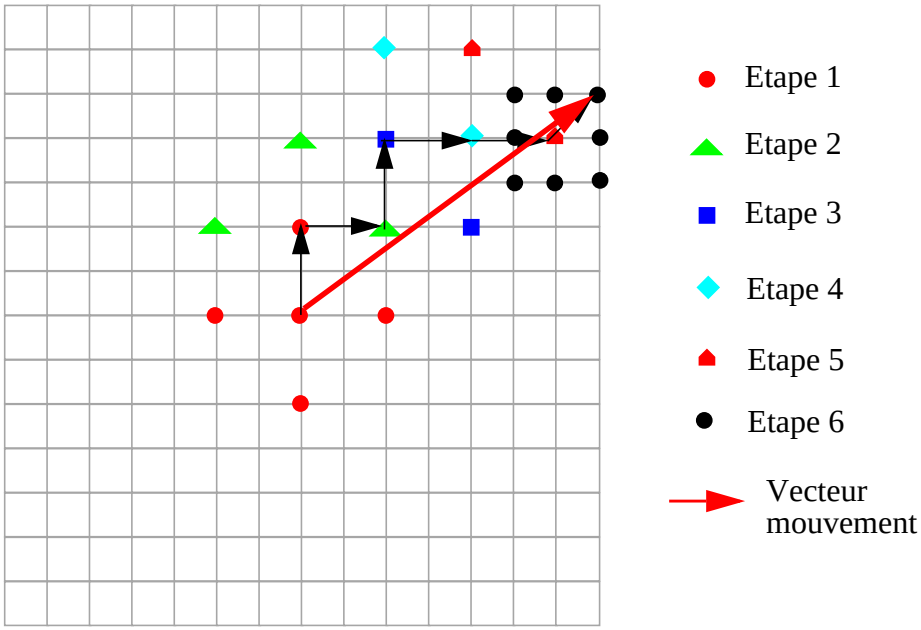


Figure 22: recherche logarithmique à 2 dimensions

vi. Recherche orthogonale

A. Puri [15] en 1987 propose un algorithme qui procède par bonds successifs horizontaux puis verticaux. Les positions testées sont distantes initialement de $\lceil d/2 \rceil$, le pas est divisé par 2 à chaque couple d'itérations (horizontale et verticale).

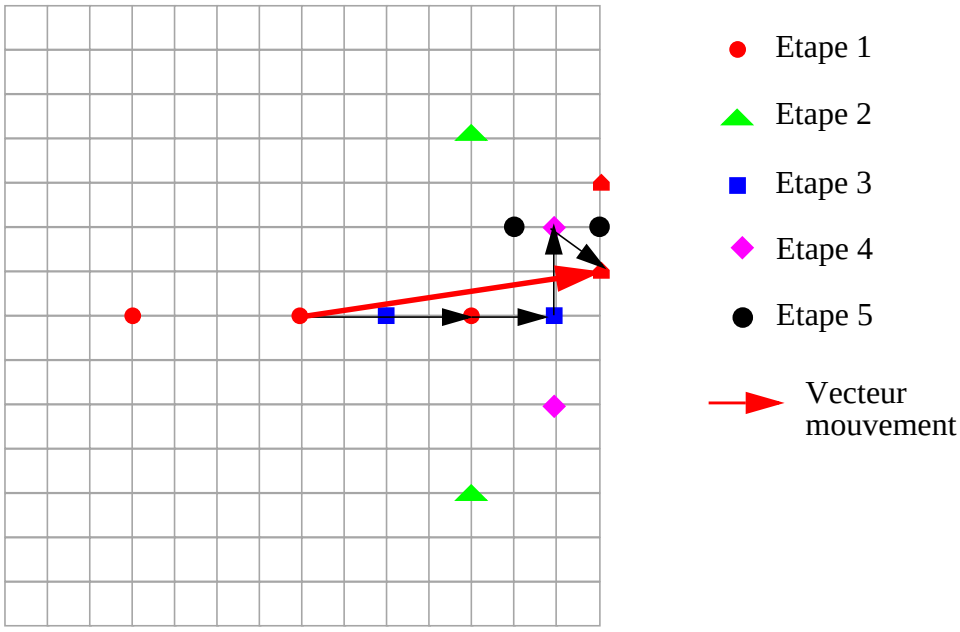


Figure 23: recherche orthogonale

vii. Recherche en croix

Proposée par Ghanbari en 1990 [16], cette recherche a également un pas à décroissance logarithmique et utilise le sous-ensemble "fois". Le pas initial est $\lceil d/2 \rceil$, il est divisé par 2 à chaque itération. Une étape supplémentaire de pas unitaire clôture l'algorithme, un "fois" ou un "plus" est utilisé suivant le résultat de l'étape précédente.
Etape 1 : calcul de la distorsion pour un mouvement nul. Si SAD(0,0) est inférieure à un certain seuil alors le bloc est dit immobile sinon étape 2.
Etape 2 : initialisation de la distorsion minimale à SAD(0,0) et du pas à $\lceil d/2 \rceil$.
Etape 3 : initialisation du bloc de référence courant : le sous-ensemble "fois" est centré sur la position courante de distorsion minimale. Recherche du minimum parmi les positions pointées par "fois".
Etape 4 : si pas = 1 alors étape 5 sinon (pas = pas/2) puis étape 3.
Etape 5 : si la dernière étape fournit un minimum à une position du type centre, bas droit ou haut gauche sur le sous-ensemble "fois" alors étape 6 sinon étape 7.
Etape 6 : recherche du minimum parmi les positions pointées par un "plus" de pas égal à 1. Fin de la recherche.
Etape 7 : recherche du minimum parmi les positions pointées pas un "fois" de pas égal à 1. Fin de la recherche.
La figure 24 illustre les deux déroulements possibles de la recherche en croix

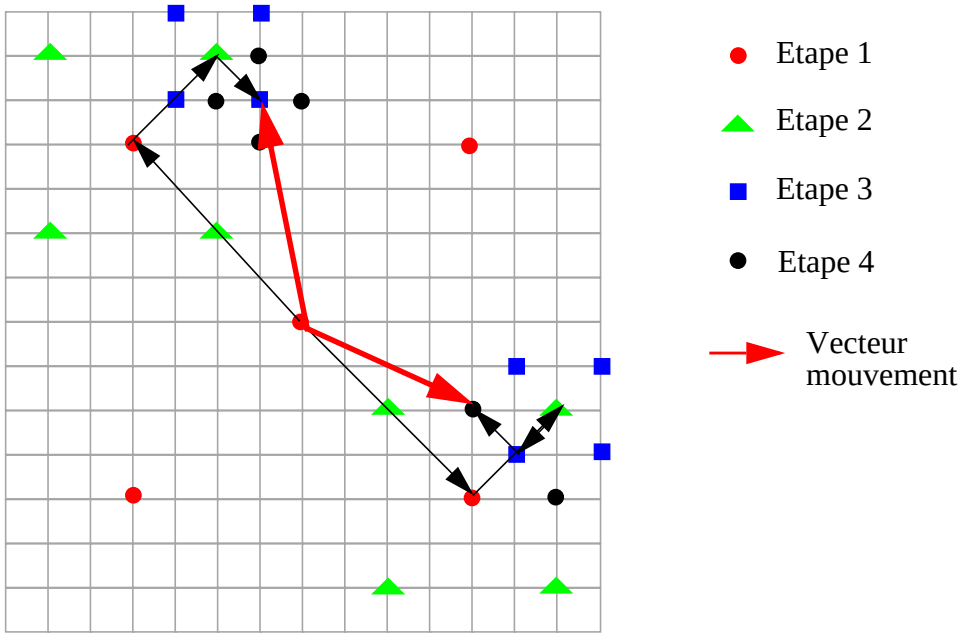
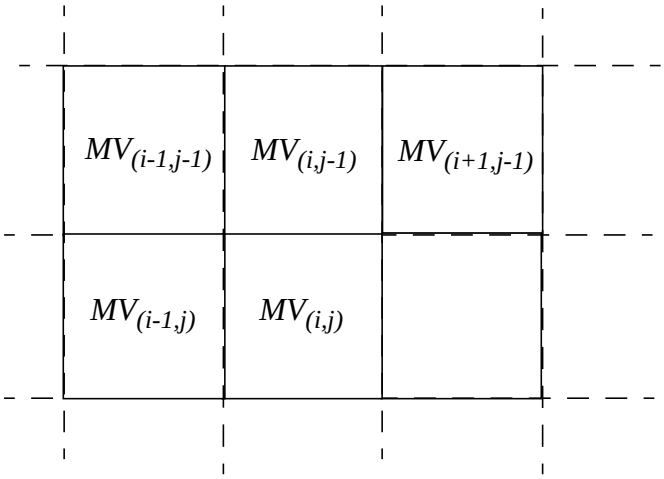


Figure 24: recherche en croix

viii. Recherche selon le gradient

Cette méthode proposée par Liu et Feig [17], exploite la corrélation entre les vecteurs mouvement des MBs voisins. Pour chaque MB de l'image, on effectue une prédiction du vecteur mouvement à partir des vecteurs mouvement des MBs voisins déjà calculés (figure 25). Il s'agit de la même prédiction que pour le mode de compensation de mouvement par blocs recouvrants (OBMC, H.263 Annexe F).



$$MV_{(i,j)} = (2 \cdot MV_{(i-1,j)} + 2 \cdot MV_{(i,j-1)} + MV_{(i-1,j-1)} + MV_{(i+1,j-1)} + 3)/6 \qquad \text{(B.7)}$$

Figure 25: prédiction du vecteur mouvement du MB courant

On calcule ensuite les distorsions des positions décrites par le sous-ensemble "carré" de pas unitaire. Si le minimum de distorsion est au centre du sous-ensemble ou si le bord de la fenêtre de recherche est atteint, alors la recherche est terminée, sinon, on centre le sous-ensemble "carré" sur le nouveau minimum et on itère.
La complexité dans le pire cas est égale à $(2.d+1)^2$ (calcul de SAD sur toutes les positions de la fenêtre de recherche). Cependant, le profil de distribution de la distorsion réalisant ce cas est très peu probable. De plus, dans une image naturelle, les vecteurs mouvement des MBs voisins sont corrélés et on peut supposer que la prédiction du vecteur mouvement du MB courant est relativement bonne. Toutefois, on peut estimer, dans l'hypothèse de monotonie de la distorsion et de l'existence d'un minimum global que le pire cas correspond au parcourt de la diagonale de la fenêtre de recherche, ce qui correspond au calcul de $5.(2d-1)+4$ distorsions.

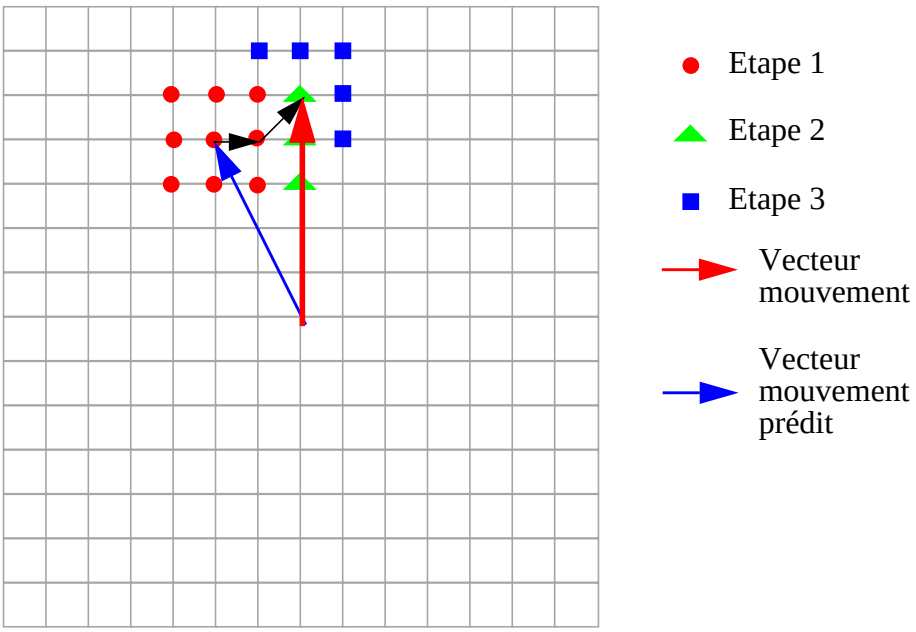


Figure 26: recherche selon le gradient

Le tableau 14 donne les complexités algorithimiques des différentes techniques d'estimation de mouvement présentées.

Tableau 14: Complexité algorithmique pire cas des méthodes discrètes

Algorithme de recherche	Complexité ($\leq$)	complexité pour d = 7 ($\leq$)
Exhaustive	$h \cdot w \cdot (2 \cdot d + 1)^2 \cdot N^2$	$225 \cdot h \cdot w \cdot N^2$
3SS	$h \cdot w \cdot (1 + 8 \cdot \lceil \log_2(d+1) \rceil) \cdot N^2$	$25 \cdot h \cdot w \cdot N^2$
4SS	$h \cdot w \cdot \left(9 + 18 \cdot \left\lceil \log_2 \frac{(d+1)}{4} \right\rceil\right) \cdot N^2$	$27 \cdot h \cdot w \cdot N^2$
2D-LOG	$h \cdot w \cdot (2 + 7 \cdot \lceil \log_2(d+1) \rceil) \cdot N^2$	$23 \cdot h \cdot w \cdot N^2$
OSA	$h \cdot w \cdot (1 + 4 \cdot \lceil \log_2(d+1) \rceil) \cdot N^2$	$13 \cdot h \cdot w \cdot N^2$
CSA	$h \cdot w \cdot (5 + 4 \cdot \lceil \log_2(d+1) \rceil) \cdot N^2$	$17 \cdot h \cdot w \cdot N^2$
GS *	$\dot{h} \cdot w \cdot (5 \cdot (2 \cdot d - 1) + 4) \cdot N^2$	$69 \cdot h \cdot w \cdot N^2$

(*) Comme nous le verrons, la complexité de la méthode du gradient est en réalité beaucoup plus faible lorsque la corrélation des vecteurs mouvements de MBs voisins est forte.
Cette remarque peut être étendue à une majorité de ces complexités car la convergence de certains algorithmes présentés dépend de la nature de l'image.
Nous pouvons donc dire, dans un premier commentaire, que les complexités des différentes techniques d'estimation de mouvement rapides sont du même ordre de grandeur.

I.3.2.b) Méthodes hiérarchiques

L'appariement de bloc hiérarchique également appelé recherche télescopique [23], est une approche multi-résolution travaillant en plusieurs passes sur des MBs de résolutions différentes.
Le point de départ est une image sous-échantillonnée, le vecteur mouvement, à ce niveau, est estimé à partir d'un macrobloc de taille réduite (généralement la recherche s'effectue en un maximum de trois passes, un MB fait donc au minimum 4*4 pixels).
La position pointée par le vecteur mouvement issue de la première étape de l'algorithme est utilisée comme point de départ pour l'estimation dans l'image de résolution supérieure. La fenêtre de recherche autour de cette prédiction peut être réduite du fait que l'étape précédente de l'algorithme fournit déjà une bonne approximation du vecteur mouvement.

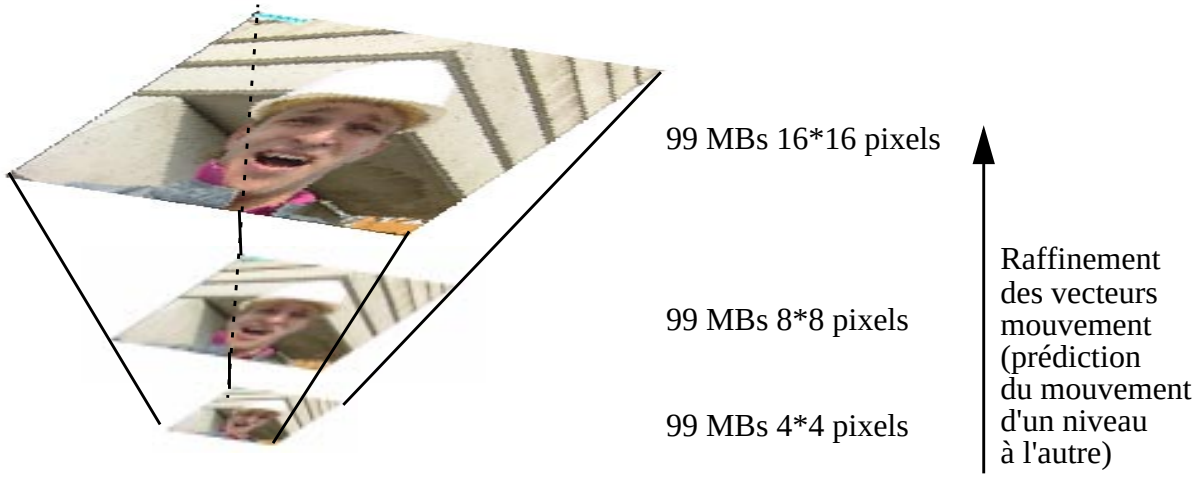


Figure 27: recherche hiérarchique

I.3.2.c) Méthodes hybrides

Différentes techniques hybrides sont proposées dans la littérature [24][25], nous nous contentons d'en citer une en exemple. L'approche consiste à effectuer les calculs de distorsion non plus sur la luminance de l'image mais sur une image des contours des objets contenus dans la scène [26]. Dans ce cas les MBs ne contiennent plus que des valeurs binaires et la somme des valeurs absolues des différences devient une accumulation de "ou exclusif" entre le MB courant et le MB de référence. La réduction de complexité de l'estimation de mouvement obtenue est importante mais la qualité de l'estimation est fortement dépendante de l'algorithme de détection de contours choisi. De plus, cet algorithme peut s'avérer coûteux. L'intérêt de cette approche devient alors discutable.

I.3.2.d) Recherche en spirale avec échappement

Une autre méthode de réduction de la complexité de l'EM est basée sur la constatation suivante: dans les vidéos naturelles, la répartition des vecteurs mouvement peut être vue comme une loi gaussienne centrée sur le vecteur mouvement nul. Un parcours astucieux de la fenêtre de recherche peut être choisi afin d'exploiter cette observation. En effet, un parcours de la fenêtre de recherche en spirale partant du centre de celle-ci réalisé conjointement avec un test d'échappement du calcul de distorsion permet en moyenne de réduire de moitié le temps passé dans l'opérateur de calcul de distorsion (figure 28).

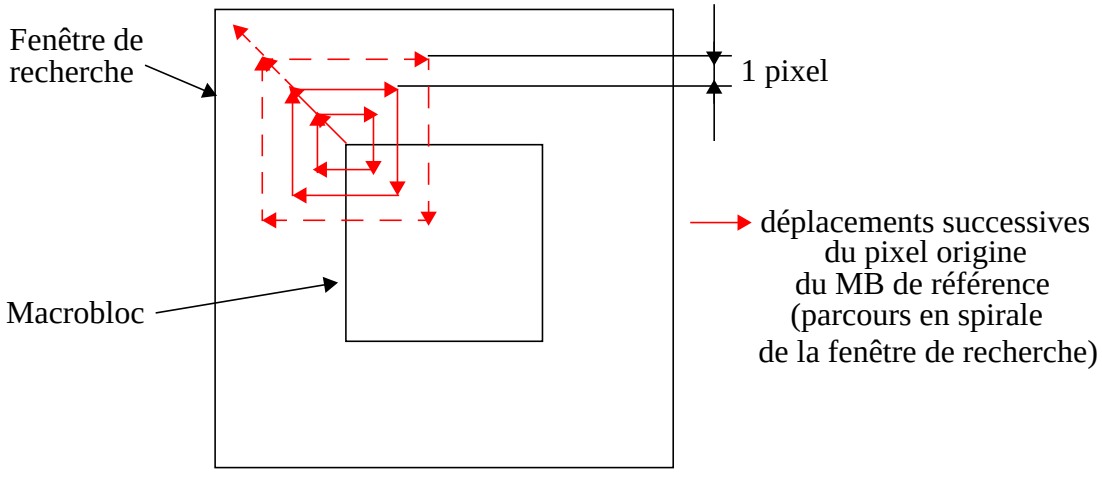


Figure 28: recherche en spirale avec échappement

Le calcul de la distorsion à la position courante est alors divisé en plusieurs calculs de distorsions partielles. A chaque nouvelle valeur de la distorsion partielle, une comparaison avec le minimum de distorsion courant est effectuée. Le choix du nombre de valeurs absolues de différences accumulées dans la distorsion partielle avant un nouveau test est fait de façon à optimiser le nombre moyen de tests avant échappement (en général, 16 ou 32 pixels sont accumulés ce qui correspond à une ou deux lignes d'un MB).

Déroulement d'une recherche en spirale avec test d'échappement dans le calcul de distorsion :

```
{
Minimum_SAD = SAD_Macrobloc(MB_référence(mouvement nul), MB_courant, INTMAX)

Pour MB_référence parcourant en spirale la fenêtre de recherche faire :
        SAD = SAD_Macrobloc(MB_référence, MB_courant, Minimum_SAD)
        Si SAD < Minimum_SAD alors :
                Vecteur_mouvement = position courante de MB_référence sur la spirale
                Minimum_SAD = SAD
}
Avec
 SAD_Macrobloc(MB_référence(mouvement nul), MB_courant, INTMAX) :
{
SAD = 0
j = Nombre_de_lignes_dans_un_MB
Tant que j>0 faire :
        Pour i parcourant une ligne de pixels du MB :
                SAD = SAD + Somme(|MB_référence(i,j) - MB_courant(i,j)|)
        Si SAD > Mimum_SAD alors
                MB_référence suivant sur la spirale
}
```

II. Complexité vs qualité

L'étude de complexité est réalisée sur une machine UltraSparc IIi, 440 MHz. Il s'agit d'une architecture Sparcv V9 [27] composée de 9 unités d'exécution (4 unités d'exécutions entières : 2 ALUs, un multiplieur et un diviseur + 8 fenêtres de registre et 4 registres globaux ; 3 unités d'exécution flottantes : multiplication, addition, division ; 2 unités d'exécution graphiques : unités adaptées aux traitements de l'image et du signal comme l'exécution d'une valeur absolue de différences en un cycle, additionneur, multiplieur divisibles en bancs de 8 ou 16 bits). Le jeu d'instruction est sur 64 bits. Nous ne chercherons pas dans notre étude à exploiter toute la potentialité de l'architecture V9 de Sparc et de son jeu d'instruction. Notre but n'est pas d'utiliser l'unité graphique qui est bien adaptée aux mesures de distorsion pour l'estimation de mouvement mais d'utiliser des instructions non spécifiques à l'architecture Sparc afin d'obtenir des valeurs réalistes et interprétables pour la complexité du codeur. Nous distinguerons dans notre étude deux types de complexité, celle correspondant aux nombre d'instructions (MIPS : Million d'Instructions Par Seconde) nécessaire pour le codage sur un processeur programmable généraliste ou RISC et celle, plus proche d'une complexité matérielle ou sur DSP correspondant respectivement à des MOPS (Million d'Opérations Par Seconde) ou à des MIPS DSP (MMAC/s : Million de Multiplications ACcumulations par seconde).

Les séquences tests utilisées sont représentatives de scènes de visiophonie. Elles correspondent à des scènes soit "statiques" en ce qui concerne les séquences Akiyo, Miss America et Claire, soit "mobiles" pour les séquences Foreman et Carphone.
Toutefois, la séquence Carphone est principalement utilisée pour les tests et les estimations car elle est représentative d'une situation de visiophonie mobile du fait d'un important mouvement du visage et de la présence d'un arrière plan non statique.
Par ailleurs, dans un contexte autre que la visiophonie des séquences sans visage sont utilisées pour tester les performances de codage.
Coatguard : deux bateaux en mouvement, l'un rapide, l'autre lent avec "traveling" de la caméra.
Container : scène comportant peu de mouvement. Déplacement lent d'un navire.
Foot : séquence d'un match de football américain composée de beaucoup d'objets en mouvement rapide.

II.1 Outils de "profilage"

Deux outils de "profilage" ont essentiellement été utilisés : *gprof* et *iprof*.

II.1.1 gprof

L'utilitaire *gprof* exploite les informations fournies par le fichier de trace d'exécution (a.out par défaut) et le fichier de description des dépendances des fonctions (gmon.out par défaut). Pour une compilation avec *cc* de SUN, l'option de compilation "-xpg" doit être activée.
gprof fournit diverses informations sur la structure du programme et sur la charge du CPU (Central Processor Unit). Dans notre cas, il s'agit d'un processeur Ultra Sparc IIi. Les options de compilation "-xchip=ultra", "-xO3" et "-xdepend" ont été retenues. "-xchip=ultra" correspond à la famille du processeur cible, "-xO3" permet une optimisation du code compilé (déroulage de boucles et pipeline logiciel), et "-xdepend" effectue une analyse des boucles suivie d'une optimisation (dépendance des données et restructurations de boucles).
gprof fournit :
- le graphe de dépendance des fonctions (indépendant de l'architecture du processeur),
- la répartition du temps CPU sur les branches de l'arbre de dépendance, en distinguant les temps

propres des fonctions et les temps cumulés (fonction mère + fonctions filles),
- un classement des fonctions selon leur temps CPU.
Les deux dernières informations sont dépendantes du processeur, mais donnent une idée de la répartition de la complexité (en terme de charge du processeur) selon les fonctions du programme. Ceci permet de dégager les fonctions qui sont a priori les plus coûteuses.

II.1.2 iprof

Iprof est un utilitaire de l'université de Munich [28] permettant d'obtenir des statistiques sur les instructions machines appelées par un programme C (ou C++) ainsi que sur la bande passante. Les architectures supportées par la version 0.4.1 utilisée pour notre étude sont l'architecture Sparc et l'architecture X86. Le programme dont on veut obtenir les statistiques doit être compilé avec le compilateur GNU *gcc* (ou *g++*) et l'option de compilation "-a" afin d'obtenir la trace d'exécution du programme (bb.out par défaut) exploitée ensuite par iprof.
L'utilitaire *iprof* fournit :
- une liste des instructions appelées avec le nombre d'appels,
- un groupement des instructions selon les catégories suivantes :
    - arithmétique : addition, soustraction, multiplication, division, décalage ("shift"), incrément/décrément,
    - branchement, test, comparaison,
    - lecture et écriture ("load/store"),
    - conversion de type,
    - instructions pour les flottants,
    - autres instructions,
- une répartition de la bande passante en fonction des écritures et des lectures. La bande passante est calculée en Mbyte, elle est basée sur le nombre de lecture/écriture en mémoire ainsi que sur la longueur des mots (8 bits, 16 bits, 32 bits, 64 bits).

Dans les tableaux de complexité fournis par la suite, nous avons choisi de grouper les instructions comme suit :
- arithmétique : addition, soustraction, multiplication, division, "shift", comparaison.
Les incréments ont été enlevés de ce groupe car ils correspondent essentiellement aux opérations sur les indices de boucles. En revanche, nous avons ajouté l'instruction de comparaison principalement due au calcul de la valeur absolue. Ce nombre d'instructions arithmétiques donne une bonne idée de la complexité en nombre d'opérations élémentaires nécessaires à l'exécution du programme et fournit donc une approximation des ressources nécessaires sur un DSP.
- branchement, test,
- incrément/décrément,
- "load/store",
- conversion de type,
- autre instructions.
Remarque : notre implémentation ne fait plus appel à des instructions de l'unité flottante.

De plus, pour les études qui vont suivre nous avons retenu les options de compilation suivantes :
- "-mv8", pour l'utilisation du jeu d'instructions de l'architecture v8 du processeur Sparc (multiplications et divisions entières),
- "-O3", pour le déroulage des boucles, l'optimisation du pipeline logiciel et l'insertion du code de routines dans le corps du programme ("inlining").

II.2 Mesure de la qualité

La qualité visuelle du codage peut être mesurée soit par le calcul du rapport signal à bruit de l'image reconstruite par rapport à l'image d'origine, soit par des tests de visualisation indiquant la qualité subjective de l'image.
Pour cette dernière, on distingue habituellement 5 niveaux de qualité : parfait, bon, acceptable, médiocre, mauvais.

Le rapport signal à bruit indique quant à lui le degré de ressemblance entre l'image d'origine et l'image codée reconstruite. Etant généralement calculé sur toute l'image, il est représentatif de la qualité moyenne mais ne rend pas compte des défauts locaux. Par exemple, dans le cas du codage d'une séquence vidéo constituée d'un objet en mouvement devant un arrière plan fixe, une valeur élevée du rapport signal à bruit ne signifie pas obligatoirement que l'image reconstruite soit de bonne qualité visuelle. Ce point est illustré sur les séquences Claire et Miss America de la figure 29 a) et b). Inversement, lorsque l'arrière plan comporte beaucoup de mouvement, un rapport signal à bruit a priori moins bon peut correspondre à une qualité subjective acceptable (figure 29 c) et d))

a) SNR en dB Y: 33,19
Cr : 34,7
Cb : 37,95

c) SNR en dB Y: 31,72
Cr : 37,53
Cb : 36,75

b) SNR en dB Y: 34,16
Cr : 36,44
Cb : 33,09

d) SNR en dB Y: 31,96
Cr : 37,6
Cb : 36,96

Figure 29: SNR et qualité subjective

Sur les images a) et b) de la figure 29, l'arrière plan fixe rehausse artificiellement le SNR. Dans notre cas où la qualité de la vidéo est fortement liée à la qualité obtenue sur l'objet d'intérêt, la mesure du rapport signal à bruit moyen sur l'image ne peut suffire à l'évaluation de la qualité perçue. Il faut pouvoir mesurer le SNR sur le visage seul conjointement à une évaluation de la qualité subjective. Ces mesures sont mises en place pour les études réalisées dans le chapitre C.

II.3 Résultats

II.3.1 Options de codage normalisées

II.3.1.a) Complexité

Conventions pour les profils :
H.263 L1 : codage intra avancé, filtrage anti effet bloc dans la boucle de codage, quantification modifiée.
H.263 L2 : vecteur mouvement non restreint, structure en "slice".
H.263 L3 : prédiction avancée, codage en image PB améliorée, codage à longueur variable inter alternatif.
MPEG-4 SP : prédiction avancée, vecteur mouvement non restreint, codage intra avancé, codage à longueur variable inter alternatif.
UMV : H.263+ utilisant le mode de vecteurs mouvements non restreints (H.263 Annexe D.2).
AP : H.263+ utilisant le mode de prédiction avancé (H.263 Annexe F et Annexe D.1)

L'ensemble des tests utilisent une recherche exhaustive des vecteurs mouvements en spiral (paragraphe I.3.2.d).
Les complexités sont classées selon leur type. Une complexité totale est donnée ainsi qu'une complexité arithmétique. Cette dernière est égale à la somme des instructions de multiplication, d'addition, de division, de soustraction, de décalage et de comparaison. Elle représente la complexité équivalente en million d'opérations.

Tableau 15: Complexité en fonction des outils de codage

Profil / Complexité (MI/image)		Sans option	UMV	AP	H.263 L1	H.263 L2	H.263 L3	MP4SP
Million Instructions/image		**77.85**	**204.69**	**95.47**	**95.86**	**206.06**	**87.13**	**227.8**
Complexité aritmétique*		***41.06***	***105.79***	***50.1***	***49.5***	***106.51***	***45.68***	***115.65***
	Add*	14.97	38.46	18.51	18.21	38.74	16.82	42.03
	Sub*	7.25	24.25	8.38	8.52	24.54	8.13	24.84
	Mult*	0.81	1.08	1.18	0.88	1.06	0.91	1.42
	Div*	0.05	0.05	0.05	0.07	0.05	0.05	0.06
	Shift*	12.41	22.14	15.7	15.46	22.05	13.55	27.19
	Autres(inc.,...)	0.65	1.81	1.45	1.44	1.81	1.06	4.24
	Cmp* (abs)	5.57	19.81	6.29	6.36	20.07	6.22	20.11
Branch. Test		9.3	21.91	11.27	12.22	21.94	16.19	25.83
Load / Store		22.04	60.88	27.18	26.98	61.39	24.96	67.28
Conversion de type		0.53	0.8	0.55	0.56	0.78	0.45	0.79
Autres instructions		4.28	13.51	4.92	5.15	13.62	4.79	14.01
Bande passante (MB/image)		**29.77**	**66.24**	**35.97**	**34.72**	**66.45**	**32.18**	**72.63**
	Lecture	25.7	61.7	31.34	30.11	61.97	28.16	66.84
	Ecriture	4.07	4.54	4.63	4.61	4.48	4.02	5.79

Les complexités sont données pour 380 images de la séquence Carphone au format QCIF (30 Hz) codées au débit cible de 40 kbit/s et à la fréquence image cible de 15 Hz. Dans tous les modes une image I suivie de 185 images P sont codées.
(*) Instructions comptabilisées pour la complexité arithmétique.

Commentaires :

• La première remarque porte sur la complexité de base du codeur H.263 (sans option), qui est de 1.15 GIPS pour le codage QCIF@15 Hz. Cette valeur est trop élevée au regard des processeurs RISC embarqués dont la puissance ne dépasse guère les 200 MIPS (ARM9TDMI : 220 MIPS@200 MHz).
Les limitations sont de deux ordres : la puissance de calcul (environ 600 MOPS) et la bande passante (450 MByte/s).
La première semble être dépassable aujourd'hui sur un DSP mais en accaparant l'essentiel des ressources (StarCore : 480 MMACS@120 MHz). La deuxième directement liée aux accès mémoire en lecture et écriture, et à la dynamique des données conduit à une forte activité sur les bus de données,et provoque pour les technologies actuelles une forte dissipation d'énergie (consommation dominée par la dissipation d'énergie des communications). La figure 30 illustre l'importance des accès mémoire pour le codage vidéo (environ 1/3 des instructions)
.

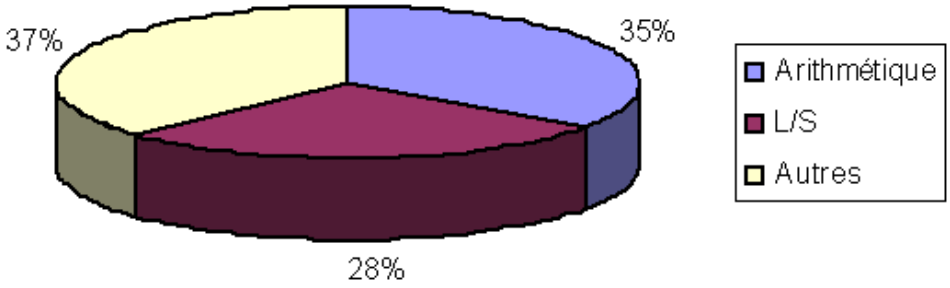


Figure 30: importance des instructions Lecture/Ecriture (L/S)

• La deuxième remarque concerne la forte variation de la complexité du codeur suivant le profil de codage retenu. Celle-ci varie de 1.3 GIPS pour la recommandation L.3 à 3.42 GIPS pour MP4SP. Cette forte différence est principalement due au mode sélectionné pour l'estimation de mouvement. En effet, dés lors qu'on sélectionne le mode de vecteurs mouvement non restreints (UMV, H.263 L2, MP4SP) le nombre de distorsions à calculer augmente considérablement ce qui entraîne des accumulations, des soustractions et des comparaisons supplémentaires (prés de 3 fois plus). Les autres profils ne conduisent qu'à un surcoût de complexité d'environ 20 % qui reste raisonnable si l'apport en qualité est remarquable (ce que nous verrons un peu plus loin dans ce paragraphe).
L'autre aspect à prendre en considération est la bande passante. Elle augmente également fortement avec le mode UMV du fait de l'utilisation d'une fenêtre de recherche plus grande (lecture de luminances de pixels supplémentaires pour la recherche des vecteurs mouvement) et passe de 480 MBytes/s pour H.263 L.3 à 1.1 GBytes/s pour MP4SP.

Tableau 16: répartition des temps CPU en fonction des outils de codage

	EM/CM	DCT/IDCT	Q/IQ	VLC	Autres
sans option	80.84	11.18	3.14	3.28	1.56
UMV	93.81	3.82	1	1.2	0.17
AP	84.83	8.76	2.45	3.13	0.83
H.263 L1	83.54	8.89	2.96	3	1.61
H.263 L2	93.87	3.41	1.17	1.33	0.22
H.263 L3	85.22	8.76	2.79	2.57	0.66
MP4SP	93.23	2.8	1.97	1.65	0.35

Les estimations du taux d'occupation du CPU en fonction des différents traitements composant le codage vidéo (tableau 16) montrent très clairement l'importance de l'estimation de mouvement par rapport aux autres traitements.
L'analyse plus fine du bloc de traitement effectuant l'estimation/compensation de mouvement (figure 31) indique que les ressources sont principalement utilisées par l'opérateur de calcul de distorsion, à savoir, la somme des valeurs absolues des différences : 68% du temps CPU total pour le codeur H.263 sans option et jusqu'à 84% pour le mode UMV (car comme nous l'avons souligné précédemment cette option autorise une recherche élargie de vecteurs mouvement entraînant un plus grand nombre de calculs de distorsions).

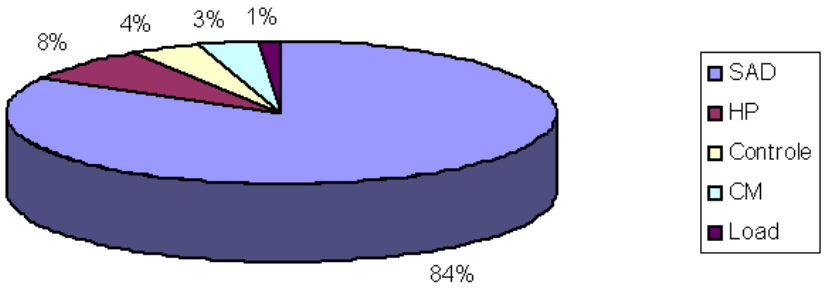


Figure 31: répartition de la complexité dans l'EM

Ces résultats permettent de faire une première hypothèse sur la répartition "matérielle" et "logicielle" des traitements : blocs susceptibles d'être implantés sous forme logicielle (choix d'une architecture de processeur programmable) et blocs qui doivent être totalement ou partiellement réalisés sous forme matérielle (circuit câblé, IP "matériel").
L'estimation de mouvement est un bloc candidat à une réalisation matérielle. En effet, ce traitement est le plus coûteux et, de plus, il se prête bien à une réalisation matérielle du fait de sa bonne régularité et de la possibilité de paralléliser les calculs. Sans aller jusqu'à une réalisation d'un estimateur de mouvement entièrement câblé qui serait coûteuse et empêcherait toute modification de la technique d'estimation de mouvement, il est intéressant de réaliser l'opérateur de calcul de la somme des valeurs absolues des différences en "matériel" [33]. En effet, cet opérateur est générique à toutes les techniques d'estimation de mouvement basées blocs et une implémen-

tation en matériel n'enlève en rien de la flexibilité. De plus, le calcul de distorsion est parfaitement régulier et offre la possibilité de haut degré de parallélisation.
D'autre part, la transformée en cosinus discrète et son inverse peuvent faire l'objet d'une implémentation matérielle mais sont également bien adaptées à une implantation sur un coeur de DSP : la forme directe avec sa bonne régularité est adaptée à une architecture MAC, pour la version rapide utilisant une structure en papillon, l'utilisation d'une instruction spécifique pour les cellules papillons implantée sur certain DSP (par exemple le Blackfin ADSP d'Analog Devices [55]) permet une implémentation efficace de la DCT.
Par ailleurs, la quantification et le codage entropique ne requièrent pas beaucoup de ressources (moins de 10% du CPU) et sont a priori réalisables en "logiciel". Cependant, le codage à longueur variable effectue des manipulations de bits qui ne se prêtent pas à une implantation sur DSP mais plutôt à une implantation sur processeur RISC.
Enfin, le reste du codage correspond au contrôle du système, à la gestion des tâches et au contrôle du débit et peut être fait sur un coeur de processeur RISC.

II.3.1.b) Qualité du codage

L'étude de faisabilité d'un service de visiophonie mobile doit tenir compte, bien entendu, des ressources nécessaires à la réalisation des traitements mais également de la qualité de service et, par conséquent, de la qualité du rendu visuel. C'est ce que nous proposons de faire ci-après.

Tableau 17: qualité (dB) en fonction des profils de codage

	Nombre d'images codées	Y	Cr	Cb
sans option	186	30.54	36.62	35.8
UMV	186	30.68	37.1	36.31
AP	186	30.73	36.67	35.82
H.263 L1	186	30.4	37.79	37.31
H.263 L2	186	30.47	37.03	36.14
H.263 L3	186	31.25	38.13	37.17
MP4SP	186	30.47	36.9	36.04

D'autres mesures de qualité pour des séquences typiques de visiophonie ou non sont fournies en Annexe B.2.

II.3.1.c) Analyse des résultats

- *Le mode UMV*

Le mode UMV présent dans les configurations L2 et MP4SP conduit à une complexité 2.8 fois plus importante en moyenne que le profil sans option. Ce mode n'améliore pas grandement la qualité des estimations dans le cas de scènes de visiophonie "mobile" (en moyenne +0.15 dB en luminance et +0.56 dB en chrominance) et n'apporte rien dans le cas de scènes visiophoniques fixes.
De plus, l'utilisation d'une structure en slice dans ces profils nécessite des bits supplémentaires pour coder la séparation des paquets, ce qui induit une baisse de qualité à débit constant. Un

compromis doit dans ce cas être trouvé entre la taille des slices, le taux d'erreur du canal et la qualité.
Cette option est cependant intéressante pour les vidéos composées de mouvements de forte amplitude telle que la séquence "Football". En effet, on obtient un nombre d'images codées plus grand pour le mode UMV utilisé seul (cf. annexe B.2).

- *Le profil L.3*

Il s'agit du profil le mieux adapté en terme de qualité de codage d'une scène de visiophonie

• Intérêt

Une amélioration significative de la qualité (voir tableau 18) est obtenue sur la luminance et les chrominances pour une augmentation de la complexité de seulement 12 % par rapport à la version sans option.

Tableau 18: Gain en qualité par rapport au codage sans option

SNR \ Séquences	Carphone, Foreman	Akiyo, Claire, Miss America	Coastguard, Container
Y	+ 0.92 dB	+ 1.24 dB	+ 1.38 dB
Cr, Cb	+ 1.42 dB	+ 1.34 dB	+ 1.66 dB

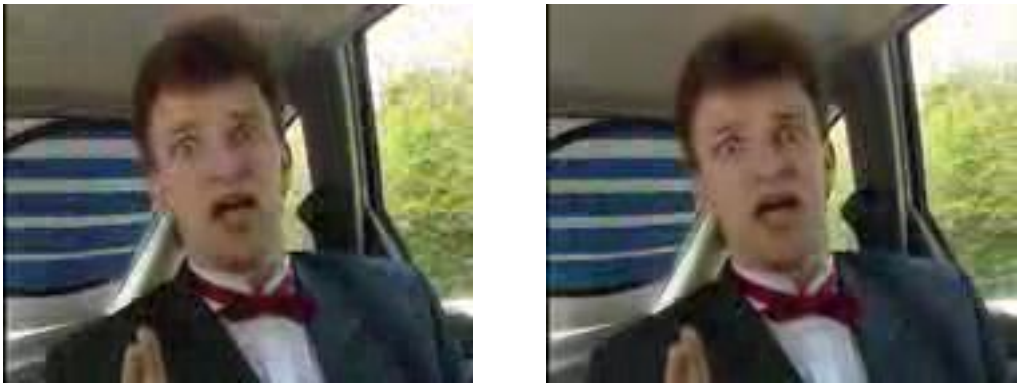
L'amélioration de la qualité peut être attribuée à différentes options :
- l'utilisation du mode PB permet de réduire considérablement le nombre de bits nécessaires au codage de deux images consécutives ce qui équivaut à une augmentation de la qualité à débit constant,
- l'influence de la prédiction avancée (mode AP) est étudiée séparément,
- le mode de codage inter alternatif est également implanté. Les tables de codes du mode Intra sont utilisées pour coder les MBs Inter contenant beaucoup d'information ce qui permet de sauver des bits. Toutefois, bien que permettant une amélioration de la qualité, cette option n'est pas à l'origine des gains significatifs obtenus. Une amélioration de plus de 1 dB est obtenue sur des séquences visiophoniques n'exploitant pas cette option, en effet, les faibles différences inter images sur les séquences Akiyo, Miss America et Claire, ne justifient pas l'emploi des tables de codes Intra.

• Limitations

Cependant, du fait de l'utilisation du mode PB amélioré (codage conjoint d'images P et B) un ré-ordonnancement de la vidéo doit être effectué au décodage ce qui introduit un retard supplémentaire (d'au moins une image). La latence augmente donc de 70 ms pour un codage à 15 Hz ce qui entraîne un perte d'interactivité importante.
De plus, le codage en mode PB nécessite de mémoriser une image de référence supplémentaire ce qui pose un problème de ressource mémoire.
Enfin, une fréquence de 15 Hz en mode PB implique une fréquence de 7.5 Hz sur les images P (utilisées comme images de référence pour la compensation de mouvement) ce qui n'est pas suffisant pour garantir un bon fonctionnement du codeur dans des conditions de mouvements extrêmes comme pour la séquence "Football". Pour cette séquence, le profil L.3 conduit à une perte d'images supérieure de 50 % au cas sans option.
Ainsi, il est plus intéressant de retenir seulement le mode AP de ce profil, en rejetant le mode PB. C'est pourquoi, nous traitons séparément ce cas.

- *Le mode AP*

Aux vues de ces premiers résultats, il semble donc intéressant d'analyser plus précisément les performances du mode AP. En effet, ce mode fait appel à la technique de compensation de mouvement par blocs recouvrants (OBMC) qui conduit au filtrage des contours des blocs 8*8 et ainsi réduit les effets blocs (défaut le plus visible sur le codage à très bas débit). L'augmentation du SNR n'est pas significative : +0.3 dB en moyenne pour la luminance et les chrominances. Cependant, le mode AP entraîne en général une amélioration de la qualité subjective du fait du lissage de la luminance comme l'illustre la figure 32 sur la séquence Carphone.



a) b)

Figure 32: Qualité subjective sur l'image 130 de la séquence Carphone. a) FS, b) AP

- *Le profil L.1*

La recommandation H.263 L.1, dont la complexité est supérieure de 23 % à la version sans options, permet d'obtenir, grâce au filtre dans la boucle de codage, un aspect plus lissé de l'image similaire au mode AP. A noter que l'utilisation du mode de quantification modifiée implique l'utilisation de tables de quantification différentes pour la luminance et les chrominances et entraîne le rehaussement du SNR des chrominances : + 1.51 dB pour les séquences Carphone et Foreman. Cette recommandation conduit à une meilleure qualité subjective de l'image.

II.3.2 Comparaison des performances en fonction de l'EM

Dans l'étude qui suit, aucune option d'amélioration du codage n'est utilisée afin de s'affranchir de l'influence de ces modes et de pouvoir ainsi analyser l'influence des différentes techniques d'estimation de mouvement. L'étude de complexité est réalisée sur la séquence Carphone codée à 15 Hz pour un débit de 40 kbit/s.

Tableau 19: Complexité en fonction de l'EM

Profil / Complexité (MI/image)		FS	FS spirale	3SS	4SS	GDS	HS	2DLOGS	OS	CS
Million Instructions/image		**158.25**	**77.45**	**31.63**	**32.33**	**31.69**	**51.16**	**31.94**	**30.9**	**30.67**
complexité arithmétique *		***84.95***	***41.23***	***17.57***	***17.73***	***17.39***	***26.85***	***17.52***	***17.1***	***17.1***
	Add *	32.55	14.9	6.36	6.41	6.28	9.59	6.33	6.18	6.19
	Sub *	20.64	7.25	1.03	0.96	0.89	2.9	0.9	0.87	0.82
	Mult *	0.73	0.73	0.66	0.66	0.66	0.73	0.66	0.66	0.67
	Div *	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
	Shift *	13.74	12.71	9.16	9.38	9.3	12.01	9.36	9.13	9.23
	Autres(inc.,...)	0.55	0.63	0.31	0.51	0.51	0.88	0.51	0.31	0.31
	Cmp * (abs)	17.25	5.6	0.32	0.27	0.21	1.57	0.22	0.19	0.14
Branchement / test		11.94	9.26	4.93	5.11	5.05	8.21	5.08	5.06	4.88
Load / Store		48.9	21.61	7.54	7.73	7.55	12.65	7.61	7.24	7.19
Conversion de type		0.33	0.41	0.34	0.34	0.33	0.41	0.34	0.34	0.35
Autres instructions		11.58	4.31	0.94	0.91	0.87	2.16	0.88	0.86	0.84
Bande passante (MB/image)		**53.63**	**28.09**	**14.85**	**15**	**14.76**	**19.39**	**14.91**	**14.64**	**14.81**
lecture		49.59	24.05	11.19	11.23	11.01	15.64	11.14	10.97	11.09
écriture		4.04	4.04	3.66	3.77	3.75	3.75	3.77	3.67	3.72

La complexité du codage de la séquence Miss America (séquence possédant peu de mouvement) est fournie en annexe B.3.

Remarques sur le tableau 19 :
- les instructions majoritairement appelées pour les recherches exhaustives avec et sans spirale sont les instructions d'addition/soustraction et de valeur absolue (cmp) qui proviennent principalement du calcul de distorsion (somme des valeurs absolues des différences).
- l'utilisation d'une technique de recherche rapide des vecteurs mouvement entraîne une réduction du nombre d'appels à l'opérateur de distorsion. L'instruction 'cmp' étant appelée uniquement par l'opérateur de distorsion, le facteur de réduction obtenu sur cette instruction (40 pour la méthode 3SS) est similaire à la valeur théorique fournie au paragraphe I.3.2.a). Globalement le facteur de réduction pour les méthodes de recherche rapide est de 5 par rapport à une recherche exhaustive.

Tableau 20: Pourcentage du temps CPU en fonction des traitements

	EM/CM	DCT/IDCT	Q/IQ	VLC	Autres
FS	94.22	3.94	0.92	0.86	0.06
FS spirale	84.21	10.97	2.79	1.91	0.12
3SS	48.02	34.22	10.44	7.09	0.23
4SS	48.18	34.52	10.24	6.91	0.15
GDS	47.53	37.47	7.78	6.99	0.23
HS	73.07	18.77	4.6	3.35	0.21
2DLOGS	48.02	33.94	10.81	7	0.23
OS	47.55	35.46	10.29	6.45	0.25
CS	47.07	38.07	8.68	6.02	0.16

Remarque sur le tableau 20 :
- quelque soit l'estimateur de mouvement retenu, le traitement dominant est l'estimation / compensation de mouvement. Toutefois, l'utilisation d'une technique de recherche rapide réduit fortement la part de temps passée dans l'EM (de plus de 80 % du temps CPU pour les recherches exhaustives à environ 50 % pour les méthodes rapides), la transformée en cosinus discrète et son inverse représentent alors environ 1/3 du temps CPU et deviennent critiques pour une réalisation temps réel embarquée. En analysant plus finement le traitement d'estimation / compensation, on s'aperçoit que pour les méthodes de recherche rapide l'estimation de mouvement au demi pixel représente prés de 50 % de l'EM/CM. Une étude plus approfondie sur la recherche au demi pixel est par conséquent utile en vu d'optimisations supplémentaires. La figure 33 donne la répartition des traitements dans le bloc estimation / compensation de mouvement pour une méthode rapide.

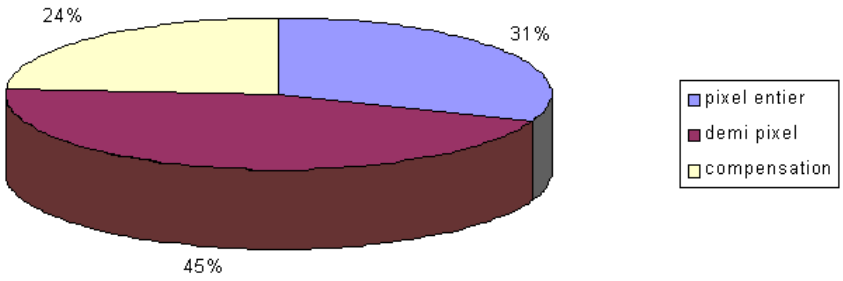


Figure 33: répartition des traitements dans l'EM/CM

Passons maintenant à l'analyse de la qualité du codage en fonction de l'estimateur de mouvement. Le tableau 21 donne la qualité de codage sur la séquence Carphone, d'autres mesures de qualité sont fournies en annexe B.4.

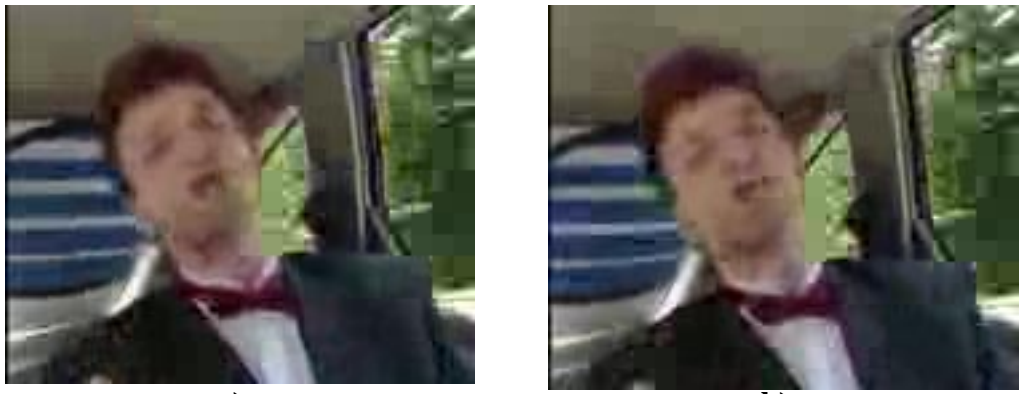
Tableau 21: Qualité (dB) en fonction de l'EM

SNR	FS	FS spirale	3SS	4SS	GDS	HS	2DLOG	OS	CS
Y	30.49	30.49	30.37	30.4	30.41	30.36	30.41	30.28	29.9
Cr	36.53	36.53	36.49	36.53	36.55	36.4	36.55	36.52	36.24
Cb	35.86	35.86	35.84	35.87	35.88	35.71	35.89	36.24	35.61

Interprétation

- *Les méthodes OS et CS*

Ces techniques bien qu'étant les plus intéressante en terme de complexité sont écartées car elles conduisent à des pertes en qualité trop importantes : - 0.21 dB pour la méthode OS et - 0.59 dB pour la méthode CS. De plus, la qualité subjective obtenue sur des séquences comportants beaucoup de mouvement est mauvaise (figure 34).



a) **b)**

Figure 34: image 145 de la séquence Carphone a) OS, b) CS

- *La méthode 3SS*

La méthode 3SS est intéressante du fait de sa bonne régularité permettant une implantation matérielle efficace [56]. Cependant, les vecteurs mouvement sont généralement limités à +/- 7 pixels ce qui est insuffisant. En effet, la fréquence retenue est de 15 Hz et de nombreux mouvements, dans le cas d'une séquence de visiophonie mobile, dépasseront la borne des 7 pixels. Cette borne peut bien entendu être étendue à +/- 15 pixels, mais dans ce cas, le pas entre les positions testées devient important (cf. I.3.2.a) et risque de conduire à de mauvaises prédictions.

- *La méthode HS*

Avec la recherche hiérarchique des vecteurs mouvement la réduction de complexité est seulement d'un facteur 3 par rapport à la recherche exhaustive sans test d'échappement. Cependant, cette technique peut être couplée à une autre méthode de recherche rapide afin de réduire sa complexité. Dans ce cas, la recherche des vecteurs mouvements bénéficie des apports venant des deux méthodes pour une performance accrue.

Les trois autres méthodes (GDS, 4SS et 2DLOGS) conduisent à des performances équivalentes. On obtient une réduction d'un facteur 5 de la complexité et une perte négligeable en qualité (0 dB sur les chrominances, - 0.08 dB sur la luminance). La perte de qualité par rapport à "l'optimum" utilisant une recherche exhaustive n'est pas très importante et ce quelque soit le type de mouvement. Ceci est principalement due aux conditions d'expérimentation et plus particulièrement aux faibles débits visés. En effet, comme l'illustre la figure 35 sur la séquence Foreman, la qualité du codage à faible débit n'est plus dominée par la qualité des estimations mais par la stratégie d'allocation des bits.
Nous allons toutefois faire quelques remarques sur les particularités de chacune ces méthodes.

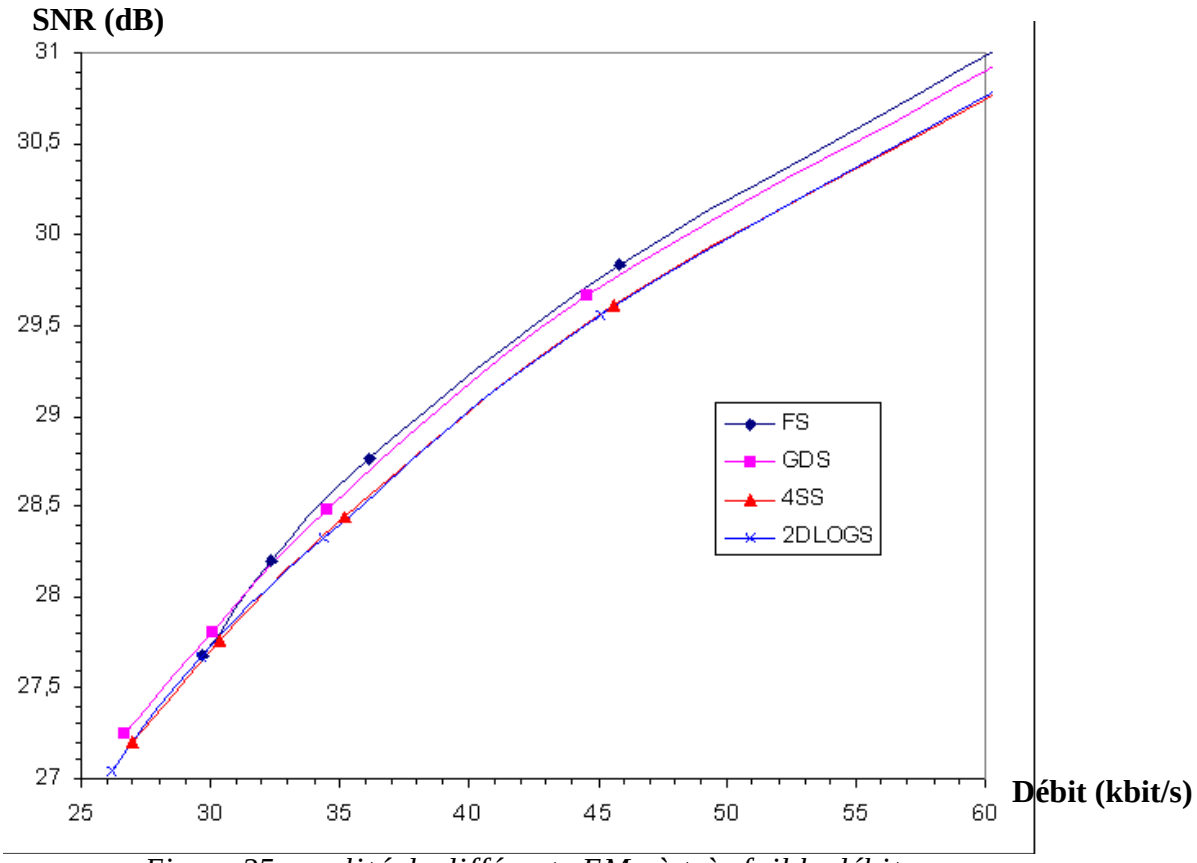


Figure 35: qualité de différents EMs à très faible débit

- *La méthode GDS*

Cette méthode, une fois la prédiction du vecteur mouvement du MB courant réalisée, recherche à une distance de 1 pixels autour de la position prédite le MB le plus ressemblant au MB à coder. La proximité des positions testées fait que les MBs pris successivement comme référence sont fortement corrélés. Le test d'échappement est par conséquent peu efficace (réduction d'un facteur 1.2) du fait de l'intervention plus tardive de l'échappement (cf. figure 37 et 38).

- *La méthode 4SS*

Cette méthode effectue un seuillage de la valeur de distorsion pour un mouvement nul qui s'avère très efficace dans le cas d'une scène vidéo possédant peu de mouvement, comme par exemple, la séquence Miss America (cf. figure 38). Ainsi, la complexité relative de la méthode 4SS passe de 2.2 pour la séquence Carphone à 1 pour la séquence Miss America alors que les complexités relatives des autres méthodes restent stables : 3 pour 3SS, 1.45 pour GD, 1.6 pour 2DLOG (sans test d'échappement).

- *La méthode 2DLOGS*

Cette méthode semble a priori aussi intéressante que les méthodes GDS et 4SS, cependant, la technique de recherche est moins régulière du fait de la dernière étape qui teste 8 positions contre 4 pour les étapes précédentes.

Enfin, nous terminons cette étude par une analyse de l'influence du test d'échappement sur la complexité de l'estimation de mouvement.

- *Influence du test d'échappement*

Le gain obtenu par l'utilisation de ce test est surtout efficace avec une recherche exhaustive des vecteurs mouvement : réduction d'un facteur 3 du nombre de valeurs absolues de différence calculées. Le graphe de la figures 36 montre l'influence du test d'échappement, la valeur 100 correspond au pire cas qui est la recherche exhaustive sans test d'échappement.

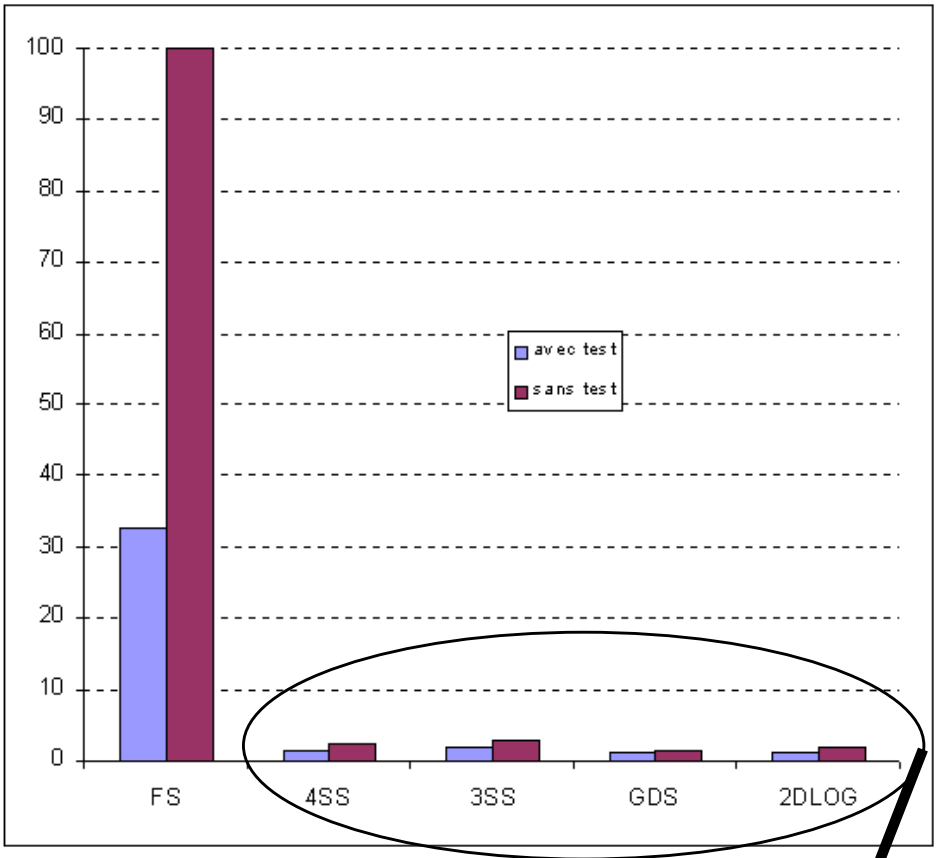


Figure 36: Influence du test d'échappement [1]

Figure 37

1. nombre relatif d'accumulations dans le registre mémorisant la distorsion d'un MB

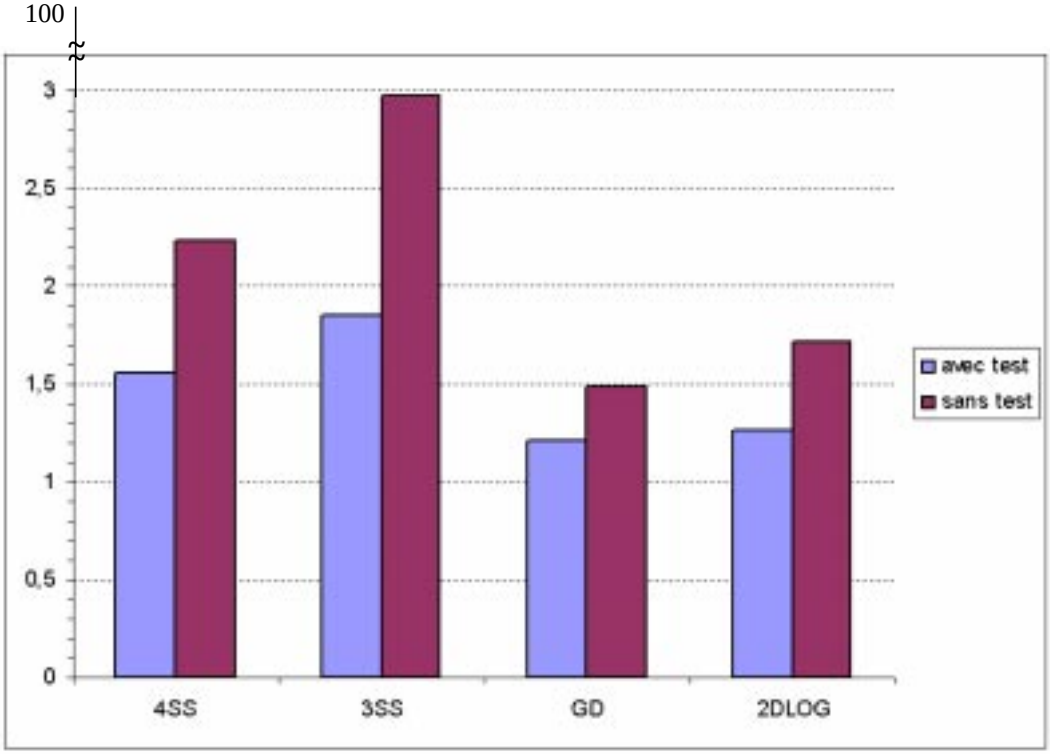


Figure 37: comparaison des méthodes rapides (avec et sans test d'échappement) séquence Carphone[1]

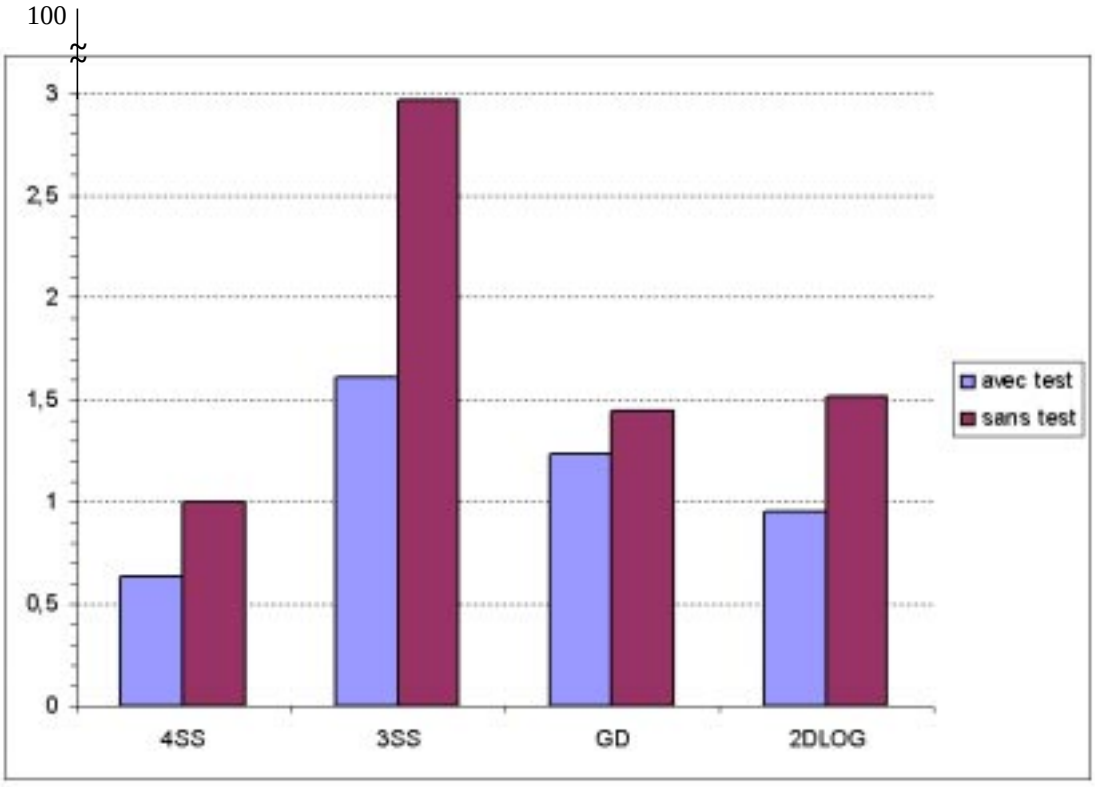


Figure 38: comparaison des méthodes rapides (avec et sans test d'échappement) séquence Miss America[1]

1. nombre relatif d'accumulations dans le registre mémorisant la distorsion d'un MB

Le test d'échappement est intéressant pour les méthodes testant des positions éloignées, et donc généralement peu corrélées comme la méthode 3SS pour laquelle un facteur 2 est obtenu.

Les graphes ci-dessus révèlent également que les complexités algorithmiques théoriques (fondées sur le pire cas) fournies au paragraphe I.3.2.a. ne permettent pas de conclure sur la complexité réelle des méthodes. Une comparaison objective doit être réalisée sur des vidéos naturelles du fait que certaines méthodes sont basées sur la nature des données traitées. En effet, par exemple, pour la méthode 4SS la valeur théorique de la complexité pour une fenêtre de recherche de +/- 15 pixels est 21 fois plus faible que la complexité d'une recherche exhaustive alors qu'expérimentalement, nous obtenons une complexité 45 fois plus faible que la méthode FS (et jusqu'a 100 fois pour une séquence visiophonique en environnement fixe).

II.4 Conclusions

Nous avons vu au cour de cette première phase de l'étude de faisabilité que le choix d'un profil de codage pour l'application de visiophonie mobile est fait en fonction des ressources en terme de complexité et de débit, et également de la qualité de service souhaitée.
Au vu de ces premiers résultats, les options normalisées de codage intéressantes sont :
- le mode de prédiction avancé (AP) ou le filtrage dans la boucle de codage afin de réduire les effets blocs souvent présent dans la gamme de débits visée,
- les options de résistance aux erreurs (découpage en 'slice', partitionnement en groupe de données indépendantes, codage à longueur variable réversible,…) bien que n'ayant pas été étudiées dans le cadre de cette thèse devront être pris en compte lors de la réalisation finale du fait du fort taux d'erreurs sur les réseaux mobiles.

Par ailleurs, afin de réduire la complexité de façon significative, une technique rapide de recherche des vecteurs mouvement doit être retenue. Nous en préconisons deux :
- la méthode en "4 pas" car elle est adaptée à la nature de la vidéo (peu ou beaucoup de mouvement) et conduit ainsi à une complexité moyenne faible. De plus, la qualité de ses prédictions est satisfaisante.
- la méthode du gradient conduit à une complexité moyenne faible en faisant l'hypothèse de la corrélation des mouvements de deux MBs voisins et fournit une qualité très proche de celle obtenue avec une recherche exhaustive.

Enfin, cette étude algorithmique nous permet d'orienter nos choix de partitionnement "matériel / logiciel" de la partie III. En effet, le bloc le plus critique en terme de temps CPU a été révélé. Il s'agit de l'estimation de mouvement et plus particulièrement de l'opérateur de calcul de distorsion. L'orientation retenue pour l'implémentation du codeur est donc la réalisation en matériel de l'opérateur de distorsion, et une implantation sous forme logicielle de la stratégie d'EM conférant ainsi la flexibilité nécessaire. La réalisation sous forme matérielle ou sous forme logicielle des autres blocs de traitement est, a priori, moins évidente et doit être étudiée lors de l'exploration d'architectures.

III. Architecture système

Nous allons, dans cette partie, mettre en place notre méthodologie d'exploration d'architecture. Celle-ci se situe au niveau système et répond à l'évolution de la conception des circuits VLSI.
En effet, l'augmentation incessante de la complexité des systèmes embarqués et l'apparition de systèmes complets intégrés sur une même puce (SoC) ont entraîné ces dernières années un renouvellement de la méthodologie de conception. Historiquement, les étapes de conception, d'implantation et de validation matérielles et logicielles étaient concurrentes et se déroulaient séquentiellement. Aujourd'hui, la spécification système, la conception matérielle, le développement logiciel et la synthèse des interfaces, sont réalisés en parallèle (ou du moins sont recouvrantes). De plus, la validation et le test interviennent maintenant plus tôt dans le flot de conception comme illustré sur le schéma de la figure 39.

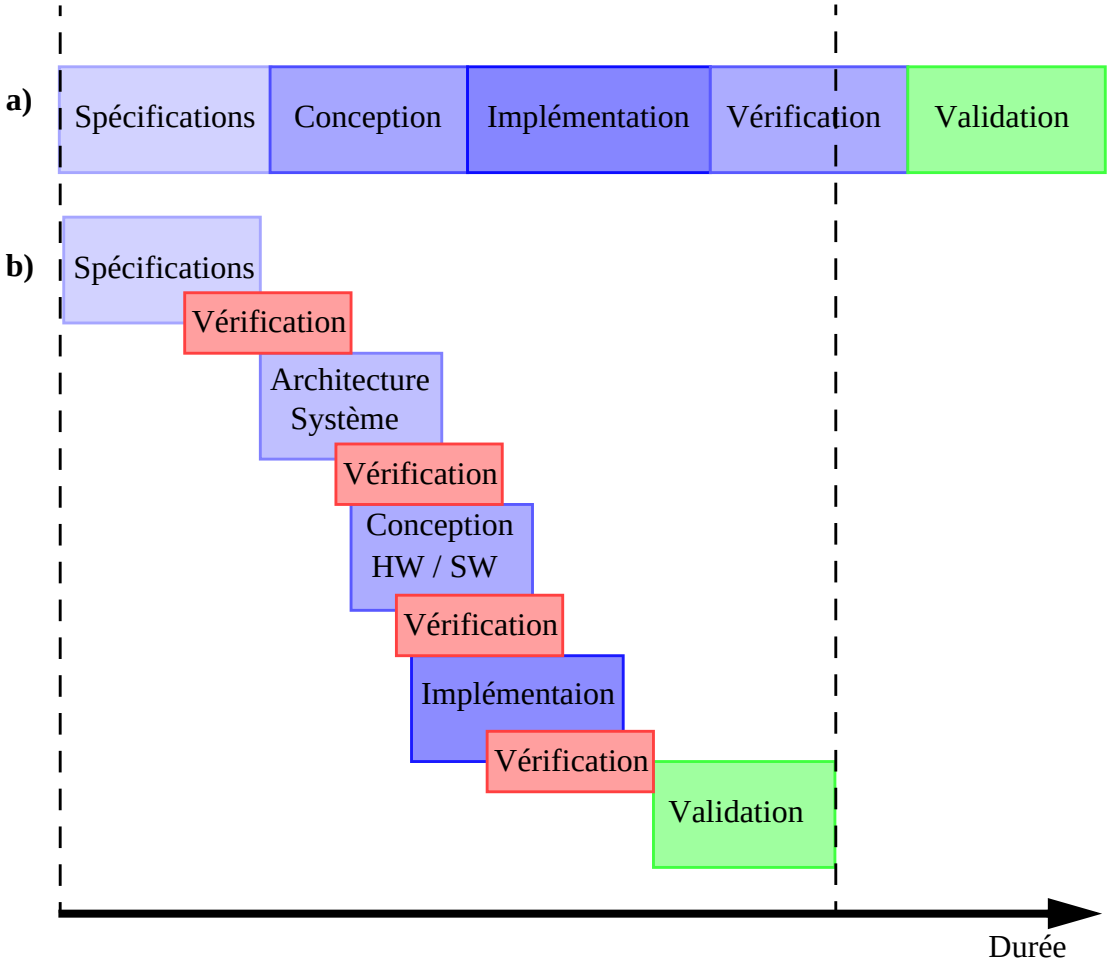


Figure 39: Evolution du flot de conception : a) ancienne méthodologie, b) nouvelle méthodologie

La problématique du concepteur se situe désormais au niveau architecture système : quels circuits (DSP, RISC, GPP, ASIC, SoC) ? quels bus ? quels protocoles de communication ? quelle quantité de mémoire "on chip" et "off-chip" ? quelle quantité de matériel faut-il ajouter ? existe-t-il des IPs utilisables ?
Le concepteur recherche par conséquent des outils efficaces de conception au niveau système. Le tableau 22 présente différents outils du commerce permettant un partitionnement et/ou une

co-simulation matérielle/logicielle à partir d'une spécification au niveau système.

Tableau 22: Les outils de conception haut niveau

Compagnie / outils	Fonctionnalités
Cadence Design Systems / Cierto Virtual Component Co-Design (VCC)	Diagramme de spécification comportementale Diagramme de l'architecture Processus d'allocation Modèles de performance pour la simulation Simulateur Environnement d'analyse Passerelle vers des outils d'implémentation
Mentor Graphics / Seamless CVE Monet	Co-vérification HW/SW Prototypage virtuel Optimisations dynamiques Supporte les architectures multi-processeurs API simulateur/debogger ouvert Kit d'intégration de processeurs (nombreux CPUs, DSPs supportés) Vérification formelle des SOCs Outil de génération de testbench Synthèse comportementale
Synopsys / Eaglei	Supporte une large gamme de processeurs (ARM, MIPS, MPC, PPC, Intel, Futjitsu, Hitachi, NEC, Toshiba, TI, STm, IDT, LSI, ...) Optimisation personnifiée des performances et précisions Intégration dans le flot de conception des autres outils synopsys (MemPro : modèle de mémoires, VERA : outil de tests, ...) Deboggage à tous les niveaux d'abstraction. Supporte les architectures multi-processeurs RTOS
Co-Design Automation / Superlog	Système : interfaçage, protocole, machine d'état, file d'attente,... logiciel : processus dynamiques, tableaux, structures, pointeurs (C/C++, Java) Matériel : évènements, concurrence, logique multi-valuée, tâches/fonctions, logique séquentiel/combinatoire (Verilog/VHDL) Validation : Générateur de tests aléatoires

Tableau 22: Les outils de conception haut niveau

Compagnie / outils	Fonctionnalités
Innoveda / Virtual-CPU Pro SW/HW Co-development Environment Visual Elite eArchitect	Co-développement logiciel / matériel RTOS Modèles fonctionnels logiciels Modèle C matériel pour la simulation Editeur graphique C/C++, Système C Modèle haut niveau de communications et des blocs fonctionnels Exploration d'architectures et analyse des performances Génération prototype virtuel du système Deboggage (GDB)
CoWare / CoWare N2C Design system	Spécifications exécutables : C/C++, SystemC, RTC Modélisation fonctionnelle et modèles de performance Génération automatique d'interfaces Conception matériel par raffinement du modèle C C2HDL, intégration d'IPs Co-simulation HW/SW
Cynergy system design Inc. / RTL2C, ArchGen, ASVP Builder	Transcription HDL vers C (RTL) Description comportementale graphique Synthèse des spécifications RTL et C Outil de vérification et de deboggage ASVP : Application Specific Virtual application Prototypes
Arexsys / ArchiMate CosiMate	Exploration d'architectures HW/SW Partitionnement à partir d'une description système (SDL,C/C++) Générateur de code RTL (VHDL/Verilog), de code C et d'assembleur, intégration d'IPs Environnement de co-simulation
C Level / System Compiler (C2HDL) System Modeler	Conception C/C++, SystemC comportementale et RTL Génération de VHDL/Verilog Modèles C pour la concurrence et le pipeline Simulation modèle HDL parallèle/distribué au niveau système
Adelante Technologies / Frontier Design AIRT (Algorithm to RT) Builder Architectural Synthesis Toolkit	Conception niveau système basée C Co-design HW/SW, synthèse d'interfaces, intégration d'IPs Générateur C vers VHDL/Verilog Supporte développement en virgule fixe Fonctions pré-définies pour la manipulation de bits Génération automatique de tesbench

Tableau 22: Les outils de conception haut niveau

Compagnie / outils	Fonctionnalités
Xilinx / Forge	Générateur C/C++, Java vers Verilog personnification des contraintes Vérification, simulation, caractérisation Partitionnement Allocation HW/SW, parallèle, pipeline, distribué
VaST Systems Technology / CoMET	Conception interactive HW/SW au niveau système Exécution du logiciel sur un processeur virtuel (ARM, Hitachi, MIPS, Intel, Motorola,...) Simulation : liens vers les outils de Mentor Graphics, Cadence, Synopsys,...
Foresight systems Inc. / Foresight Co-Design System	Environnement de modélisation et de simulation pour la conception de SoCs Descriptions VHDL, Verilog, C, C++ Co-simulation avec les outils de Mentor Graphics Seamless CVE et Modelsim

Notre choix s'est porté dans un premier temps sur les outils de la société CoWare (CoWare N2C [29]). Cependant, les premières études de codesign réalisées au printemps 2000 ont révélé différents problèmes liés à l'immaturité de l'outil : tout d'abord, l'encapsulation de code C bien que possible ne conduisait pas à un modèle facilement manipulable (outil de co-simulation et de deboggage non stabilisés) et une ré-écriture en RTC ("Register Transfer C" : langage de spécification du "matériel" de la société CoWare) des blocs à partitionner était le plus souvent nécessaire. En effet, l'outil est adapté à la génération d'interfaces entre les blocs écrits pour le "matériel" et les blocs en C mis sur un processeur cible, mais n'est pas adapté à l'exploration d'architectures pour un passage rapide d'une réalisation "logicielle" à une réalisation "matérielle". Ensuite, lors des simulations la quantité de données générées allait au delà des capacités mémoire de notre environnement de travail, de plus, ces données n'étaient pas clairement interprétables pour l'analyse du système. Cet outil davantage orienté co-simulation qu'exploration d'architectures ne satisfaisait donc pas à nos attentes du moment.

Un deuxième outil a donc été étudié, celui de la société ArexSys [30]. La société ArexSys propose un outil d'exploration d'architectures : ArchiMate et un environnement de co-simulation : CosiMate qui seront présentés au paragraphe III.3. Dans le cadre de cette thèse cet outil était intéressant à plus d'un titre, d'une part à cause de son origine universitaire (laboratoire TIMA [31]) et d'autre part, du fait de sa jeunesse (caractéristiques non figées) et de la possibilité de collaboration avec les équipes de développement présentes sur la région Grenobloise. De plus, le langage accepté en entrée par l'outil est le langage de spécification de systèmes, SDL, qui est d'approche facile du fait de la possibilité de décrire le système via une interface graphique (outil ObjectGeode [32]).

III.1 Méthodologie de codesign

La méthodologie suivie procède en deux étapes (figure 40) :
- la spécification système et l'analyse de la complexité,
- le partitionnement matériel / logiciel et l'analyse de l'efficacité de l'implémentation.

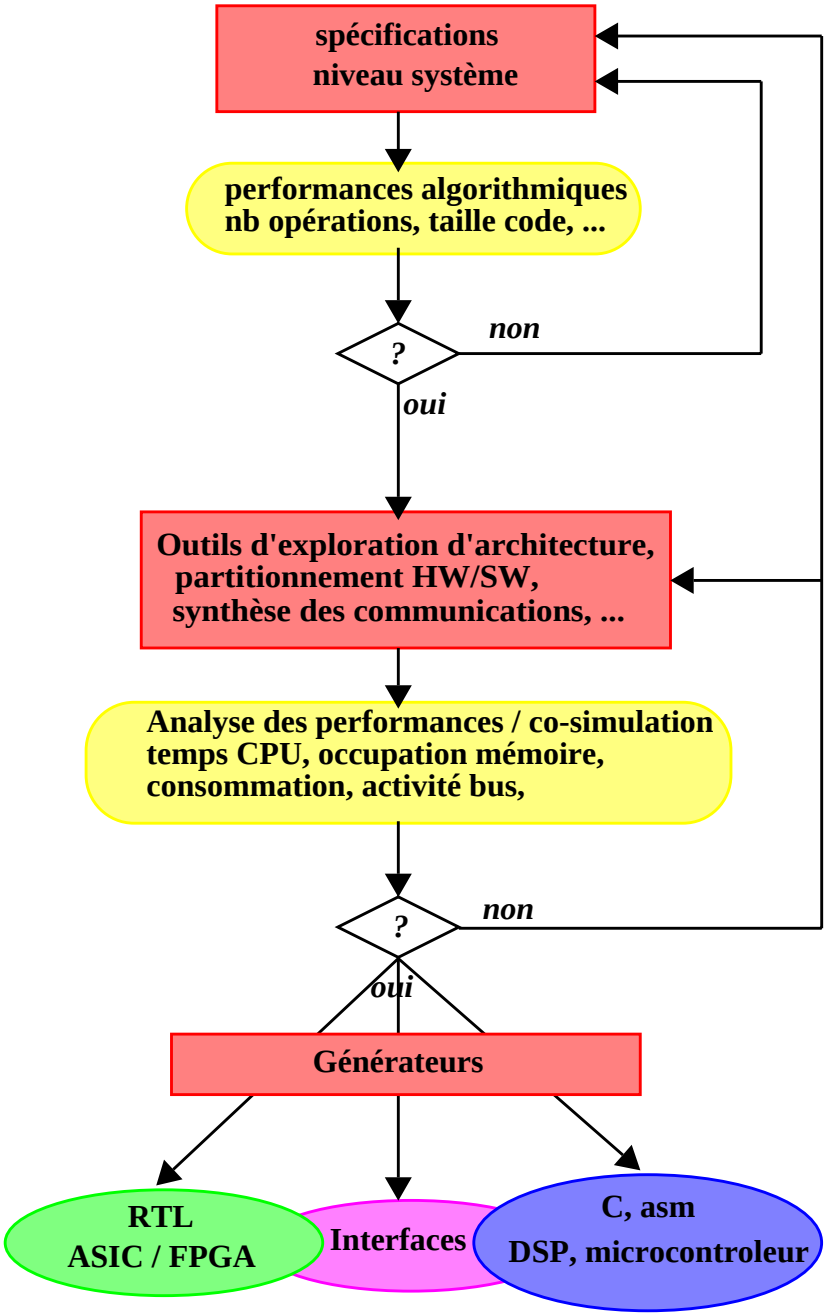


Figure 40: Méthodologie de conception de SoC

Notre approche décrite figure 40 se déroule comme suit :

- dans un premier temps, une spécification au niveau système est réalisée à l'aide d'un langage de haut niveau (SystemC, SDL, Esterel, C/C++, ...). L'objectif est ici d'obtenir un modèle simulable et représentatif de l'implémentation finale. Dans cette description, on doit pouvoir spécifier des tâches qui communiquent entre elles via des signaux, c'est pourquoi les langages SystemC et SDL seront préférés au C pour la spécification. Cependant, certains traitements sont implantés sur des circuits programmables (ARM7, ST20, StarCore, C54x, ...) pour lesquels une description en C est mieux adaptée (compilateur C spécifique à la cible). La solution retenue doit donc autoriser l'encapsulation de fonctions C dans la description au niveau système.

• Après cette spécification, une étape d'analyse de la complexité est effectuée. Le but est de fournir au plus tôt des informations sur l'efficacité de l'algorithme tant en terme de complexité algorithmique, de bande passante et de quantité de mémoire. En outre, une répartition de la complexité d'un algorithme selon les traitements le composant doit également être effectuée car elle fournit des informations pertinentes pour une première orientation des choix de partitionnement matériel / logiciel de l'étape suivante.

• Dans un deuxième temps, on effectue le partitionnement, le choix des protocoles de communications à l'aide d'un outil de codesign du commerce (par exemple ArchiMate, CoWare N2C, ...). Lors du partitionnement, le choix est fait de l'implémentation en matériel de tel traitement et en logiciel de tel autre ou de l'utilisation d'un IP réalisant le traitement souhaité [33] (cf. A.I.3). Un IP est généralement paramétrisable ce qui permet d'explorer rapidement différentes solutions.
• Une analyse des performances est également effectuée à cette étape. Elle consiste en une mesure des taux d'occupation des mémoires embarquées, des charges des coeurs de processeur (s'il y a lieu), des taux d'activité sur les bus et enfin de la dissipation d'énergie du matériel dédié.

Nous allons maintenant détailler notre méthodologie pour une spécification en SDL et l'utilisation de l'outil Archimate.

III.2 Le langage SDL

Le langage SDL (Specification and Description Language) est un langage formel de l'UIT spécifié en 1980 sous la recommandation Z.100 [34]. Ce langage est adapté à la spécification d'applications complexes, interactives, temps réel et événementielles mettant en oeuvre des tâches concurrentes communiquant par le biais de signaux. Il est largement utilisé dans le domaine des télécommunications mais ne lui est pas entièrement dédié. Il peut être utilisé pour spécifier tout système communiquant. La base de cette description est la communication entre machines d'états apparaissant comme des processus. Les communications sont des signaux connectant des processus entre-eux et/ou avec l'environnement du système.
Le langage SDL a été privilégié par la société Arexsys comme entrée de leur outil car il satisfait aux principales contraintes de description de systèmes matériels :
- La plupart des systèmes matériels possèdent des processus concurrents. Les systèmes matériels sont parallèles et le langage SDL permet de décrire des tâches parallèles.
- Les systèmes matériels sont distribués, un système peut être composé de processus travaillant à des cadences différentes. Du fait de la nature asynchrone des communications en SDL, une telle description est rendue possible.
- Les systèmes matériels possèdent une horloge. Cependant, la spécification des horloges est faite au niveau physique (VHDL, Verilog). Le langage SDL n'est utilisé que pour une spécification à un haut niveau d'abstraction.
- L'utilisation du langage SDL conduit à une description sémantique formelle qui permet une vérification formelle ainsi qu'une validation au niveau système.
- Le langage SDL n'est pas performant pour décrire des blocs hautement calculatoires mais cette restriction est levée par la possibilité d'inclure des descriptions en C dans la spécification SDL.

Nous allons illustrer la spécification d'un système en SDL dans l'environnement ObjectGeode [32] sur un exemple simple de ping pong.

• Description fonctionnelle de l'exemple

- Soit deux processus, P1 et P2 qui communiquent via un canal bi-directionnel (*ch1*) sur lequel transitent des signaux (*to_block1* et *to_block2*).
- P1 est le maître de P2, il initialise une variable (*val*), envoie une requête de calcul à P2 ainsi que la valeur de "*val*" (signal to_block2) et attend la réponse de P2.
- P2 est activé par le signal *"to_block2"*. L'entier lié à la requête est affecté à une variable locale (*val*). "*val*" est ensuite passée comme paramètre à une routine C (*function()*) déclarée en externe. Celle-ci effectue un décrément de 2 de "*val*". Cette nouvelle valeur est ensuite retournée au processus P1 après un acquittement (signal *to_block1*). P2 se retrouve alors dans un état d'attente d'une nouvelle requête de calcul.
- P1 qui était en attente de l'acquittement de P2, effectue un incrément de 1 de la valeur retournée puis la teste. Si "*val*" est supérieur à 1 alors une nouvelle requête est envoyée à P2, sinon le système s'arrête.

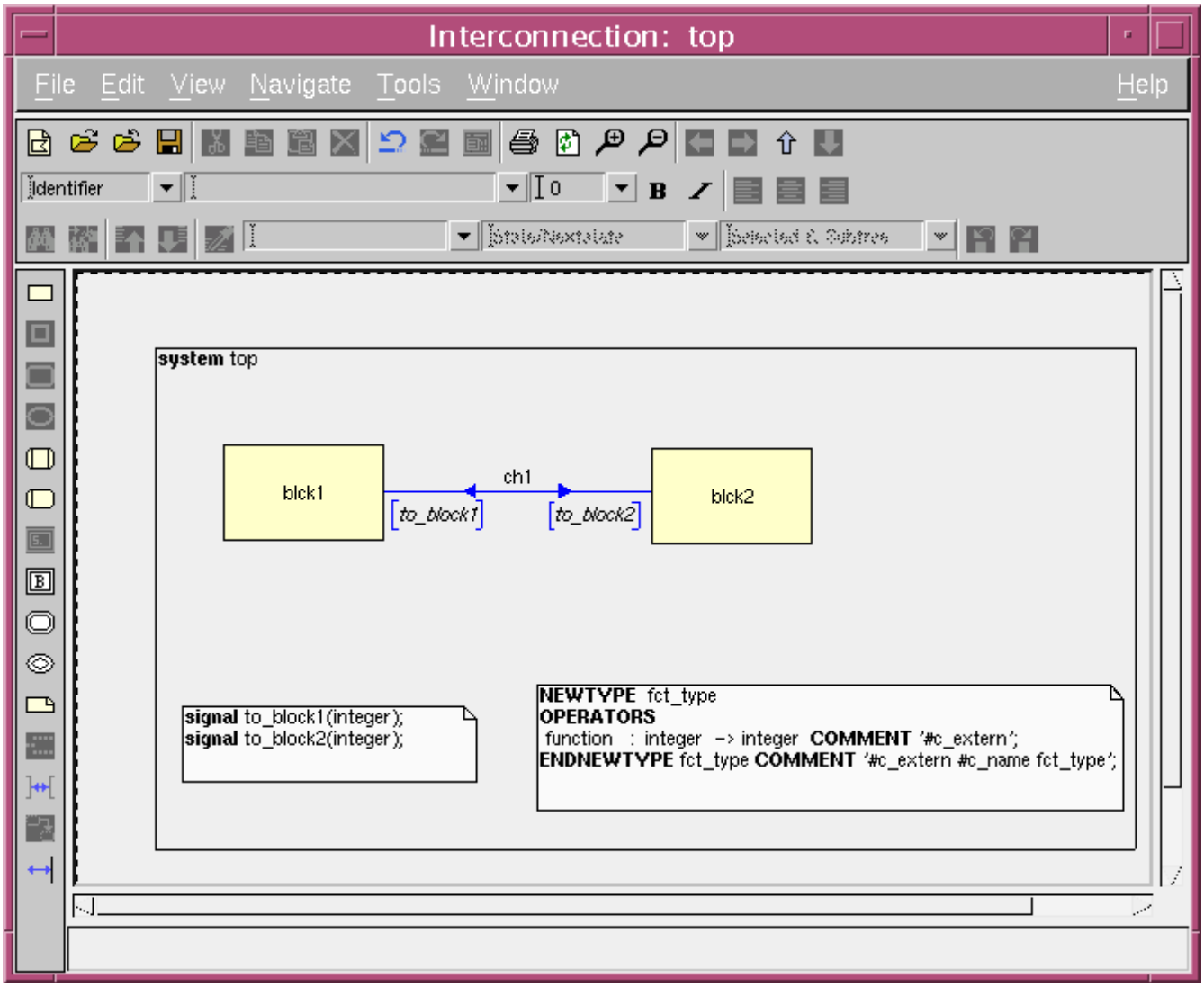
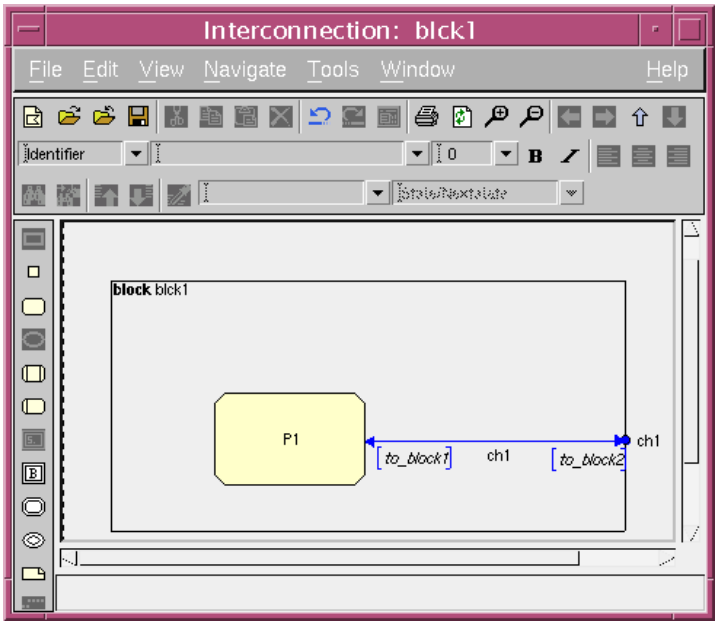


Figure 41: Spécification du système (signaux, blocs de traitement)

Le bloc 1 (*blck1*) communique avec le bloc 2 (*blck2*) via un canal (*ch1*) sur lequel transitent les signaux "*to_block1*" et "*to_block2*" envoyés par *blck2* (respectivement *blck1*) aux blocs *blck1* (respectivement *blck2*). Ces signaux sont définis dans la fenêtre en bas à gauche et correspondent tous les deux à un entier. Dans la fenêtre en bas à droite, un opérateur (*function*) est défini

comme opérateur externe écrit en C.



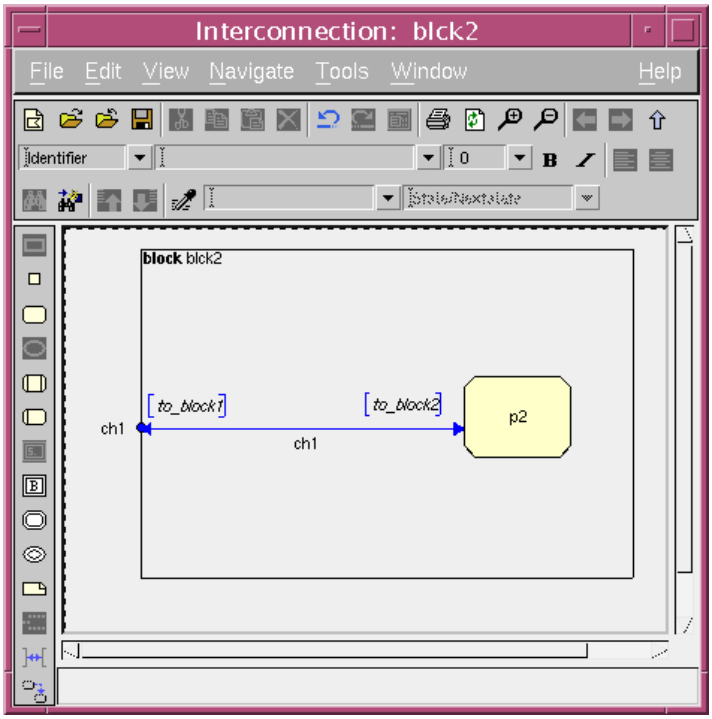


Figure 42: spécification des blocs (processus)

Chaque bloc contient au moins un processus. Ici, un seul processus est affecté à chaque bloc: P1 ou P2. Ces deux processus communiquent avec leur environnement respectif via le canal "*ch1*".

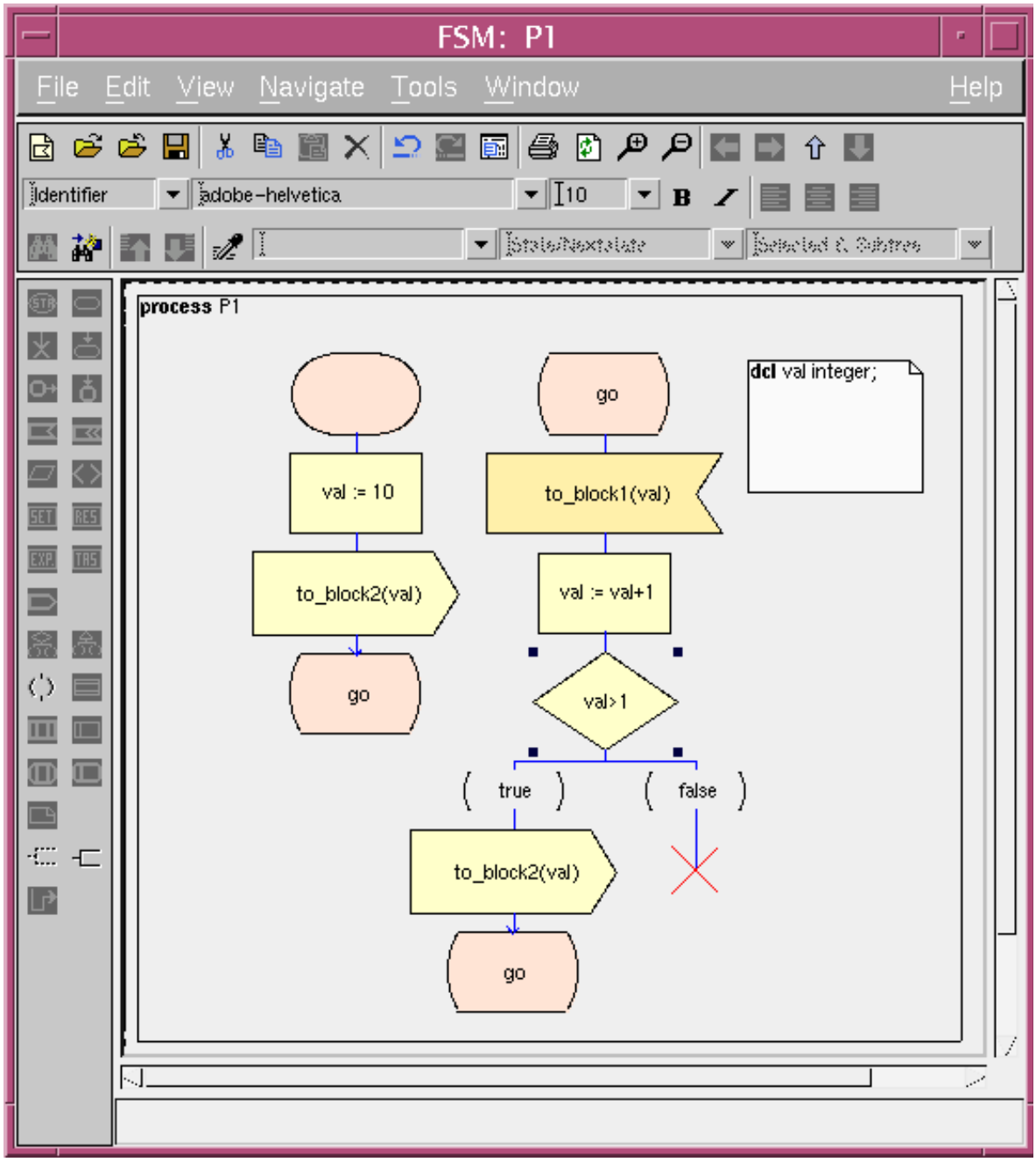


Figure 43: Machine d'états du processus 1

L'état de gauche correspond à l'initialisation du processus et envoie une requête à P2. L'état de droite est un état d'attente de l'acquittement de P1. Dans ce deuxième état le décrément ainsi que le test de "*val*" sont effectués. Un signal correspond dans ce cas à la fois à un événement et à une donnée (*val*).

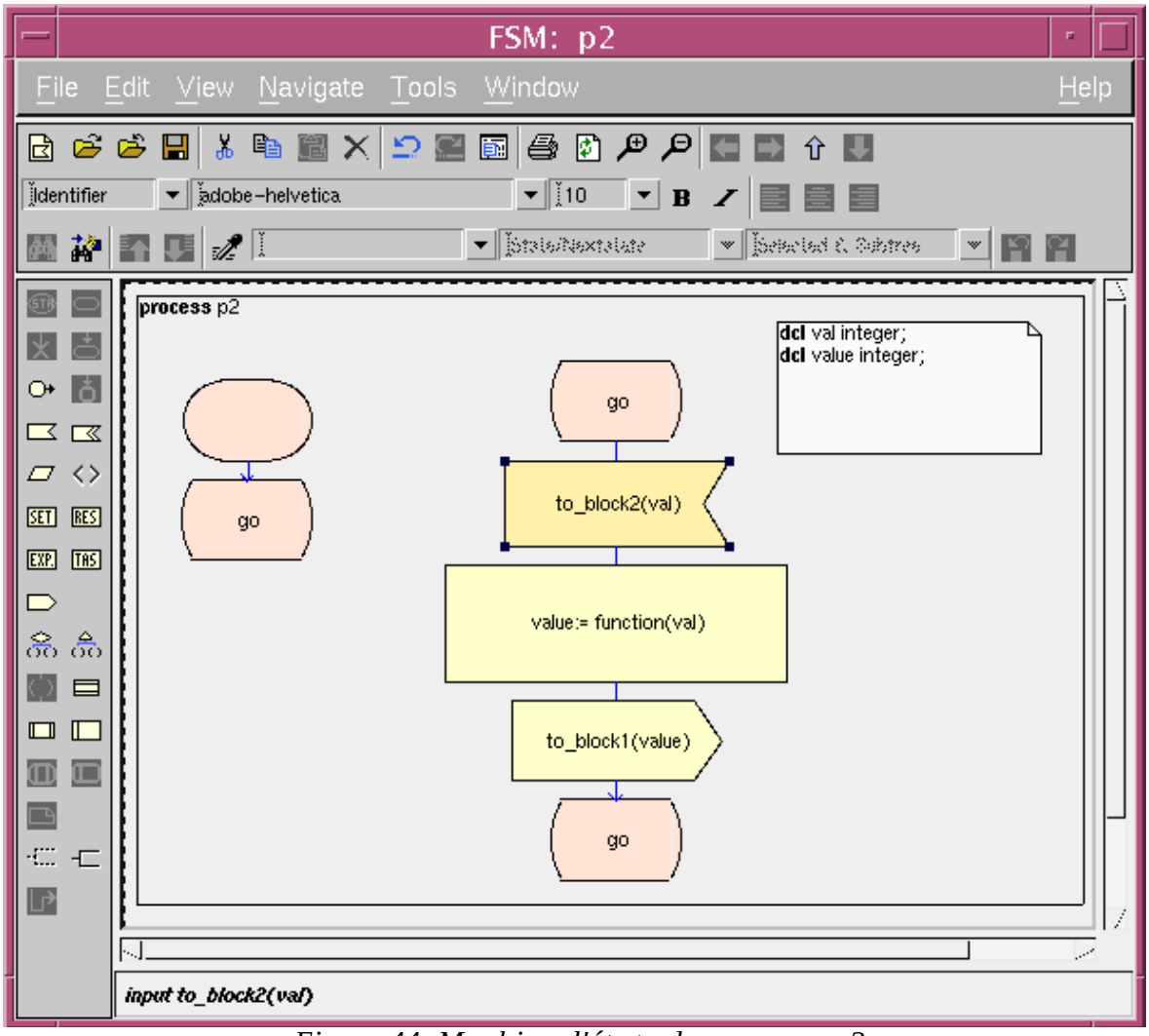


Figure 44: Machine d'états du processus 2

La machine d'état du processus P2 est en attente d'une requête de calcul (état *go*) après l'état d'initialisation (état de gauche). Dès la requête reçue, un appel à la routine "*function*" est effectuée et la valeur issue de ce traitement est retournée à P1.

Le système peut être simulé sous ObjectGeode. L'interface de simulation fournit des indications sur les états actifs des machines d'états et sur les signaux transitant sur les canaux de communication.

III.3 Outils de la société Arexsys

La conception de systèmes complexes tels que les codeurs vidéo serait grandement accélérée par une prise en compte au plus tôt des contraintes de l'application. L'approche adoptée ici repose donc sur une modélisation de notre système à un haut niveau d'abstraction et sur l'utilisation d'outils de génération / exploration d'architectures matérielles et logicielles. Les outils présentés issus de la recherche Universitaire (Tima) [31] sont développés et commercialisés par la société ArexSys [30]. Ils se décomposent en deux briques :
- ArchiMate, pour le partitionnement "matériel / logiciel" et la génération des communications entre processus matériels et/ou logiciels.
- CosiMate, pour la co-simulation du système spécifié.

III.3.1 ArchiMate

Cet outil prend en entrée une spécification au niveau système (langage SDL) et la transforme soit en une description "matérielle" au niveau RTL (Register Tranfer Level), soit en un code C pour un processeur cible.
ArchiMate facilite la tâche du concepteur pour :
- le partitionnement structurel,
- la synthèse des communications,
- le choix de l'affectation des tâches au matériel ou au logiciel,
- la synthèse "matérielle" (génération de la spécification RTL des blocs matériels),
- le ciblage "logiciel" (choix d'un processeur cible),
- la génération d'interface.

La figure 45 ci-dessous représente les fonctionnalités couvertes par ArchiMate.

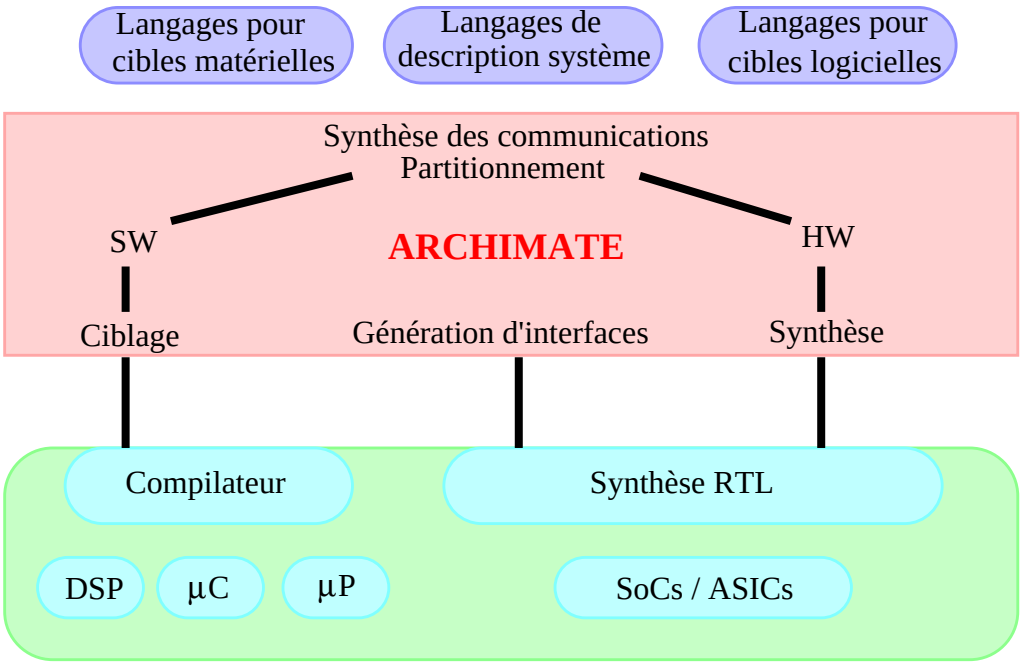
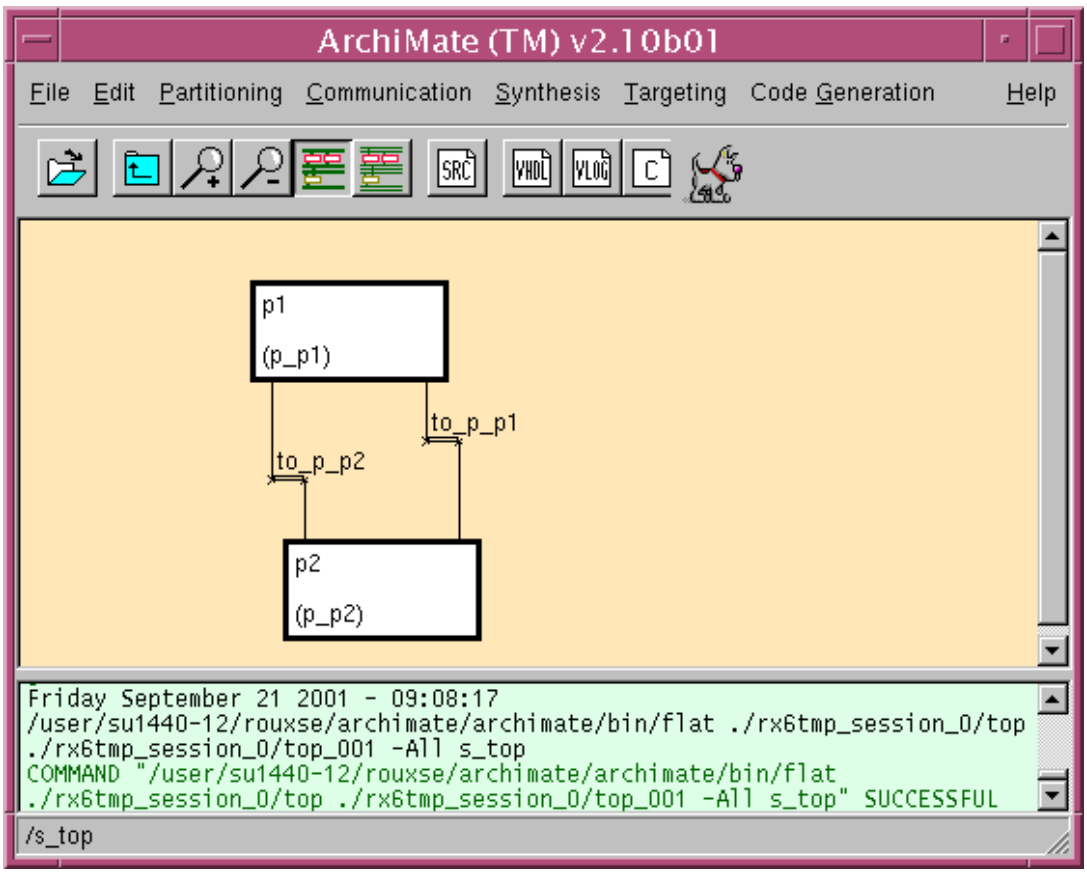


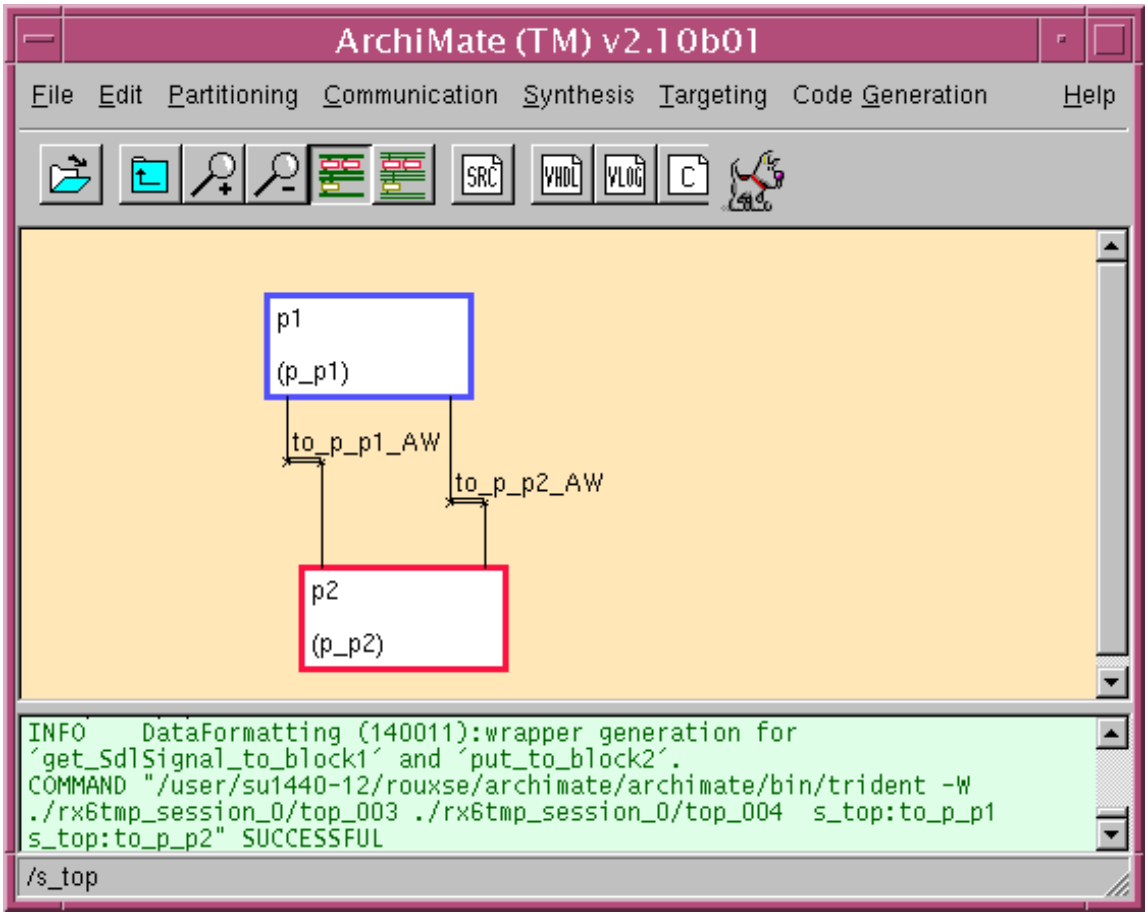
Figure 45: Flot de conception ArchiMate

Nous allons maintenant reprendre plus en détail ces différentes étapes en les illustrant sur l'exemple du ping pong.
Partitionnement structurel
Il s'agit de la première opération effectuée sur la spécification au niveau système. Cette transformation a pour but de produire à partir du modèle fonctionnel un modèle du système plus proche de son architecture finale. Pour ce faire, trois opérations peuvent être exécutées :
- tout d'abord, la mise à plat de la hiérarchie de la description fonctionnelle au niveau système (suppression de la structure hiérarchique du système, cette étape est obligatoire). La notion de *bloc* disparaît et laisse place à une description contenant uniquement des processus,
- ensuite, le regroupement des processus définissant la structure matérielle et logicielle du système,
- et enfin, le fusionnement des opérations de certaines fonctions logicielles afin de supprimer un niveau de hiérarchie et de grouper leur machine d'états.

Cette étape transforme la description SDL définissant les blocs fonctionnels du système en une spécification du système selon des blocs définissant sa structure matérielle et logicielle.
Sur le diagramme ci-dessous seule l'opération de "mise à plat" de la hiérarchie est effectuée.



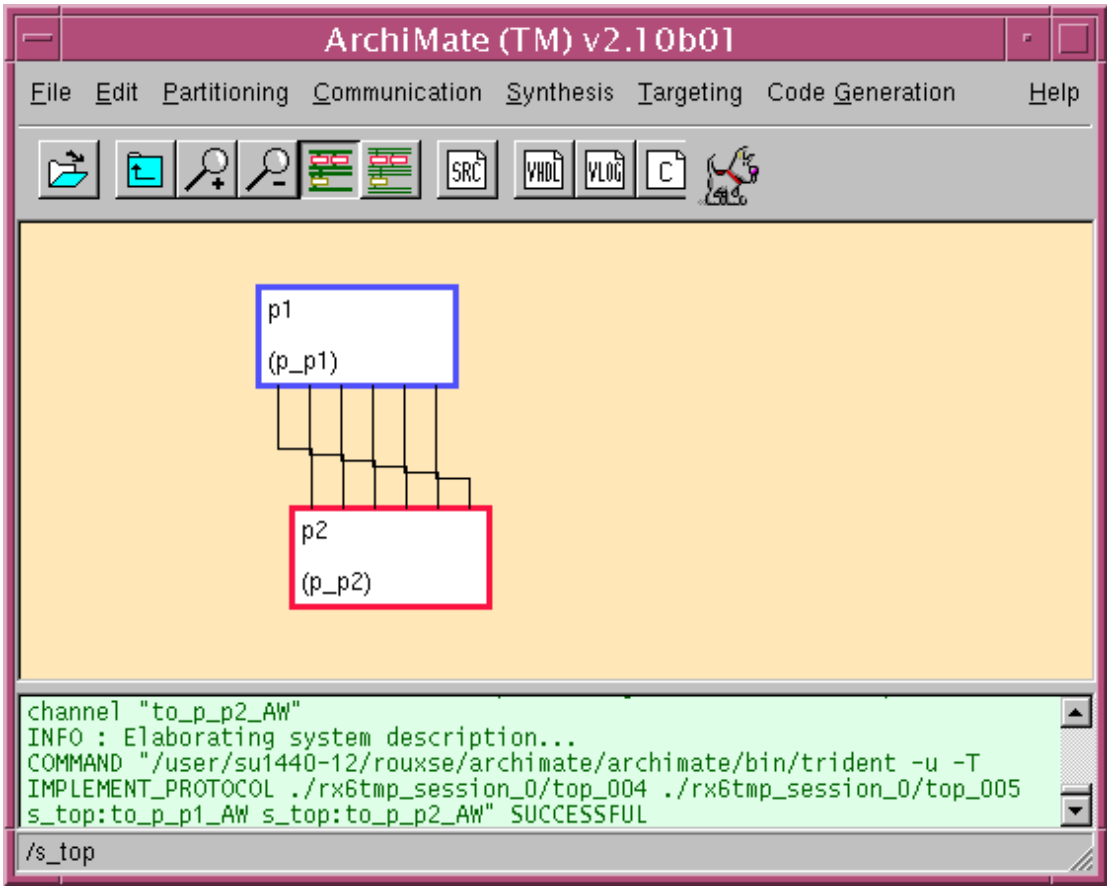
Affectation à du matériel ou du logiciel : 4 possibilités
- Matériel : génération de code VHDL,
- Logiciel : génération de code machine pour processeur cible (compilateur croisé),
- IP : bloc matériel réalisant une fonction donnée (les coeurs de processeur ne rentrent pas ici dans cette catégorie, ils sont définis lors de l'affectation du logiciel),
- Environnement : blocs faisant partie de la description de l'environnement du système. Seules les interfaces avec le système sont générées.



Ici, le processus P1 est implémenté sous forme "matériel" (bleu) et le processus P2 est réalisé sous forme "logiciel" (rouge).

Synthèse des communications

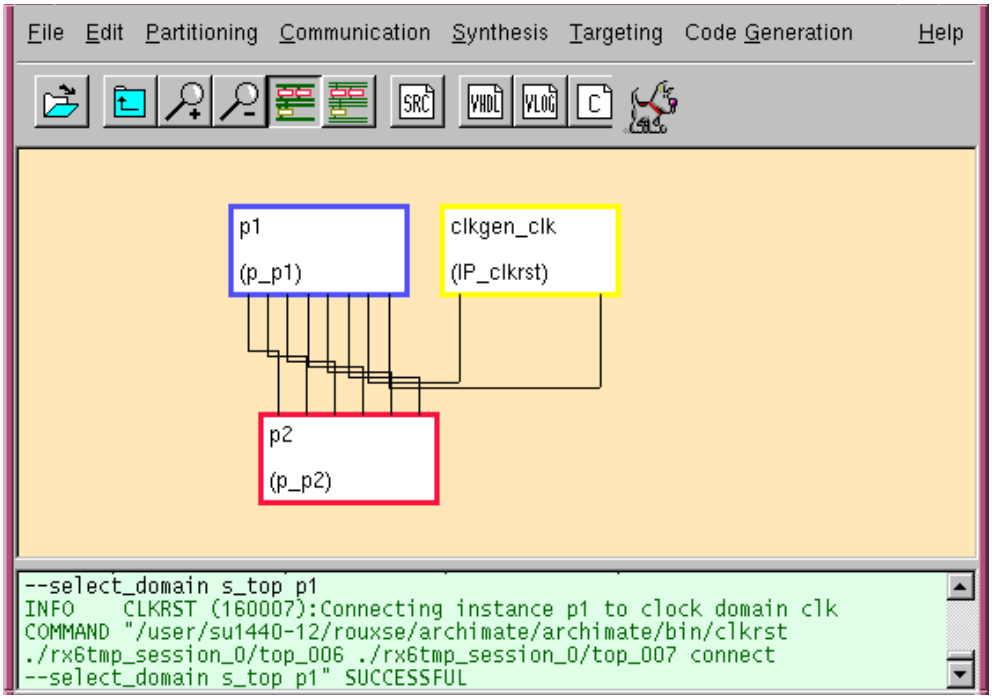
Cette étape exploite la description abstraite des canaux de communication en SDL avec une librairie de protocole afin de générer un modèle comportemental qui définit les connections, les ports et les procédures dédiées aux communications. Le protocole implanté peut aller de la communication à travers une mémoire partagée, en passant par des bus partagés jusqu'à des communications "hand-shake" dédiées.



Les protocoles implémentés entre P1 et P2 sont de type "hand-shake".

Synthèse "matérielle"

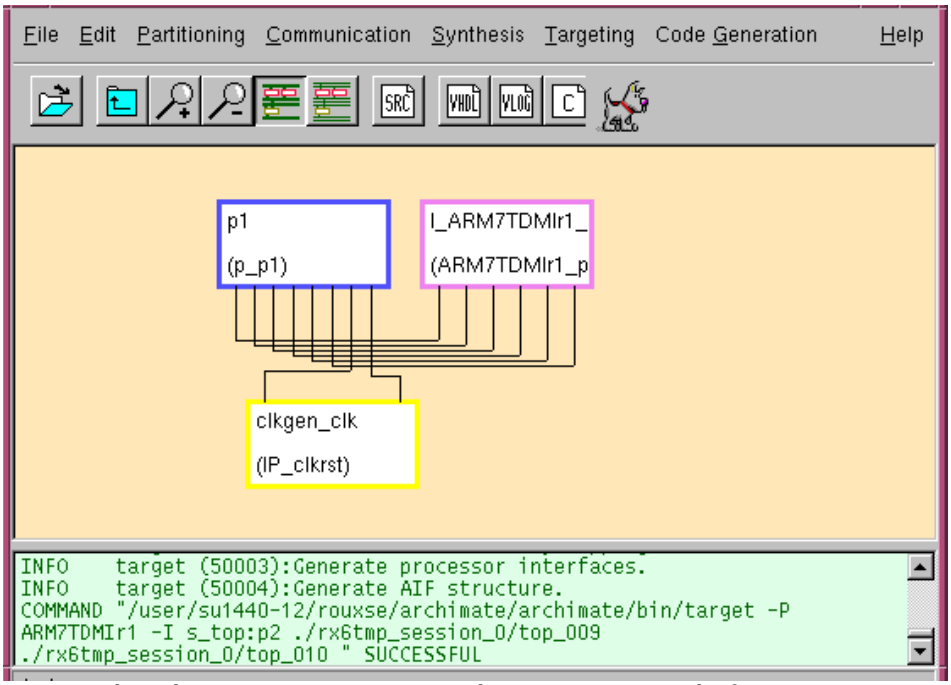
L'étape de synthèse "matérielle" de la description fonctionnelle du système génère une description au niveau comportemental ou au niveau RTL interprétable par les outils de synthèse comportementale ou RTL du commerce. Le mode comportemental permet la description d'opérations complexes telles que l'appel de procédures et de fonctions, les boucles infinies,... Cette synthèse est orientée flot de contrôle et génère pour chaque tâche une machine d'états. La transition entre état prend un cycle d'horloge, ce qui impose pour les tâches complexes des contraintes de description au niveau système (description SDL). En effet, le "scheduler" (pour le niveau compotemental et a fortiori pour le niveau RTL) ne gère pas les tâches combinatoires "trop complexes" (limite sur le nombre d'opérations élémentaires d'une tâche) ce qui implique un découpage plus fin de la machine d'états au niveau SDL. Outre la contrainte d'écriture plus sévère sur la spécification en langage SDL, ceci soulève le problème de l'optimisation du code pour les processus dont le choix d'une implémentation sous forme "matériel" ou "logiciel" n'est pas arrêté. En effet, si le processus est finalement réalisé sous forme "logiciel" l'optimisation du code pour le coeur de processeur cible sera certainement plus ardue (parallélisme d'instruction, utilisation optimale des unités arithmétiques,...).



Une horloge est spécifiée et connectée au bloc "matériel" puis une synthèse "matérielle" au niveau comportemental est effectuée. La synthèse "logicielle" est également réalisée (sans cible particulière) puis un modèle C++ exécutable pour la simulation du système est généré.

Ciblage "logiciel"

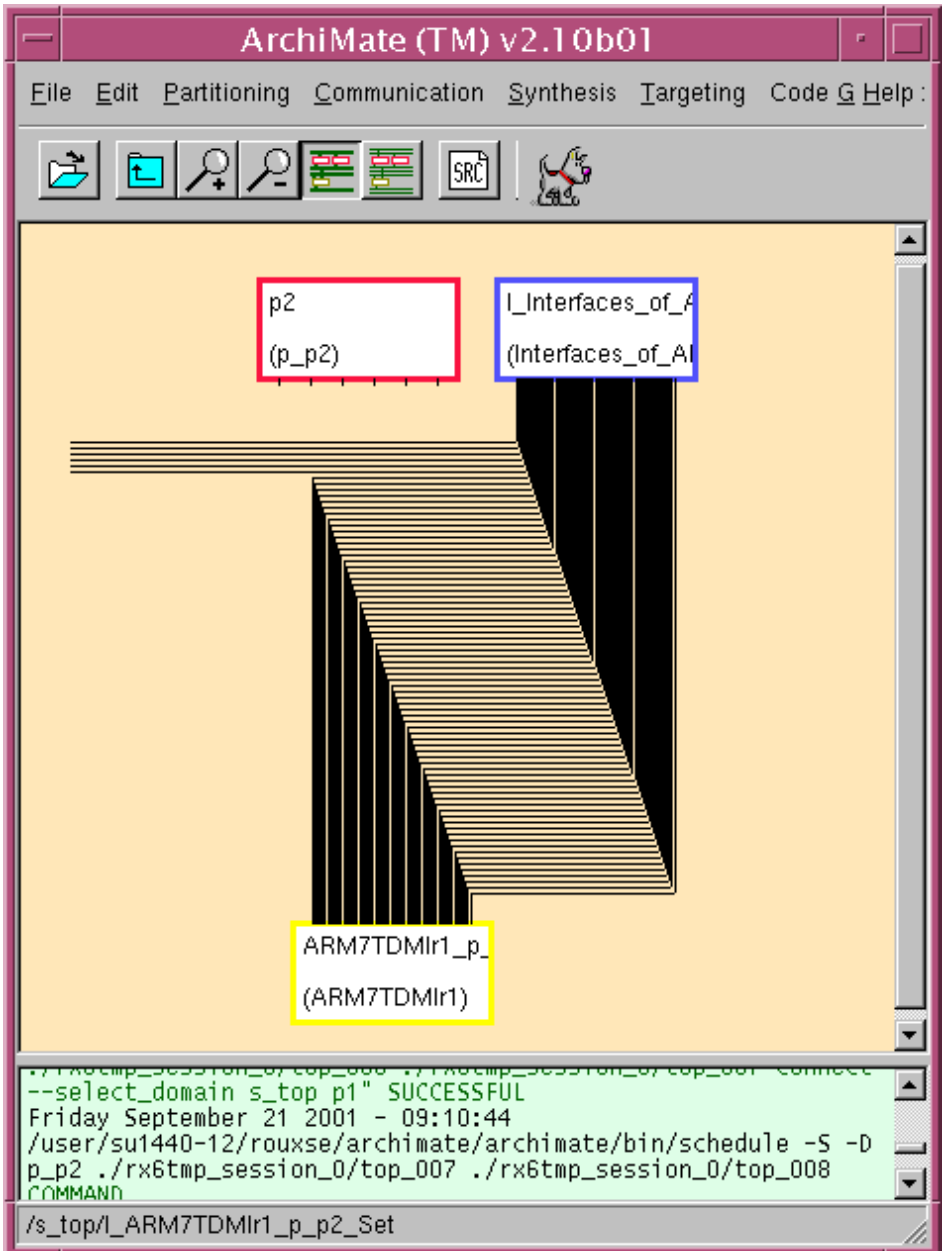
Cette procédure fixe l'ordre d'exécution des tâches parallèles et les affecte à des processeurs spécifiques (par exemple, Motorola 68000, MC68040, IBM PowerPC 603e, ARM 7 TDMI, ST 10). Le but est d'allouer les processus "logiciel" à un ou plusieurs DSPs, micro-contrôleurs ou micro-processeurs selon la nature du processus et sa complexité.



Le processus P2, réalisé en logiciel, est compilé pour la cible spécifique ARM7TDMI.

Génération d'interfaces

Il s'agit, ici, de générer l'environnement nécessaire à l'implantation des processeurs spécifiés à l'étape de ciblage "logiciel". La synthèse des interfaces entre le "matériel" et le "logiciel" ainsi que la synthèse des "drivers" logiciels sont effectuées.



Lors du ciblage "logiciel", l'interface entre l'ARM7TDMI et le "matériel" est générée (boîte bleue en haut à droite). On retrouve en haut à gauche une boîte rouge symbolisant le logiciel (processus P2) implanté sur l'ARM7TDMI (boîte jaune en bas : IP du coeur de processeur ARM7TDMI).

III.3.2 CosiMate

Cet outil est un environnement de co-simulation. Il permet la simulation conjointe de systèmes hétérogènes. En effet, CosiMate sert de passerelle de communication entre différents simulateurs mis en oeuvre pour valider un système hétérogène (figure 46). Pour ce faire un bus de co-simulation est créé par l'intermédiaire d'un fichier de configuration et d'interfaces décrites en C.

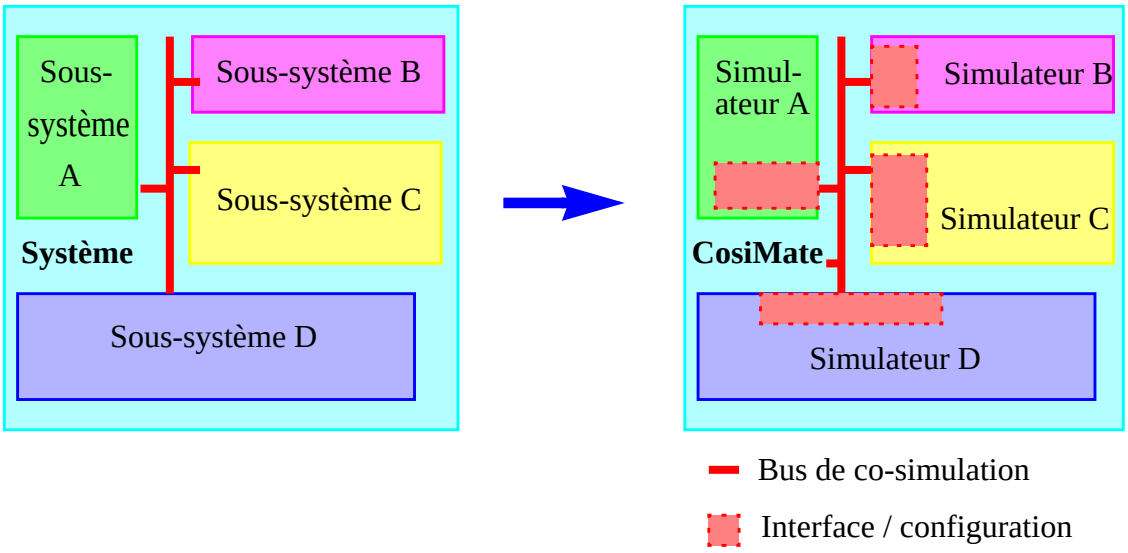


Figure 46: cosimulation sous CosiMate

Cosimate est un environnement de co-simulation supportant les simulateurs suivants : Synopsys VSS, Synopsys COSSAP, Cadence Leapfrog, Mathworks Matlab, Simulink, Mentor Graphics ModelSim, Telelogic Object Geode, ainsi que de nombreux debogger C. Il est possible de rajouter de nouveaux simulateurs.

III.4 Evaluation de l'architecture

III.4.1 Introduction

On a vu au chapitre II comment analyser les performances au niveau algorithmique. Cette complexité se résumait à la complexité arithmétique et à la bande passante. Nous allons voir dans ce chapitre la complexité système au niveau architectural, mesurée en termes de :
- goulot d'étranglement et encombrement du bus,
- temps d’exécution du logiciel,
- latence et cadence du système,
- occupation de la mémoire,
- fréquence du bus,
- latence des données,
- charge des processeurs,

se traduisant par une certaine dissipation d'énergie que l'on cherche à réduire.
L'objectif de l'exploration d'architectures est justement de fournir une évaluation de l'architecture retenue pour implémenter un algorithme selon la consommation d'énergie.
La figure 47 décrit les différents niveaux de conceptions, les techniques de réduction de la consommation associées à chacun de ces niveaux et les gains en consommation maximum envisageables.

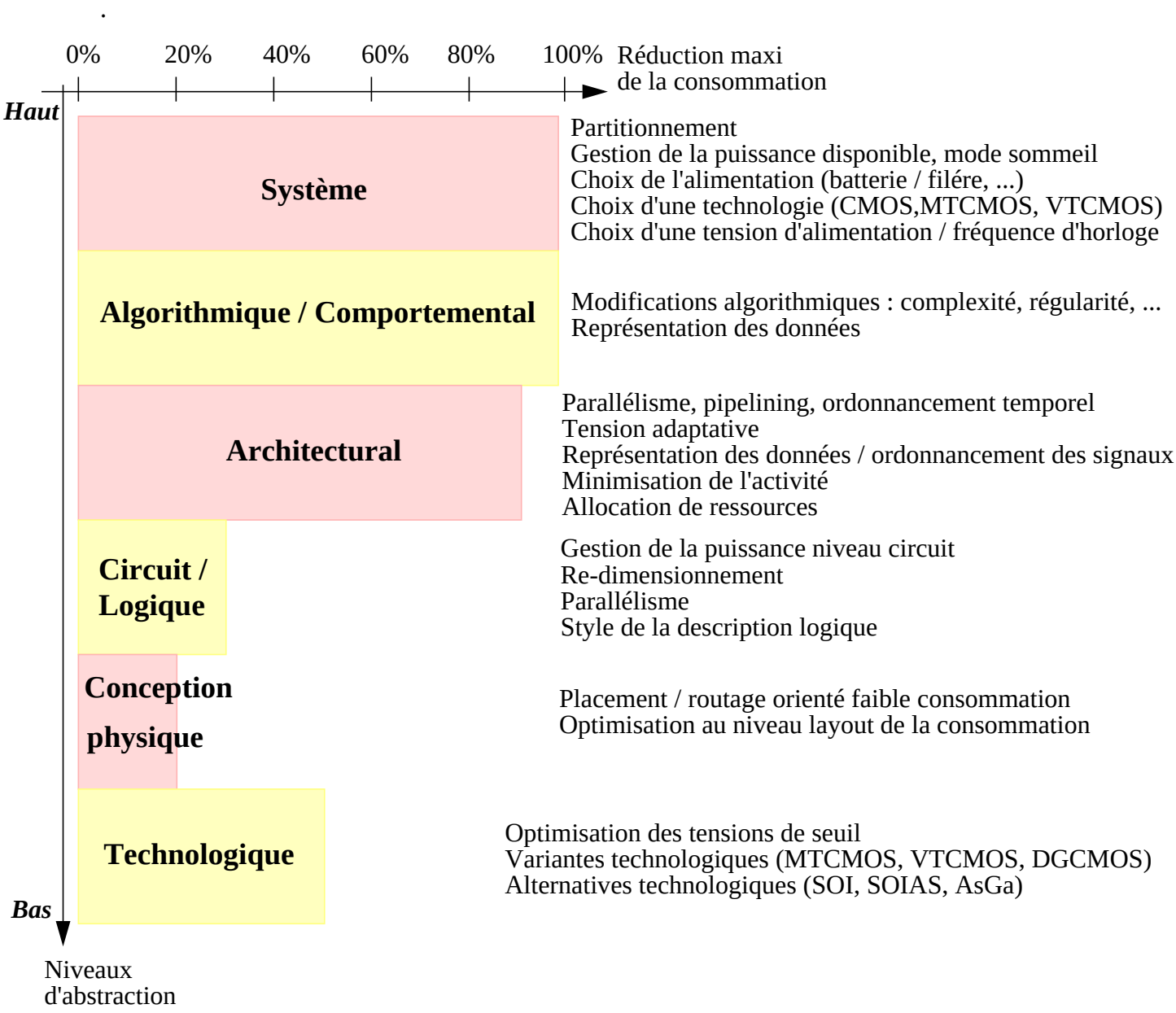


Figure 47: niveaux d'abstraction et gain en consommation

Dans cette phase de l'étude, nous nous plaçons au niveau système et architectural. Par ailleurs, l'évaluation de performance de l'architecture (adéquation algorithme - architecture) sera basée principalement sur le taux d'activité des bus, pour l'application de codage vidéo.

III.4.2 Considération sur la consommation

Nous distinguons, comme dans la première partie, deux types de SoC (cf. A.I, figure 4) : le SoC de type I basé ASIC et le SoC de type II basé coeurs de processeur.
Le SoC de type I intégrant des applications entièrement implémentées en "matériel" ne correspond pas à notre attente de flexibilité de programmation. Nous orientons, par conséquent, nos études sur le SoC de type II qui confère au système une plus grande flexibilité pour l'implantation de nouveaux algorithmes.

Différentes techniques de réduction de la consommation (les paramètres technologiques étant fixés) en fonction des éléments du SoC peuvent être employées :
- Système : globalement asynchrone-localement synchrone, tension adaptative [54].
- CPU, DSP : pour une architecture donnée, complexité algorithmique, compilation, instructions spécifiques (ex : Somme des valeurs absolues des différences de n pixels, TI C55 [55]).
- Matériel : parallélisme, pipeline, ... [56]
- Communication : représentation de données, activité, routage [57].
- Mémoire : taille, localité [58].

• Les principes généralement retenus pour estimer la consommation d'un circuit en technologie CMOS reposent sur l'équation suivante :

$$P = P_d + P_f + P_c + P_s \qquad \text{(B.8)}$$

où P_d est la puissance dynamique, P_f est la puissance due au courant de fuite, P_c est la puissance due au courant de court circuit, et P_s est la puissance statique.
Cependant, il est commun de négliger les dissipations d'énergie dues au courant de court circuit, au courant de fuite et à la puissance statique par rapport à la puissance dynamique (P_d est typiquement à l'origine de 80 % de la puissance totale) . La puissance dissipée est alors représentée par l'équation :

$$P \approx P_d = \alpha \cdot C \cdot V_{dd}^{\,2} \cdot f \qquad \text{(B.9)}$$

où
- α est l'activité du circuit (moyenne de noeuds commutés)
- C est la capacité de charge
- V_{dd} est la tension d'alimentation
- f est la fréquence de fonctionnement du circuit

Afin de réduire la consommation, il faut donc réduire ces paramètres. D'après cette équation, le plus efficace est de réduire la tension d'alimentation qui agit de manière quadratique sur la consommation. Cependant, la réduction de Vdd conduit à une augmentation du temps de traversé des portes selon l'équation (B.10), ce qui limite le gain possible et nécessite donc une optimisation des autres paramètres.

$$T \approx \frac{C_c \cdot V_{dd}}{\varepsilon \cdot (V_{dd} - V_t)^2} \qquad \text{(B.10)}$$

où
- T est le délai du circuit
- ε est le paramètre technologique
- C_c est la capacité du chemin critique
- V_t est la tension de seuil

• Toutefois, ce modèle n'est pas adapté à l'évaluation de la consommation d'un SoC. L'analyse et la réduction éventuelle des paramètres intervenant dans la dissipation d'énergie d'un circuit CMOS doit tenir compte des différents composants constitutifs du système. En effet, un système sur puce peut être composé de coeurs de processeur (DSP, RISC, GPP), de mémoire, de

matériel dédié communiquant par l'intermédiaire de bus, chaque composant ayant un comportement propre vis à vis de la consommation d'énergie. La puissance, *P*, dissipée par un système doit donc être estimée selon l'équation (B.11).

$$P = P_{CPU} + P_{DSP} + P_{HW} + P_{COM} + P_{MEM} \qquad \text{(B.11)}$$

où
- P_{CPU} , puissance consommée par le coeur de micro-contôleur
- P_{DSP} , puissance consommée par le coeur de DSP
- P_{HW} , puissance consommée par le hardware dédié
- P_{COM} , puissance consommée par les communications (bus, interfaces, ports,...)
- P_{MEM} , puissance consommée par la mémoire embarquée sur le SoC

Dans ce cas les différents paramètres (C, α, V_{dd}, f) peuvent ou non être réduits suivant l'élément concerné et de nouveaux paramètres influant sur la consommation peuvent être introduit tels que le type de mémoire (DRAM, SRAM, ...), la taille de la mémoire, le nombre de niveaux de cache, la largeur des bus, le choix d'un IP, ...
Cette approche pour l'évaluation de la consommation d'un SoC présente donc l'avantage de pouvoir distinguer les différents types de matériel en fonction de leurs caractéristiques propres, et notamment dans le cas des coeurs de processeur programmable où une mesure du nombre d'instructions (MIPS) utilisé pour implanter un algorithme peut directement être reliée à la consommation du processeur en utilisant les données technologiques de celui-ci [35][36][37]. L'énergie consommée par les blocs en "matériel" dédié est estimable par une simulation au niveau RTL ou comportementale [38][39] et peut être réduite grâce à différentes techniques d'optimisation de l'architecture (parallélisme, pipeline, "clock gating", ...) largement étudiées dans la littérature [40][41].

• Le type du SoC étant fixé et le comportement des co-processeurs ("matériel" dédié et coeur de processeur) vis à vis de la consommation étant supposé connu, la consommation dépend désormais principalement de la répartition des traitements de l'algorithme entre les blocs matériels et logiciels, ainsi que des canaux de communications. C'est ce dernier vecteur de consommation que nous proposons d'étudier par la suite. En effet, la consommation due aux interfaces de communication entre les tâches (comme les bus) intervient considérablement dans la dissipation d'énergie d'un système. D'après [42], la part de consommation due aux bus dans la consommation totale du SoC varie de 3 % à plus de 40 % suivant la nature des bus (largeur, encodage,...).

Le but est ici de pouvoir mesurer l'influence sur la consommation de différents partitionnements matériel / logiciel et de différents algorithmes à travers une évaluation de l'activité des signaux inter-blocs et le choix des protocoles de communication. Cette approche permet également de mesurer l'impact de la représentation des données sur l'activité des bus. Il existe, en effet, de nombreuses techniques de codage dans la littérature telles que le codage en mode inversé du bus [43], le codage par transmission des différences, et également "l'equal split", le "no care padding" et bien d'autres encores [44][45]. Quelques unes de ces méthodes sont présentées succinctement ci-après.

Méthode de bus inversé
Cette méthode utilise une ligne de contrôle supplémentaire ainsi qu'un bloc d'encodage et de décodage des données. A chaque nouvelle donnée, la distance de Hamming par rapport à la donnée précédente est calculée. Si la distance de Hamming est supérieure à la moitié de la taille du bus alors la valeur inverse de la donnée est transmise, la ligne de contrôle étant activée pour indiquer le type de codage. La réduction de l'activité peut atteindre 25 % suivant la distribution des données.
Une méthode généralisée est l'utilisation de "limited weight codes" [46].

Méthode du partage équitable
Lorsqu'une donnée de taille (*n* bits) supérieure à largeur du bus (*b* lignes) doit être transmise, un multiplexage dans le temps est utilisé. La donnée est divisée en 2 blocs puis transmise sur le bus. La méthode la plus "naturelle" est de transmettre autant de bits que possible à chaque transmission, les bits restants étant envoyés lors d'une dernière transmission. La méthode du partage équitable propose de diviser la donnée en *m* paquets égaux (à +/- 1 bit près). Par exemple, supposons que la donnée soit codée sur 10 bits et que le bus fasse 8 lignes, au lieu de transmettre 8 bits puis 2 (complétés par des '0') ce qui induit une activité sur toutes les lignes du bus, on transmet deux fois 5 bits en utilisant seulement 5 lignes !

Méthode du "don't care padding"
Cette méthode s'applique dans le cas où la donnée à transmettre est sur un nombre de bits inférieure à la taille du bus. La méthode classique affecte '0' aux bits inutiles. La méthode du "don't care padding" permet de ne pas modifier les bits inutiles à l'émission et d'ignorer ces mêmes bits à la réception.

Méthode basée sur la transmission des différences
Seuls les bits ayant changé par rapport à la donnée précédente sont transmis. Soit deux données de 8 bits, *a* et *b*, transmises successivement sur un bus de 8 bits (par exemple, *a* = '10101101' et *b* = '10110101'), la donnée transmise pour *b* est alors '00011000'. Ceci n'entraîne pas une réduction de l'activité mais maintient plus de lignes à '0' ce qui est moins coûteux en énergie. Cette méthode peut être étendue à l'utilisation d'un dictionnaire mémorisant les dernières données et ne transmettant que l'indice du dictionnaire (similaire au codage vectoriel en traitement du signal) correspondant à la donnée ayant la plus petite distance de Hamming avec la donnée à transmettre, ainsi que la différence avec la donnée indicée (figure 48).

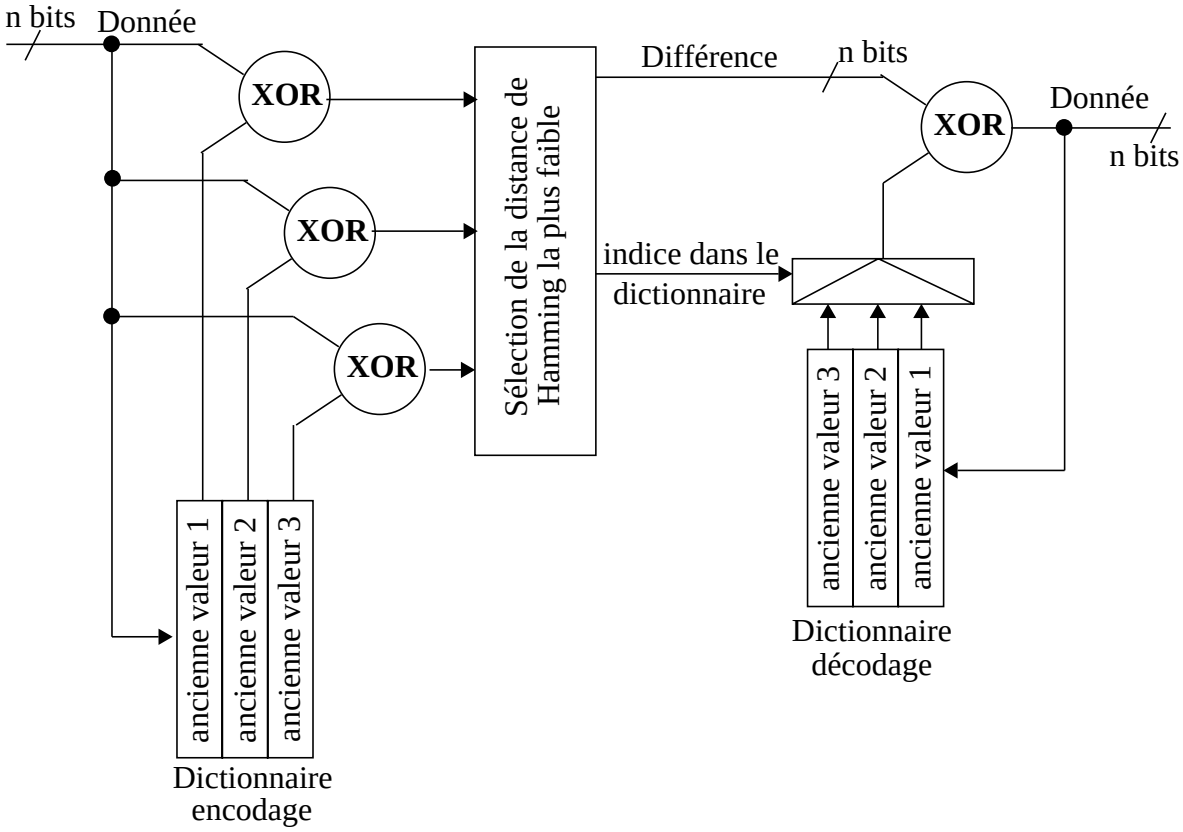


Figure 48: quantification vectorielle appliquée au codage du bus

III.4.3 Estimation de la consommation

Dans ce paragraphe, nous allons détailler notre approche pour l'évaluation de la puissance P_{COM} de l'équation (B.11). Cette puissance à tension fixée est fonction des activités, des capacités et des fréquences des bus du système, elle est définie dans l'équation (B.12) pour l'exemple d'architecture de SoC fournie en figure 49.

$$P_{COM} \propto \sum_{n \in \{N \text{ bus}\}} \alpha_n \cdot C_n \cdot f_n \qquad \text{(B.12)}$$

$$\text{avec} \begin{cases} \alpha_n, \text{ taux d'activité sur le bus } n \\ C_n, \text{ capacité du bus } n \\ f_n, \text{ fréquence du bus } n \end{cases}$$

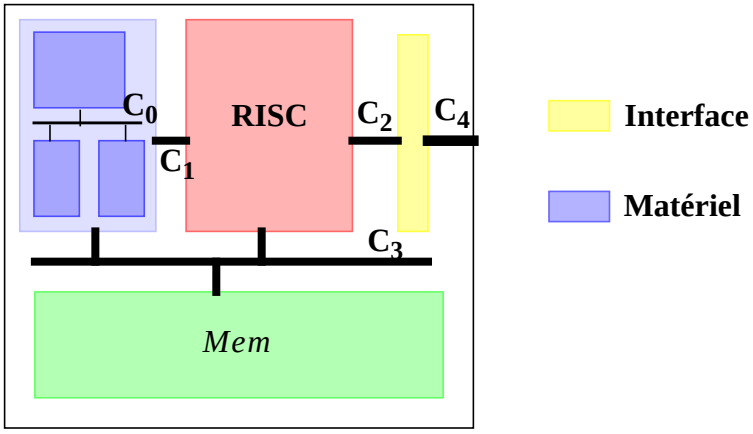


Figure 49: exemple de l'architecture d'un SoC

• Afin de mesurer le paramètre α_n , un outil d'évaluation de l'activité des signaux a été développé spécialement par ArexSys, dans le cadre de cette étude, et sera ultérieurement intégré dans l'outil ArchiMate. Cet estimateur calcule la distance de Hamming entre deux données consécutives présentent sur un bus aux instants *t* et *t+dt*, et réalise le cumul sur toutes les données transitant sur le bus.
La distance de Hamming entre 2 occurrences d'un signal correspond au nombre de bits qui diffèrent entre ces 2 réalisations.
Exemple de distance de Hamming pour un bus de 16 bits :
0001001001011001 --> 0001000101101001
La distance de Hamming entre ces deux valeurs est de 4.

• La capacité du bus *n* peut être vue comme le produit d'une capacité de référence et d'un coefficient de proportionnalité (par exemple, la capacité d'une connection est 100 fois plus grande que celle d'une porte logique pour une technologie CMOS 0.18 µm [47], et un facteur 100 peut être également admis entre la capacité d'un bus interne et celle d'une entrée/sortie du circuit [48]). Le modèle capacitif adopté est donc le suivant :

$$C_n = r_n \cdot C_{ref} \qquad \text{(B.13)}$$

où $\begin{cases} r_n, \text{ est le coefficient de proportionnalité} \\ C_{ref}, \text{ est la capacité de référence} \end{cases}$

Par ailleurs, le modèle précédent doit tenir compte du couplage inter-ligne du bus.
En effet, une transition AAAA -> 5555 (représentation hexadécimale) dissipe plus d'énergie qu'une transition 0000 -> FFFF à cause du couplage inter-ligne, alors que les distances de Hamming sont les mêmes (certains constructeur utilisent l'approximation suivante : le pire cas correspond à une transition AAAA -> FFFF (100%), alors une transition 0000 -> FFFF correspond à 95% du maximum, et le maintien d'une valeur correspond à 70% [49]). Ainsi, pour une estimation plus précise de la consommation, ce facteur est pris en compte dans la définition du coefficient r_n.

Selon [47], la capacité d'un bus peut être approximée comme suit :

$$C_n \approx (a_n + b_n \cdot k_n) \cdot C_{ref} \qquad \text{(B.14)}$$

$$\begin{cases} a_n \text{ est le facteur de proportionnalité entre } C_n \text{ et } C_{ref} \\ b_n \text{ est le surcoût capacitif d'une ligne supplémentaire} \\ k_n \text{ est le nombre de lignes du bus} \\ C_{ref} \text{ est la capacité de référence (capacité interne)} \end{cases}$$

La dissipation d'énergie de l'ensemble des bus, les bus étant à la même fréquence, est alors de la forme :

$$\begin{aligned} P_{COM} &\propto \alpha_0 \cdot C_0 + \alpha_1 \cdot C_1 + ... + \alpha_n \cdot C_n \\ &\propto \alpha_0 \cdot (a_0 + b_0 \cdot k_0) \cdot C_{ref} + \alpha_1 \cdot (a_1 + b_1 \cdot k_1) \cdot C_{ref} + ... + \alpha_n \cdot (a_n + b_n \cdot k_n) \cdot C_{ref} \\ &\propto \alpha_0 \cdot (a_0 + b_0 \cdot k_0) + \alpha_1 \cdot (a_1 + b_1 \cdot k_1) + ... + \alpha_n \cdot (a_n + b_n \cdot k_n) \qquad \text{(B.15)} \end{aligned}$$

L'utilisation de cette formule est un moyen simple et rapide d'obtenir des éléments de comparaison pour différentes configuration du système. (a_n et b_n sont des paramètres fixés par la technologie)

• Dans le flot de conception ArchiMate, l'évaluation de l'activité est réalisée par l'insertion de routines dans le modèle matériel C++ généré lors de la *synthèse matérielle*. Ces routines calculent des distances de Hamming sur les signaux transitant sur les canaux de communication.

III.4.4 Illustration sur l'exemple du ping pong : résultat et limitations

III.4.4.a) Résultat

Les traces des signaux p1top2 et p2top1 ainsi que les distances de Hamming sont fournis en annexe B.5.
Le tableau 23 présente les résultats obtenus.

Tableau 23: distance de Hamming cumulée expérimentale et théorique

signal	Hamming expérimentale	Hamming théorique
to_block2	33	17
to_block1	26	15

On s'aperçoit que la distance de Hamming calculée ne correspond pas à la valeur théoriquement attendu sur les canaux de communication. Ceci est dû au mode de fonctionnement du modèle SDL des signaux. En effet, à chaque signal est associé un identifiant du processus émetteur du dit signal. L'implémentation sous ArchiMate du canal supportant ce signal, pour un protocole "hand-shake" génère 2 signaux de contrôle, un pour la requête, l'autre pour l'acquittement, et un signal de données (figure 50).

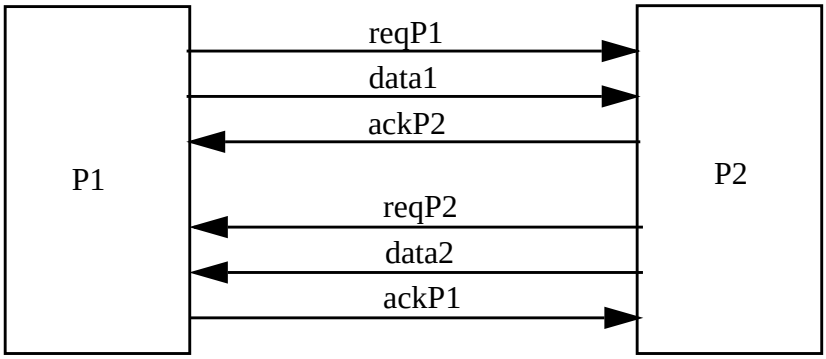


Figure 50: illustration d'une communication Hand-Shake

Pour l'émission d'une donnée par P1, le protocole de communication se déroule comme décrit sur la figure 51.

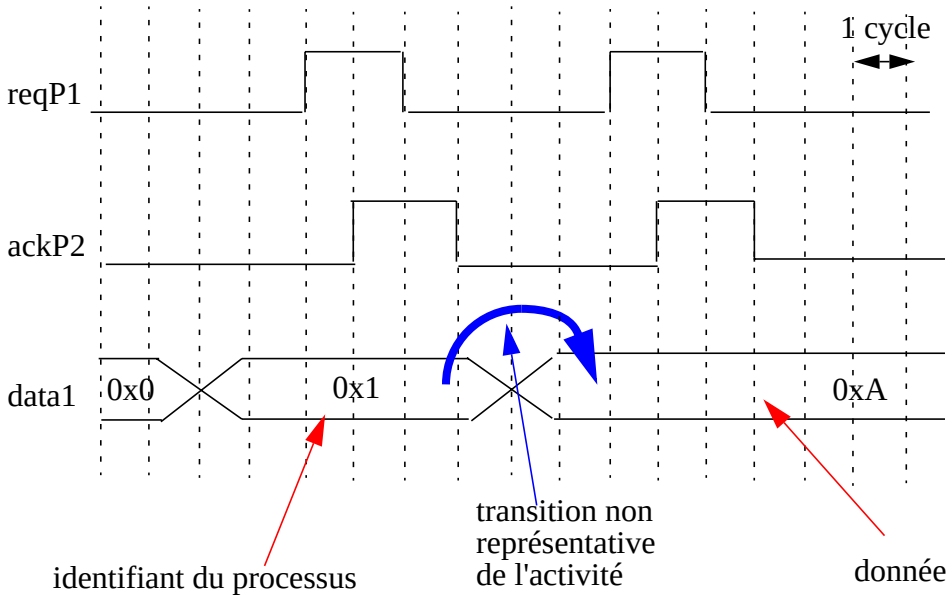


Figure 51: simulation du signal to_block2 sur l'exemple du ping-pong

L'envoie d'une donnée sur le canal *data1* est donc précédé d'une information de contrôle qui induit des transitions supplémentaires dont la part n'est pas négligeable sur cet exemple (50 % des transitions). De fait, ce protocole nécessite également l'émission de 2 requêtes par le processus P1.

III.4.4.b) Limitations

Les limitations sont de plusieurs ordres:
- la taille du bus est figée (aujourd'hui, bus de 32 bits uniquement) :
afin d'obtenir la flexibilité nécessaire à la spécification de l'architecture d'un système et de satisfaire aux contraintes de conception, il faut pouvoir ajuster la largeur des bus ainsi que la dynamique des données. Ce n'est aujourd'hui pas le cas dans l'outil ArchiMate mais la possibilité de spécifier la largeur des bus implémentés sera intégrée dans une prochaine version de l'outil.
- la sélection d'un mode de codage des données n'est pas prévue :
afin de faciliter l'exploration de différentes configurations du système, la proposition au concepteur de plusieurs mode de codage des données transitant sur un bus serait un plus (comme par

exemple le codage en bus inversé). En effet, l'intégration à l'outil de cette fonctionnalité automatiserait l'ajout du matériel supplémentaire nécessaire au codage / décodage des données.
- l'implémentation des protocoles n'est pas adaptée à une architecture "flot de données" :
en effet, on a vu qu'un signal de contrôle est envoyé sur un canal avant toute émission de donnée.
La possibilité de choisir entre ce mode de fonctionnement et l'implémentation de deux bus, un pour les contrôles, l'autre pour les données, étendrait les possibilités architecturales pour l'implémentation des inter-connections. L'utilisation de deux bus distincts conduit au comportement décrit en figure 52 sur l'exemple précédent. L'activité cumulée (donnée et contrôle) est alors dominée par les transitions sur le bus de donnée (pour l'exemple du ping pong, on obtiendrait une distance de Hamming cumulée de 18 sur le canal *to_bock2,* la distance de Hamming sur le bus de donnée représentant 95 %).

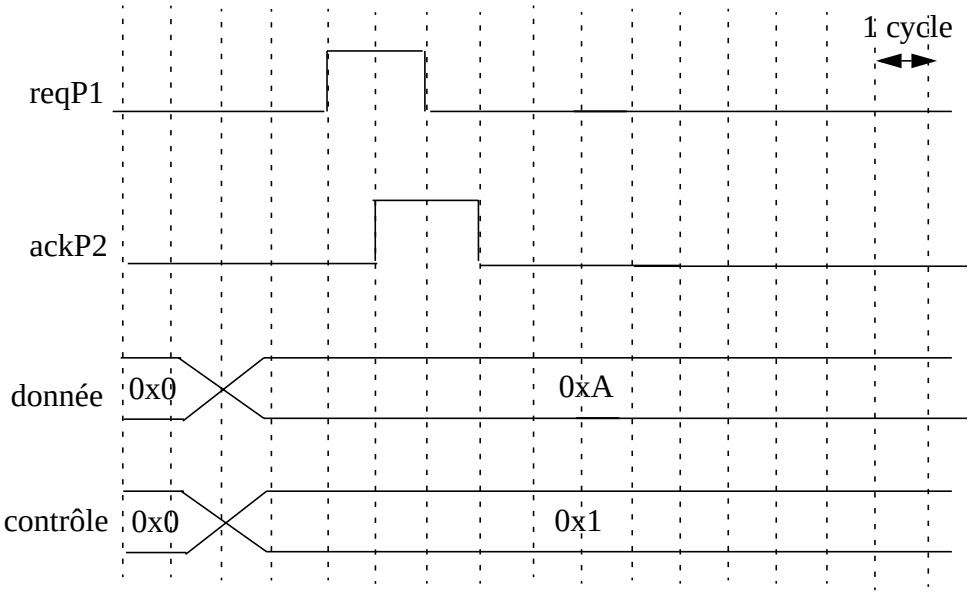


Figure 52: implémentation de deux bus : donnée + contrôle

III.5 Architecture du codeur H.263

Une description entièrement SDL du codeur a été réalisée, en voici le schéma blocs sous ObjectGeode.
Les blocs du schéma de la figure 53 peuvent être regroupés par traitements élémentaires du codage vidéo :
- estimation de mouvement : *blck_em_pict, blck_em_mb, blck_sad*
- transformée en cosinus discrète : *blck_transf*
- quantification : *blck_quantif*
- codage à longueur variable et formation du bitstream : *blck_vlc, blck_putbits*
- mémoire d'images : *blck_load*
- contrôle des stratégies de codage et séquencement du codage : *blck_newcoding, blck_codeintra, blck_codeinter*
- environnement : *blck_testbench*

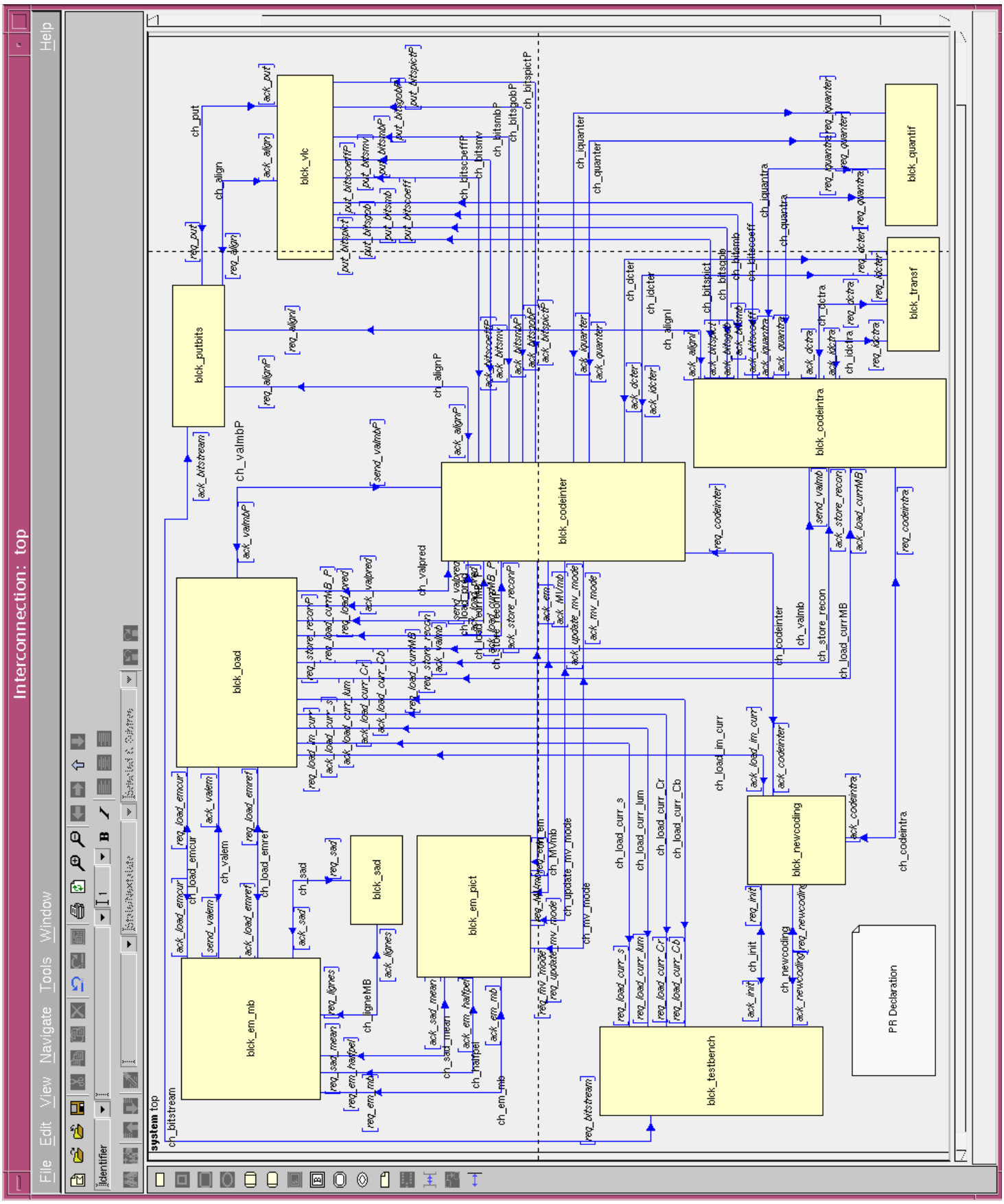


Figure 53: Spécification SDL du codeur vidéo

Outre le partitionnement au niveau système par blocs fonctionnels, certains blocs de traitement ont été ré-écrit. C'est le cas de l'estimateur de mouvement au demi-pixel qui nécessite l'interpolation de la luminance de l'image de référence. Dans la version C, cette interpolation est faite sur toute l'image et est stockée dans une mémoire image avant d'être exploitée par l'estimateur ce qui conduit à un surcoût inutile de mémoire (mémorisation d'environ 100kB supplémentaires pour une vidéo QCIF). Nous proposons dans la version SDL un calcul au vol des valeurs interpolées des pixels. De plus, et afin de réduire l'activité sur le canal entre la mémoire image et la mémoire de la fenêtre de recherche, la fenêtre de recherche est étendue de 1 pixel en hauteur et en largeur. L'interpolation est faite après l'estimation de mouvement en pixel entier. Elle nécessite une mémoire de seulement 1089 octets pour la fenêtre de recherche interpolée et conduit à 82500 interpolations au lieu de 76000 pour la version originelle. La taille de la mémoire nécessaire permet désormais de l'associer à l'opérateur d'estimation de mouvement, ce qui conduit à une forte réduction de l'activité sur le bus de la mémoire principale.

Amélioration supplémentaire

En partant de la constatation que les fenêtres de recherche pour des MBs successifs sont recouvrantes (figure 54), l'activité sur le bus d'écriture dans la mémoire de la fenêtre de recherche peut être grandement réduite. Plusieurs techniques de parcours de l'image ont déjà fait l'objet de publications [50][51], nous présentons une possibilité.

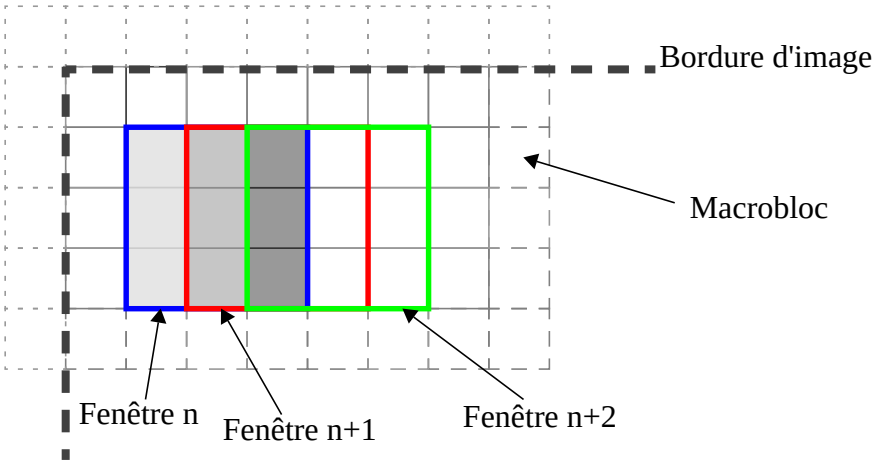


Figure 54: recouvrement des fenêtres de recherche consécutives

Au lieu de charger à chaque nouvelle recherche d'un vecteur mouvement 48*48 pixels de l'image de référence (sauf pour les bords de l'image), on ne charge que les 16 colonnes de 48 pixels non présentes dans la fenêtre de recherche précédente. La mémoire de la fenêtre de recherche peut alors être vue comme 3 bancs de mémoire de 16*48 pixels, un seul de ces bancs étant écrit à chaque nouvelle recherche (excepté pour les fenêtres de bord d'image). Cette approche permet de réduire fortement l'activité sur le bus mais au coût d'une gestion plus délicate des accès à la mémoire de la fenêtre de recherche. Une amélioration de cette configuration peut être obtenue en ajoutant un banc de 16 *48 pixels à la mémoire permettant d'écrire par anticipation (en parallèle à la recherche du vecteur mouvement du MB courant) les nouveaux pixels de référence.

Résultat

Le modèle matériel de ce système a été généré, cependant l’exécution du modèle C++ s'est avérée beaucoup trop longue et posant des problèmes de ressource de mémoire vive pour la compilation et de CPU pour l’exécution du modèle compilé. Nous avons donc décidé de valider

notre approche sur le sous-ensemble le plus critique du système de codage vidéo. Comme nous l'avons montré lors de l'étude sur la complexité algorithmique (cf. II.3.1), ce sous-ensemble correspond au bloc d'estimation de mouvement.

III.6 Analyse de l'EM

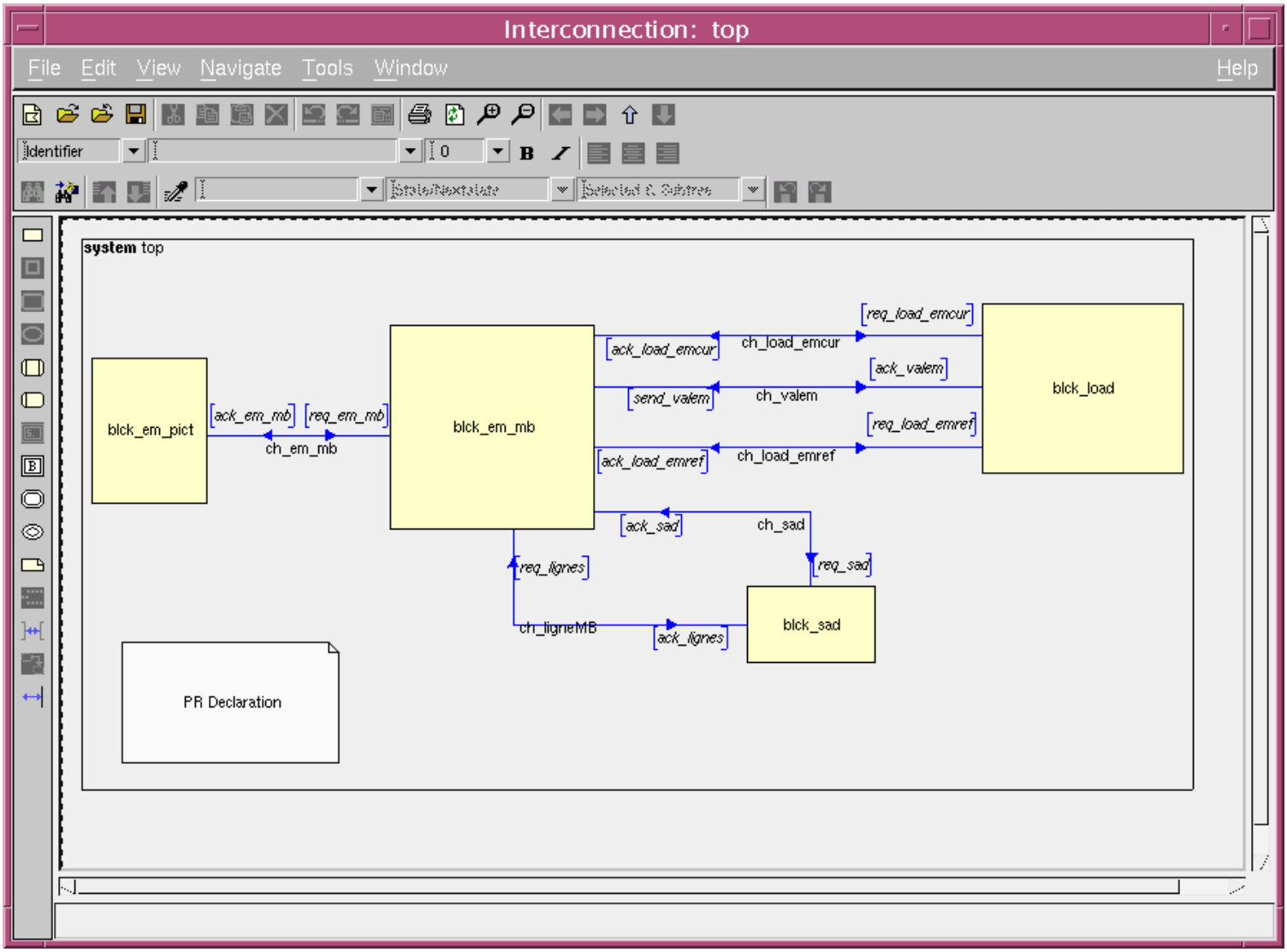


Figure 55: Schéma bloc de l'EM: spécification SDL

Le sous-système présenté figure 55 reprend les blocs de traitements déjà spécifiés pour l'estimation de mouvement dans le codeur complet.
Les figures 56 et 57 ci-après illustrent les étapes de conception suivies sous ArchiMate pour générer le modèle exécutable de l'estimateur de mouvement.
blck_em_pict : contrôle de la recherche des vecteurs mouvement (indice de macrobloc, ...), mémorisation des vecteurs mouvement.
blck_em_mb : gère toutes les opérations au niveau MB, mémorisation de la fenêtre de recherche et du MB courant
blck_sad : opérateur de calcul de la somme des valeurs absolues des différences.
blck_load : mémoire de l'image courante et de l'image de référence.

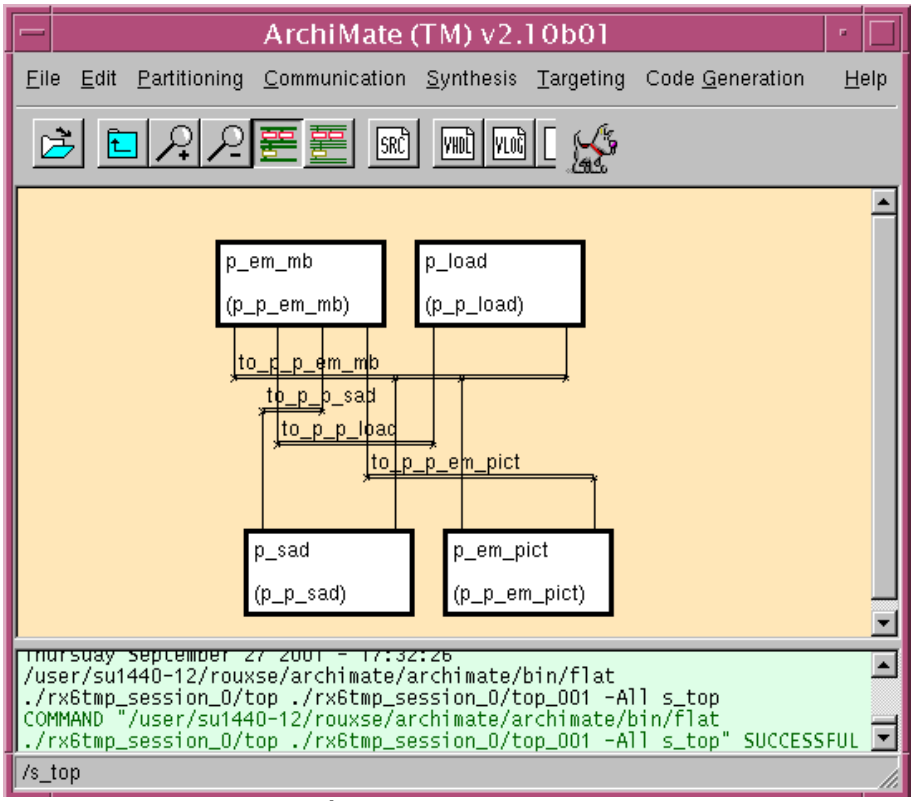


Figure 56: schéma des blocs structurels de l'EM

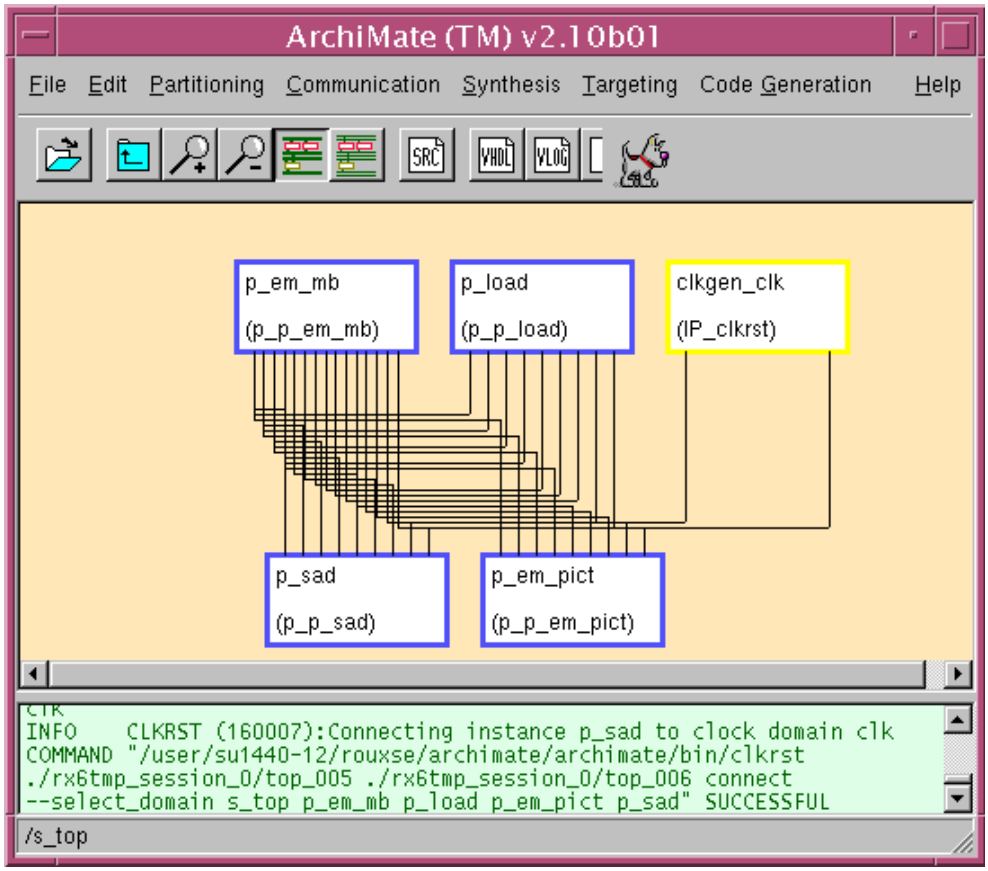


Figure 57: architecture "matérielle" de l'EM

Le tableau 24 fournit les taux d'activité par ligne et par cycle sur les canaux :
- *load_to_emmb* sur lequel transitent les pixels de la fenêtre de recherche et du MB courant chargés à partir de la mémoire de l'image de référence et de l'image courante,
- *sad_to_emmb* retournant la valeur de la distorsion courante, calculée par l'opérateur de calcul de distorsion (*sad*), au bloc de contrôle de la recherche du vecteur mouvement (*emmb*),
- *emmb_to_sad* sur lequel transitent les pixels de la fenêtre de recherche et du MB courant chargés dans les registres de l'opérateur de calcul de distorsion.
Ces mesures sont effectuées pour la recherche exhaustive avec échappement du vecteur mouvement d'un MB. Le nombre total de cycles pris en compte comprend le chargement de la fenêtre de recherche à partir de la mémoire image, le parcours en spirale de cette fenêtre et les opérations associées à la recherche du minimum de distorsion. De plus, bien que tous les canaux correspondent à 32 lignes, nous ne considérons que les lignes nécessaires à la transmission des données, c'est à dire, 8 bits pour les canaux *load_to_emmb* et *emmb_to_sad*, et 16 bits pour le canal *sad_to_emmb*.

Tableau 24: taux d'activité

	load_to_ emmb	sad_to_emmb	emmb_to_sad
Distances de Hamming cumulées	0.0012	0.0002	0.0223

Comme nous l'avons vu précédemment le taux d'activité sur un canal (ou bus) correspond à la transmission d'un signal de contrôle suivi d'un certain nombre de données. Le nombre de données envoyées en série est spécifié au niveau SDL. Par exemple, nous avons choisi d'envoyer les pixels par paquet de huit sur le canal *load_to_emmb*, alors qu'un envoie en série de 32 pixels (16 du MB courant puis 16 de la fenêtre de recherche) a été retenu pour le canal *emmb_to_sad*.
Ces choix influent plus ou moins fortement sur la part d'activité due aux signaux de contrôle sur un canal donnée. En effet, dans le cas présent, on constate que l'activité due aux signaux de contrôle représente 40 % de l'activité totale sur le bus *load_to_emmb*, alors qu'elle représente 20 % de l'activité totale sur le bus *emmb_to_sad*.
Dans le cas où deux bus serait implémentés (contrôle + donnée), les taux d'activité serait les suivants :
- pour le bus de donnée *load_to_emmb* : 0.0007
- pour le bus de donnée *emmb_to_sad* : 0.0185
Ces taux d'activité conduisent à une première vision sur la répartition de l'activité entre les bus du système et orientent a priori les efforts d'optimisation. Cependant, nous ne pouvons pas conclure sur la consommation du système car, comme nous l'avons vu, les capacités des deux bus peuvent être très différentes.
En effet, dans le cas de l'estimateur de mouvement, suivant que la mémoire de l'image de référence et de l'image courante soit interne ou externe le rapport capacitif entre le bus *load_to_emmb* et *emmb_to_sad* varie fortement.
On peut prendre, par exemple, un facteur 10 pour le premier cas (mémoire interne) et un facteur 100 pour le deuxième (mémoire externe).

Si $C_{load} = 10 \cdot C_{sad}$

alors $P_{COM} = P_{load} + P_{sad}$

$$P_{COM} \propto \alpha_{load} \cdot C_{load} + \alpha_{sad} \cdot C_{sad} \qquad (B.16)$$

$$P_{COM} \propto 0.007 + 0.0185$$

$\begin{cases} (C_{load}, P_{load}, \alpha_{load}) \text{ sont la capacité, la puissance dissipée et l'activité de } load\_to\_emmb \\ (C_{sad}, P_{sad}, \alpha_{sad}) \text{ sont la capacité, la puissance dissipée et l'activité de } emmb\_to\_sad \end{cases}$

Dans ce cas, la consommation des communications est dominée par l'activité sur le bus *emmb_to_sad*.

Si $C_{load} = 100 \cdot C_{sad}$

alors $P_{COM} \propto 0.07 + 0.0185 \qquad (B.17)$

Dans ce cas, la consommation des communications est dominée par l'activité sur le bus *load_to_emmb*.

Remarques

• On constate que le taux d'activité sur le bus *load_to_emmb* est très faible, car il est calculé par rapport au nombre total de cycles nécessaire à la recherche d'un vecteur mouvement. La recherche exhaustive d'un vecteur mouvement étant dominée par les calculs de distorsion (94 % des cycles), ce faible taux est donc normal. Toutefois, si on ne considère que le nombre de cycles dû au chargement du MB courant et de la fenêtre de recherche, on obtient un taux d'activité de 0.011 par bit et par cycle. Ce taux qui devrait être similaire au taux d'activité obtenu sur le bus *emmb_to_sad*, du fait que les mêmes données transitent sur les deux bus, est plus faible. Cette différence est due à une meilleure corrélation entre deux données émises sur le bus *load_to_emmb* que sur le bus *emmb_to_sad*. En effet, sur le bus *load_to_emmb*, tous les pixels du MB courant puis tous ceux de la fenêtre de recherche sont transmis consécutivement alors qu'une succession de 16 pixels du MB courant puis 16 pixels de la fenêtre de recherche sont transmis sur le bus *emmb_to_sad* conduisant à une moins bonne corrélation des données.

• Il est important de pouvoir estimer les activités lorsque les données transitant sur les bus sont fortement corrélées et qu'il n'existe pas de modélisation simple. En effet, si on suppose pour notre étude que les valeurs de pixels sont aléatoires, on peut prendre comme hypothèse que l'activité d'une ligne du bus est de 1/2 (pour un bus 8 bits, la distance de Hamming moyenne est donc de 4), ce qui conduit pour la recherche exhaustive du vecteur mouvement d'un MB (vecteur mouvement $\in [-15, 15]$), à un taux d'activité de 0.0019[1] sur le canal *load_to_emmb* (au lieu de 0.0007), et à un taux d'activité de 0.141[1] sur le canal *emmb_to_sad* (au lieu de 0.0185).

• Le taux d'activité sur le bus d'écriture dans les mémoires du MB courant et de la fenêtre de recherche peut être davantage réduit par l'utilisation de l'une des techniques de codage intro-

1. sans considérer l'activité due aux signaux de contrôle

duite au paragraphe III.4.1.
Le codage en bus inversé n'est pas adapté au cas présent car la forte corrélation existant entre les pixels conduit à très peu de changement sur le bus. La distance de Hamming entre deux valeurs consécutives est par conséquent rarement supérieure à la moitié de la largeur du bus et l'activité du bus ne peut donc pas être grandement réduite par la technique du bus inversé.
En revanche, précisément du fait de la nature particulière des pixels, l'utilisation d'un dictionnaire de valeurs (cf. III.4.1) serait sans doute efficace.

III.7 Conclusion

Nous avons établi dans cette partie, une méthodologie d'exploration d'architecture pour les SoCs embarqués. Cette méthodologie facilite l'adéquation algorithme - architecture grâce à la rapidité des itérations. Cette efficacité est atteinte par l'utilisation d'outil de spécifications (algorithmique et architecturale) de mise en oeuvre simple et par une analyse effectuée à un haut niveau d'abstraction (complexité algorithmique et performance de l'architecture système : activité sur les bus, charge des coeurs de processeur, ...).
Notre approche a ensuite été appliquée à l'exemple du codeur vidéo, ce qui a révélé certaines limitations (spécification des bus : largeur, protocole, codage ; architecture orientée flot de données,...). Toutefois, afin de valider l'outil d'analyse de l'activité des signaux, co-développé avec la société ArexSys, nous avons étudié le bloc de traitement le plus critique révélé par l'étude algorithmique : l'estimation de mouvement. Les taux d'activité mesurés sur les bus ainsi que le modèle capacitif des bus (fixé par la technologie [59]) ont permis d'illustrer l'influence de l'architecture des bus sur la consommation.

IV. Bibliographie

[1] Codec H.263+, University of British Columbia, Image Processing Laboratory, 1997, http://www.ee.ubc.ca/image

[2] 'Video coding for low bit rate communication', Serie H : Audiovisual and multimedia systems, infrastructure of audiovisual services – coding of moving video, ITU-T, recommendation H.263, Février 1998.

[3] Digital cellular telecommunication system (phase 2+) (GSM), 'Multiplexing and multiple access on the radio path (GSM 05.02 version 6.4.1, release 1997)', document ETSI EN 300 908 V6.4.1.

[4] Report on UMTS, 'Final Draft 2.5 enabling UMTS (3G) services and applications', No. 11, Report from the UMTS Forum, Juillet 2000.

[5] T. R. Gardos, 'Video codec testmodel, near term', ITU-T, study group 16, video coding experts group, document Q15-D-65d1, Finland, Avril 1998.

[6] J.Ribas-Corbera et S. Lei, 'Rate control for low delay video communications', ITU, study group 16, video coding experts group, document Q15-A-20, Portland, Juin 1997.

[7] J.L. Barron et al., 'Systems and experiment performance of optical flow techniques ', Int. Journal of Computer Vision, pp 43-76, 1994.

[8] B.D. Lucas et T. Kanade, 'An iterative image registration technique with an application to stereo vision', Proc. of the 7th Int. Joint Conf. on Artificial Intelligence, pp 121-130, 1981.

[9] B.K.P. Horn et B.G. Schunck, 'Determining optical flow', Artificial Intelligence, Vol. 17, pp 185-204, 1981.

[10] H.-H. Nagel, 'Displacement vectors derived from second order intensity variations in image sequences', Computer Vision, Graphics and Image processing, Vol. 21, pp85-117, 1983.

[11] A.N. Netravali et J.D. Robbins, 'Motion compensated television coding: Part I', Bell Syst. Tech. J., Vol. 58, pp 631-670, Mars 1979.

[12] T. Koga et al., 'Motion-compensated interframe coding for video confrencing', Proc. Nat. Telecommun. Conf., New Orleans, pp G5.3.1-G5.3.5, Novembre 1981.

[13] L.-M. Po et W.-C. Ma, 'A novel four step search algorithm for fast block matching estimation', IEEE Trans. Circuits and System for Video Tech., Vol. 6, No 3, pp 313-317, 1996.

[14] J.R. Jain et A.K. Jain, 'Displacement measurement and its application in interframe image coding', IEEE Trans. Commun., Vol. 29, pp 1799-1808, Décembre 1981.

[15] A. Puri, H.-M. Hang et D.L. Schilling, 'An efficient block-matching algorithm for motion compensated coding', Proc. IEEE Int. Conf. Acoust. Speech, Signal Processing, pp 25.4.1-25.4.4, Dallas, Avril 1987.

[16] M. Ghanbari, ‘The cross search algorithm for motion estimation’, IEEE Trans. Commun., Vol. 38, pp 950-953, Juillet 1990.

[17] L.-K. Liu et E. Feig, 'A block based gradient descent search algorithm for block motion estimation in video coding', IEEE Trans. on Circuits and System for Video Tech., Vol. 6, No. 4, pp 419-422, 1996.

[18] J. Lu et M.L. Liou, 'A simple and efficient search algorithm for block matching motion estimation', Vol. 7, No. 2, pp 429-433, Avril 1997.

[19] J.M. Jou, P.-Y. Chen et J.-M. Sun, 'The gray prediction search algorithm for block motion estimation', IEEE Trans. on Circuits and System for Video Tech., Vol. 9, No 6, pp 843-848, Septembre 1999.

[20] B. Liu et A. Zaccarin, 'New fast algorithms for the estimation of block motion vectors', IEEE Trans. Ciruits and System for Video Tech., Vol. 3, No. 2, pp 148-157, Avril 1993.

[21] H.-M. Hang, Y.-M. Chou et S.-C. Cheng, 'Motion estimation for video coding standards', Journal VLSI Signal Processing, Vol. 17, pp 113-136, 1997.

[22] S. Zhu et K.-K. Ma, 'A new diamond search algorithm for fast block matching motion estimation', IEEE Trans. on Image Processing, Vol. 9, No. 2, pp 287-290, Février 2000.

[23] Y.Q. Shi et X. Xia, 'A thresholding multiresolution block matching algorithm', IEEE Trans. on Circuits and System for Video Tech., Vol. 7, No. 2, pp 437-440, Avril 1997.

[24] K. Lengwehasatit et A. Ortega, 'Probabilistic partial distance fast matching algorithms for motion estimation', IEEE Trans. on Circuits and System for Video Tech., Vol. 11, No. 2, pp 139-152, Février 2001.

[25] Y.-S. Chen Y.-P. Hung et C.-S. Fuh, 'Fast block matching algorithm based on the winner update strategy', IEEE Trans. on Image Processing, Vol. 15, Mars 2001.

[26] M.M. Mizuki, U.Y. Desai, I. Masaki et A. Chandrakasan, 'A binary block matching architecture with reduced power consumption and silicon area requirement', Int. Conf. On Audio, Speech and Signal Proc. (ICASSP), pp 3248-3251, 1996.

[27] 'UltraSparc IIi CPU, Data sheet', Sun Microsystems, Mai 1999.

[28] iprof : 'Intruction Level Profiler V0.4.1', 1998, Technical University of Munich, Institute for Integrated Circuits, ftp.lis.e-technik.tu-muenchen.de/pub/iprof

[29] Outil de co-design CoWare N2C : http://www.coware.com

[30] Outil d'exploration d'architecture, ArchMate, et de co-simulation, CosiMate : http://www.arexsys.com

[31] Laboratoire Technique de l'Informatique et de la Microélectronique pour l'Architecture d'ordinateurs (TIMA) : http://tima.imag.fr

[32] ObjectGeode : http://telelogic.com/products/objectgeode

[33] E. Casseau, S. Pillement, D. Chillet, P. Coussy, E. Martin, G. Savaton, O. Sentieys, S. Roux, 'Design and synthesis of behavioral level virtual components', 11th IFIP international conference on very large scale integration (VLSI-SOC), Montpellier, Décember 2001.

[34] 'Specification and Description Language (SDL)', Serie Z : Language and general software aspects for telecommunication systems, ITU-T recommandation Z.100, Novembre 1999.

[35] C.T. Hsieh, M. Pedram, G. Mehta et F. Rastgar, 'Profile driven program synthesis for evaluation of system power dissipation', Proc. IEEE 34th Design AutomationConf. (DAC), pp 576-581, 1997.

[36] T. Simunic, L. Benini et G. De Micheli, 'Cycle accurate evaluation of energy consumption in embedded systems', Proc. Design Automation Conf. (DAC), 1999.

[37] V. Tiwari, S. Malik, A. Wolfe et M. Lee, 'Instruction level power analysis and optimization of software', Journal VLSI Signal Processing Systems, Vol. 13, No. 2-3, 1996

[38] N. Vijaykhrishnan et al., 'Energy driven integrated hardware-software optimizations using SimplePower', Proc. 27th Int. Symp. Computer Architecture, pp 24-30, 1999.

[39] F.N. Najm, 'A survey of power estimation techniques in VLSI circuits', IEEE Trans. on VLSI Systems, Vol. 2, No. 4, pp 446-455, Décembre 1994.

[40] A.P. Chandrakasan et R.W. Brodersen, 'Minimizing power consumption in digital CMOS circuits', Proceedings of IEEE, Vol. 83, No. 4, pp 498-523, Avril 1995.

[41] J.D. Meindl, 'Low power microelectronics: retrospect and prospect', Proceedings of IEEE, Vol. 4, No. 4, pp 619-635, Avril 1995.

[42] T.D. Givargis, J. Henkel et F. Vahid, 'Interface and cache power exploration for core based embedded system design', ICCAD 1999.

[43] M.R. Stan et W.P. Burleson, 'Bus invert coding for low power I/O', IEEE trans. VLSI, Vol. 3, Mars 1995.

[44] E. Musoll, T. Lang et J. Cortadella, 'Working zone encoding for reducing the energy in microprocessor address buses', IEEE Trans. on VLSI Systems, Vol. 6, No. 4, pp 568-571, Décembre 1998.

[45] L. Benini, G. De Micheli, E. Macii, M. Poncino et S. Quer, 'Power optimization of core based systems by address bus encoding', IEEE Trans. on VLSI Systems, Vol. 6, No. 4, Décembre 1998.

[46] M.R. Stan et W. P. Burleson, 'Limited weight codes for low power I/O', Int. Workshop on Low Power Design, Avril 1994.

[47] N.H.E. Weste et K. Eshraghian, 'Principles of CMOS VLSI design', Reading, MA: Addison Wesley, 1998.

[48] T.D. Givargis, F. Vahid et J. Henkel, 'Evaluating power consumption of parameterized cache and bus architectures in system on a chip designs', IEEE Trans. on VLSI Systems, Vol. 9, No. 4, pp 500-508, Aout 2001.

[49] C. Turner, 'Calculation of TMS320LC54x power dissipation', Texas Instruments application report, Digital Signal Processing Solutions-Semiconductor Group, SPRA164, Juin 1997.

[50] G.L. Tzeng et C.Y. Lee, 'An efficient memory architecture for motion estimation processor design', Proc. Int. Conf. Audio, Speech and Signal Processing (ICASSP), Detroit, Mai 1995.

[51] Y.-H. Yeh et C.-Y. Lee, 'Cost effective VLSI architectures and buffer size optimization for full search block matching algorithms', IEEE Trans. on VLSI Systems, Vol. 7, No. 3, pp 345-358, Septembre 1999.

[52] Analog Devices, Blackfin DSP, 'Preliminary technical data : ADP-21532, Septembre 2001, http://www.analog.com

[53] H.-M. Jong, L.-G. Chen et T.-D. Chiueh, 'Parallel architectures for 3 step hierarchical search block matching algorithm', IEEE Trans. on Circuits and Systems for Video Tech., Vol. 4, No. 4, pp 407-415, Aout 1994.

[54] J.-M. Chang et M. Pedram, 'Energy minimization using multiple supply voltage', Int. Symp. on Low Power Electronics and Design (ISLPED), Monterey, pp 157-162, 1996.

[55] ‘TMS320C55x DSP functional overview’, Texas Instruments, No. SPRU312, Juin 2000, http://www.ti.com

[56] J.M. Rabaey et M. Pedram, 'Low power design methodologies', Kluwer Academic Publishers, 1995.

[57] S. Kim et J. Kim, 'Low power data representation', Electronics letters, Vol. 36, No. 11, pp 958-959, Mai 2000.

[58] B. Kapoor, 'Low power memory architectures for video applications', Proc. 8th Great Lakes Symp. VLSI, pp 2-7, 1998.

[59] International Technology Roadmap for Semiconductor (ITRS) : http://public.itrs.net

Chapitre C : Amélioration du codage vidéo pour la visiophonie

Les recherches sur la segmentation de scène vidéo en objets d'intérêt ont pris depuis quelques années un essor tout particulier avec l'apparition des normes MPEG-4 [1] et MPEG-7 [2]. La norme MPEG-4 intègre en effet la notion d'objet dans le codage vidéo ce qui suppose l'extraction préalable des objets vidéos et motive donc de nombreuses recherches sur la segmentation. Quant à MPEG-7, il s'agit d'une norme spécifiant les méthodes et moyens d'indexation et de recherche par contenu vidéo. Elle se doit donc de proposer des outils de segmentation. Quitte à faire cette segmentation, pourquoi ne pas l'exploiter à l'intérieur des codeurs vidéo et de manière à prendre en compte le système visuel humain [3] ?

C'est ce que nous proposons de faire dans ce chapitre, sans perdre de vue les contraintes de l'embarqué.

Dans la première partie, on s'intéresse à la détection d'un objet propre à la visiophonie : le visage. Un tour d'horizon non exhaustif des différentes méthodes de détection du visage est tout d'abord réalisé. Puis, nous présentons un nouvel algorithme de détection / localisation du visage adapté aux contraintes de l'embarqué.

La deuxième partie traite de l'intégration de notre algorithme dans un schéma de codage H.263 et de l'apport des techniques de codage orientées visage pour l'implantation de la visiophonie sur un terminal mobile.

Enfin, la troisième partie introduit l'architecture retenue pour le traitement de détection du visage et propose différents axes de recherche pour la réalisation du système de codage complet basé sur ce nouvel algorithme.

I. Détection du visage

La détection de visage à l'instar de la reconnaissance de visage n'est réellement étudiée que depuis le début des années 90. Depuis, de nombreuses techniques de détection du visage ont vu le jour dans la littérature, utilisées comme pré-traitement à une reconnaissance de visage [4] ou à un marquage de traits caractéristiques du visage [5] ou comme traitement permettant l'asservissement d'un système aux mouvements d'un individu [6].

I.1. Etat de l'art

On distinguera quatre techniques de détection :
- la détection exploitant la géométrie et les caractéristiques du visage,
- la détection par mouvement,
- les approches connexionnistes,
- et les approches basées sur des modèles probabilistes de l'intensité et de la couleur des pixels de l'image.

I.1.1. Méthodes basées sur la géométrie et les caractéristiques du visage

Les premières recherches sur la détection de visage se sont basées sur l'extraction de traits caractéristiques. Avec l'adjonction de contraintes géométriques, il a été possible de définir le positionnement relatif de ces traits caractéristiques et ainsi d'obtenir une information sur la présence ou non d'un visage.

Ainsi, le système proposé par Govindaraju et al. [7], basé contour, modélise le visage comme étant un agencement de trois courbes (sommet, coté droit et coté gauche du visage). La détection de ces courbes et leur regroupement selon leur position relative permet de connaître la position du visage.
Soulignons que les méthodes basées contours les plus abouties reposent sur l'utilisation de contours actifs telles que celles proposées par Waite et Welsh [8], Craw et al. [9] ou Cootes et Taylor [10]. L'utilisation de contours actifs permet une adaptation aux variations de forme du visage et conduit ainsi à une meilleure délimitation du visage.
Une autre approche basée contours consiste en une extraction des contours de l'image et leur mise en adéquation avec une ellipse modélisant un visage (Jacquin et Eleftheriadis [11]).

Les méthodes exploitant les traits caractéristiques du visage sont très variées du fait du nombre de caractéristiques définissables et du nombre de techniques de détection. Les traits du visage les plus souvent retenus sont les yeux, les sourcils, la bouche, le nez, mais peuvent être d'un niveau de détail beaucoup plus fin. La détection est alors basée sur le fait que les caractéristiques d'un visage ont des positions relatives fixes (ou statistiquement modélisables).
L'une des premières réalisations est celle de Kanade en 1973 [12][13], orientée reconnaissance de visage, constituant un premier pas vers les techniques de détection et de localisation. Les contours de l'image sont tout d'abord extraits puis projetés selon des axes horizontaux et verticaux. Selon le modèle obtenu, le système est capable de retrouver le positionnement des yeux, de la bouche, du nez ainsi que les contours du visage et des cheveux. Si le profil obtenu ne satisfait pas au modèle de référence l'image est rejetée comme n'étant pas un visage.
La méthode la plus répandue pour extraire ces caractéristiques est de réaliser une corrélation entre l'image et un masque de caractéristiques génériques (Sumi et Otha [14], Zelinsky et Heinzmann [15]).

Une autre méthode utilise des filtres successifs pour l'extraction des paramètres du visage ce qui constitue une approche généralisée de la méthode précédente (Leung et al. [16], Graf et al. [17], Burl et Perona [18]). Cependant, l'ensemble des caractéristiques retenues ne constitue pas un modèle robuste et conduit à un grand nombre de fausses alertes. Toutefois, si les positions relatives des traits du visage détectés sont pris en compte par des techniques de comparaisons avec un modèle de déformation, alors la robustesse du système est grandement améliorée.
Dans [19], K.C. Yow et R. Cipolla décrivent un système de détection de traits du visage où les yeux, la bouche et le nez sont extraits par un filtrage en quadrature de phase et de regroupés selon leur probabilité de constituer un visage. Le problème de l'angle de vue est résolu par l'utilisation d'invariants affines augmentant la robustesse de leur approche.
Les approches basées sur l'extraction de traits du visage sont plus efficaces que les approches basées contours car elles analysent des structures locales de l'image conjointement à des modèles géométriques de visages. De plus, il s'agit d'approches ascendantes permettant d'écarter les fausses alertes à chaque étape de détection d'un nouveau trait de visage.

I.1.2. Méthodes basées sur le mouvement

L'utilisation de l'information de mouvement peut être un moyen simple pour mettre en œuvre une technique rapide de détection de visages dans une vidéo. Cette catégorie de systèmes suppose généralement que l'arrière plan de la scène vidéo est stationnaire et que le visage est toujours en mouvement. Dans ce cas, le visage peut être détecté par simple différence entre l'image courante et l'image précédente. Il semble évident que les hypothèses retenues sont trop fortes. C'est pourquoi l'information de mouvement n'est jamais utilisée seule pour la détection. On la trouve utilisée conjointement à l'information de couleur de peau dans [20] et couplée à un système basé sur la reconnaissance de traits caractéristiques dans [21] [22].
L'exploitation de l'information de mouvement oriente la détection vers des zones préférentielles et permet une forte réduction de la complexité de la détection. Cependant, les performances de ces systèmes sont fortement réduites lorsque la scène vidéo contient de nombreux objets en mouvement.

I.1.3. Méthodes basées sur les réseaux neuronaux

L'approche la plus répandue est sans doute celle utilisant un réseau de neurones pour classifier les pixels de l'image en tant que visage ou non-visage. Différentes architectures de réseaux et méthodes d'apprentissage ont été proposées par Cottrell et al. [23], Burel et Carel [24], Sung et Poggio [25], Lin et al. [26], Osuna et al. [27], [28] et on peut trouver dans [29] une présentation générale des méthodes basées sur les réseaux de neurone pour le traitement des visages humains.
H. A. Rowley et al. [30] présentent un système de détection reposant sur des réseaux neuronaux connectés en rétine. Les images subissent d'abord un pré-traitement par fenêtre glissante à travers un premier réseau permettant par exemple de déterminer l'inclinaison du visage afin de fournir au traitement multi-réseaux suivant une entrée plus facilement exploitable. Le système effectue ensuite un tri parmi les réponses des réseaux par l'intermédiaire d'un nouveau réseau,ou bien, il prend en considération l'ensemble des sorties afin d'achever la détection. Une version "rapide" est obtenue en utilisant des réseaux ayant des apprentissages différents et ne travaillant plus sur la base du pixel mais sur des zones recouvrantes de l'image. Une deuxième passe réalisée au niveau pixel par un autre réseau ne traite alors que les positions ayant une forte probabilité de contenir un visage.
Une technique similaire est proposée par Vaillant et al. [31], elle repose sur deux réseaux de neurones. Le premier indique s'il s'agit d'une image avec ou sans visage, il est appliqué à tous les pixels, mais reste rapide grâce à des connexions en rétine et des poids partagés. De plus, une grande partie des opérations effectuées pour un pixel peut être exploitée pour le pixel voisin. Le

deuxième réseau n'a pas d'apprentissage sur les images sans visage, il sert à la localisation précise du visage et indique si celui-ci est centré ou non.
Enfin, notons l'approche de S.Oka et al. [32] qui utilise un réseau de neurones récurrent composé d'un système oscillatoire non-linéaire et précédé d'un traitement d'extraction de particularités du visage telles que les yeux et la bouche.

I.1.4. Méthodes probabilistes

A. J. Colmenarez et al. [33] utilisent pour leur détection de visage les modèles de Markov discrets afin de modéliser le visage et l'arrière plan. La détection est faite en calculant le degré de ressemblance avec le modèle issu de la phase d'apprentissage.
B. Moghaddam et A.Pentland [34] proposent une approche basée sur l'estimation de densité de régions étendues en utilisant une décomposition en valeurs propres ("eigenface"). Deux types d'estimateurs reposant, soit, sur un modèle gaussien généralisé, soit, sur une combinaison de modèles gaussiens (loi mélange), servent à modéliser les probabilités. Pratiquement, une carte des probabilités est réalisée sur l'image par le parcours de celle-ci par une fenêtre d'observation et le calcul de ressemblance entre l'observation et la réalisation idéale. La position du visage correspond alors à la fenêtre donnant le meilleur score.
L'exploitation de l'information de couleur est également un moyen simple et efficace pour la détection de visage. De nombreuses recherches, Fleck et al. [35], Kjeldsen et Kender [36], ont démontré que la couleur peau est localisée dans une bande étroite de l'espace des couleurs. Cette information peut donc facilement être utilisée pour marquer les pixels de couleur peau. Les pixels marqués sont ensuite groupés et les régions résultantes lissées afin de connaître la position du visage.
D'autres, tels que Chen et al. [37], Dai et Nakano [38], utilisent avec succès une approche basée couleur.En effet, Chen et al.travaillent dans un repère de couleur régulier et utilisent un trieur logique aléatoire. De même, Dai et Nakano exploitent les informations de couleurs et de texture.
Retenons aussi l'approche de C. Golima et F. Meyer [39] qui proposent une détection en deux étapes, une étape d'initialisation marquant les pixels de couleur peau et vérifiant qu'une des régions de couleur peau construite correspond à un modèle de visage elliptique, et une étape de suivie basée sur une estimation/modélisation du mouvement, suivie d'une segmentation selon la couleur et éventuellement une analyse de forme.
Le système de M. Hunke et A. Waibel et al. [40][41] exploite également l'information de couleur de la peau et l'information de mouvement pour effectuer un suivi de visage en temps réel (10 à 12 extractions/s sur une station de travail).
Enfin, D. Comaniciu et P. Meer [42] effectuent une segmentation de l'image selon la couleur en prenant en compte le problème de sur ou sous-segmentation de l'image.
Pour plus d'information, [43] donne une synthèse des techniques de détection basées sur un modèle statistique de distribution de la couleur peau.

I.1.5. Autres méthodes

Dans [44], une décomposition de Gabor est utilisée pour la détection basée sur l'analogie avec le système visuel humain. Cette décomposition est orientée par un pré-traitement réalisant une détection de mouvement.
D'autres techniques font appel à des filtres morphologiques pour la construction de régions connexes pouvant être exploitées pour la détection de visage. Dans [45], on trouve une détection de visage basée sur une ligne de partage des eaux.

Ces techniques ne s'opposent pas obligatoirement mais peuvent être au contraire associées, comme on l'a vu pour les techniques basées mouvement et basées traits caractéristiques.

A ce titre, K. Sobottka et I. Pitas [46] proposent un système de détection utilisant l'information de couleur et la reconnaissance de traits caractéristiques. L'extraction des traits du visage fait appel à deux approches, la première est une variante de la méthode de la ligne de partage des eaux, la seconde est une analyse des minimum-maximum. De même, M. Kampmann et J. Ostermann [47] [48], utilisent une méthode d'analyse-synthèse basée sur une estimation des positions des yeux et de la bouche, des contours du visage et de la couleur de la peau.
On peut trouver, également, des méthodes probabilistes qui utilisent un réseau de neurones en post-traitement pour la classification des détections potentielles [49].
D'autres réalisations faisant appel aux techniques décrites ou à des variantes existent dans la littérature [50][51].

Nous venons de voir différentes techniques de détection/localisation du visage, cependant ces approches ne sont pas adaptées à la contrainte de faible complexité inhérente aux systèmes embarqués ou alors ne sont pas suffisamment robustes. En effet, les techniques de détection répertoriées restent d'une complexité trop importante incompatible avec le temps réel. De plus, elles possèdent souvent une phase d'apprentissage contraignante pour un usage multi-utilisateur en situation de mobilité.

I.2. Algorithme proposé

I.2.1. Principes

Notre objectif est de réaliser un système de détection de visage devant être intégré à un codeur vidéo embarqué pour offrir un service de visiophonie amélioré sur les terminaux mobiles. L'ergonomie du service de visiophonie mobile suppose qu'un seul visage est présent dans le champ de la caméra (mono-locuteur). Par ailleurs, la mobilité implique que l'arrière plan n'est pas connu a priori et peut contenir beaucoup de mouvement. Plus concrètement, la détection du visage doit permettre d'améliorer la qualité de la vidéo à débit constant, et sans augmenter la complexité du codeur.

Nous présentons dans ce chapitre, notre algorithme de détection du visage. Cet algorithme a été présenté à deux colloques scientifiques [52][53] et fait actuellement l'objet d'un dépôt de brevet France Télécom R&D [54]. Il exploite l'hypothèse que la texture de la peau est localisée dans une région étroite de l'espace des chrominances et se décompose en deux étapes :
- l'estimation du masque binaire des régions de couleur "peau" de l'image courante,
- la localisation et l'estimation d'un cadre circonscrit au visage.

L'algorithme présenté utilise uniquement les informations de chrominance et de position des pixels de l'image. Un pixel, p_i, correspond donc à la fois à un couple de chrominances, (Cr_i, Cb_i), et à des coordonnées dans l'image (x_i, y_i).
La première étape est basée sur une partition de l'espace des chrominances en 4 classes (notées C^i) dont une correspond à la peau. Chaque classe est construite en référence à un représentant, noté R^i de chrominances (Cr^i, Cb^i), calculé selon l'estimateur du centroïde, de manière adaptative.
A la fin de cette étape, le choix du représentant de la classe "peau" est effectué par comparaisons entre les représentants estimés et le représentant de référence de la texture peau. Puis, on attribue aux pixels de la classe "peau" la valeur '1', et la valeur '0' pour les pixels des autres classes. On construit ainsi le masque binaire des régions de couleur peau de l'image.

La deuxième étape consiste en une opération de filtrage non linéaire suivie d'une opération de

filtrage adapté. Le résultat est un rectangle circonscrit au visage du locuteur et positionné dans l'image.

Nous allons par la suite détailler ces deux étapes et justifier nos choix algorithmiques.

I.2.2. Détection du visage

I.2.2.a) Hypothèse de la couleur "peau"

Notre algorithme de détection repose donc sur l'hypothèse vérifiée que la couleur peau est localisée dans une région étroite de l'espace des chrominances. La figure 58 représente la répartition de la couleur peau dans l'espace des chrominances (Cr, Cb) que nous avons obtenu sur une centaine de visages d'individus de différentes origines (européenne, africaine, asiatique,...).

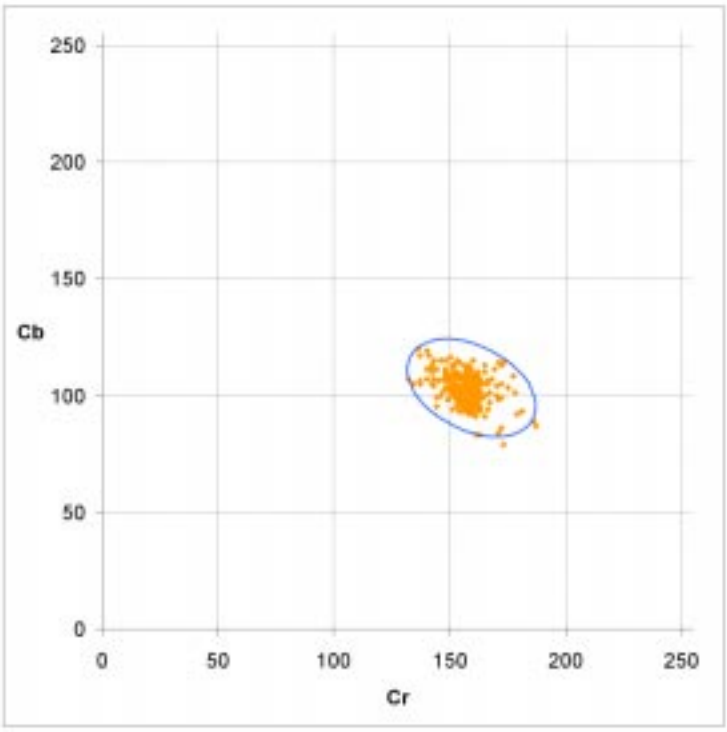


Figure 58: répartition de la couleur peau dans l'espace des chrominances (100 individus)

On peut constater que la couleur peau est localisée dans une région elliptique de l'espace des chrominances. C'est pourquoi, l'algorithme exploite cette information de chrominance sans prendre en compte la luminance (trop dépendante des conditions d'éclairage). Un avantage supplémentaire est une réduction considérable de la quantité de données à traiter du fait qu'on travaille uniquement sur les chrominances, et conduit donc à un algorithme de plus faible complexité.

I.2.2.b) L'approche basée classes

Afin de détecter les régions de couleur peau dans une image, on cherche à construire une partition de l'espace des chrominances de l'image en classes pour faire apparaître la zone d'intérêt dans l'image, en l'occurrence le visage.
La construction de ces classes est faite à partir de représentants préalablement fixés, dont le nombre est choisi de manière à faire correspondre, dans l'espace des chrominances, la classe

"peau" avec la région statistique décrite au paragraphe précédent. Par ailleurs, cette classe "peau" doit également correspondre à la région visage dans l'image. Comme nous le montrerons ci-après, un bon compromis est obtenu avec 4 représentants.

Concrètement, la construction d'une classe $C^l, (l \in [0, 3])$ associée au représentant R^l, pour une image donnée I_n, est réalisée selon la règle du plus proche voisin :

$$C^l = \{ p_i \in I_n : \left\| p_i - R^l \right\| = min_{j \in [0, 3]}(\left\| p_i - R^j \right\|) \} \qquad \text{(C.1)}$$

Afin de construire les 4 classes, il est donc nécessaire de choisir une norme pour le calcul des distances du pixel aux représentants.

i. Choix de la norme

La distance la plus souvent utilisée est la distance euclidienne aussi appelée norme 2 et définie comme suit pour un espace à 2 dimensions :

$$\left\| p_i - R^l \right\|_2 = \sqrt{(Cr_i - Cr^l)^2 + (Cb_i - Cb^l)^2} \qquad \begin{cases} p_i \in I_n \\ R^l \in \{ R^j : j \in [0, 3] \} \end{cases} \qquad \text{(C.2)}$$

Pour la construction des classes la norme 2 peut être ramenée à la somme des carrés des différences.

D'autres normes peuvent être utilisées pour déterminer la distance des pixels aux représentants.

La norme 1 [55][56] :

$$\left\| p_i - R^l \right\|_1 = \left| Cr_i - Cr^l \right| + \left| Cb_i - Cb^l \right| \qquad \begin{cases} p_i \in I_n \\ R^l \in \{ R^j : j \in [0, 3] \} \end{cases} \qquad \text{(C.3)}$$

la norme de Tchebyshev ou norme infinie :

$$\left\| p_i - R^l \right\|_\infty = sup(\left| Cr_i - Cr^l \right|, \left| Cb_i - Cb^l \right|) \qquad \begin{cases} p_i \in I_n \\ R^l \in \{ R^j : j \in [0, 3] \} \end{cases} \qquad \text{(C.4)}$$

et la distance de Mahalanobis [57][58] :

$$\left\| p_i - R^l \right\|_M = \sqrt{(p_i - R^l)^t \cdot C^{-1} \cdot (p_i - R^l)} \qquad \begin{cases} p_i \in I_n, \text{ dans l'espace (Cr,Cb)} \\ R^l \in \{ R^j : j \in [0, 3] \} \end{cases} \qquad \text{(C.5)}$$

C^{-1} est la matrice inverse de covariance de la classe

$(p_i - R^l)^t$ est la matrice transposée de $(p_i - R^l)$

Cette dernière norme pondère les composantes selon un degré d'importance pré-défini. Si C^{-1} est la matrice identité alors on retrouve la norme Euclidienne.

Ces normes sont illustrées en figure 59.

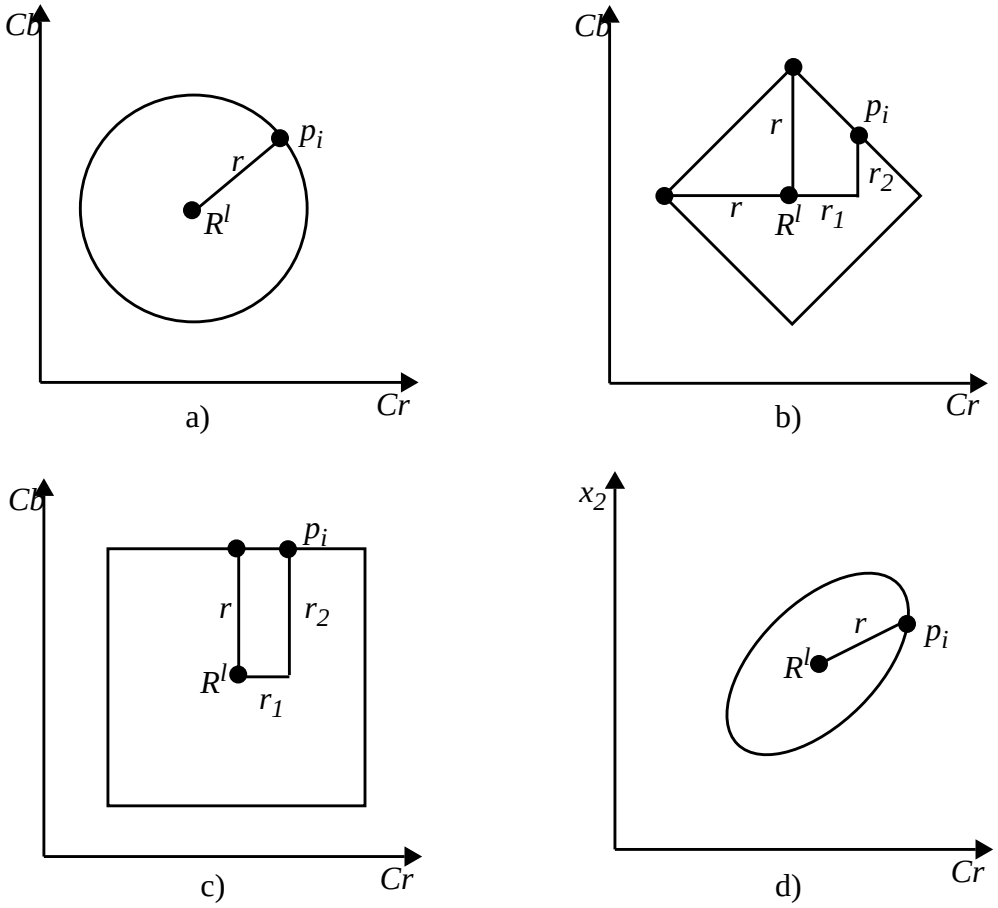


Figure 59: différentes normes et leur r-équidistants. a) Euclidienne b) Norme 1 c) Tchebytchev d) Mahalanobis

Nous choisissons maintenant de nous intéresser à la partition de l'espace des chrominances en 4 classes pour la norme 1 et la norme 2.

norme 1

La figure 60 illustre la partition de l'espace des chrominances pour la norme 1 en quatre classes dont les représentants sont $\{R^0, R^1, R^2, R^3\}$ (R^0 correspond à la classe de texture peau). L'ellipse représente la région statistique C^{ref} dans laquelle se trouve localisée la couleur peau. La zone blanche correspond à une région équidistante de R^1 et R^2.

Les inconvénients de cette norme sont la grande sensibilité des frontières vis à vis de la position des représentants et la difficulté de faire coïncider une classe avec la forme elliptique de C^{ref}. De plus, l'existence de larges zones de l'espace des chrominances équidistantes pose le problème de l'affectation des pixels à une classe plutôt qu'une autre.

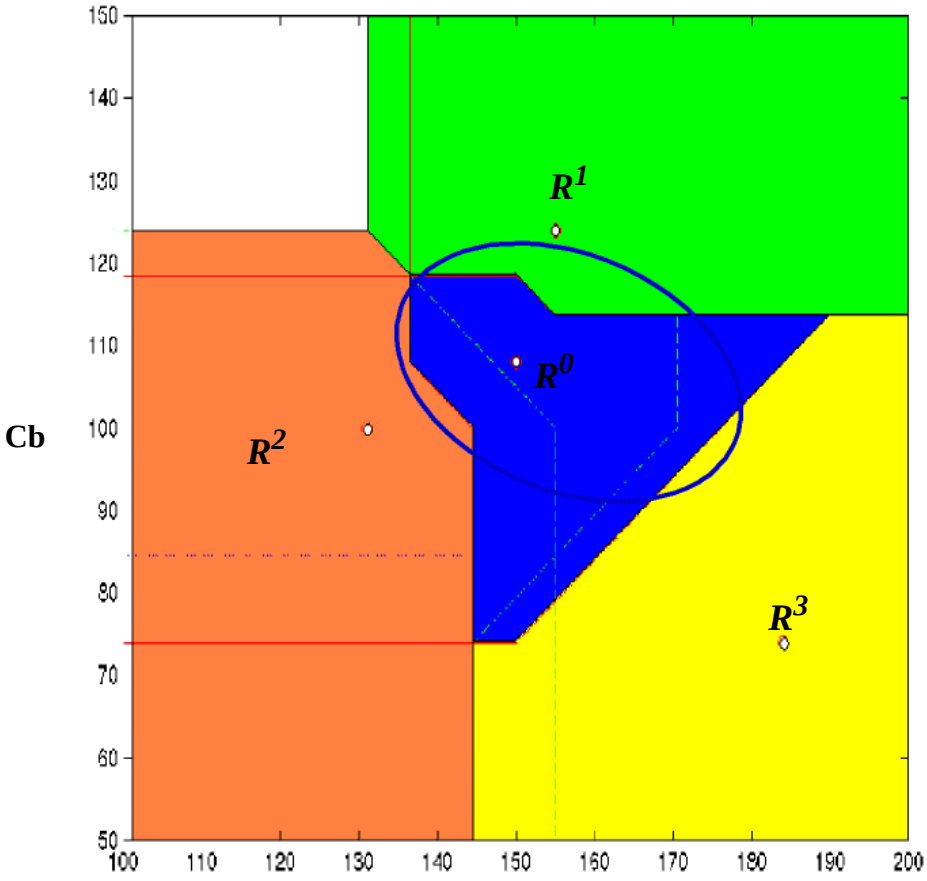


Figure 60: partition de l'espace des chrominances selon la norme 1

Le partitionnement d'une image selon les quatre classes définies ci-dessus est illustré sur les séquences Foreman et Akiyo (figure 61).

a) Foreman **b) Akiyo**

Figure 61: partition selon la norme 1 d'images contenant un visage

norme 2

Les quatre représentants étant fixés (de manière à révéler la classe "peau"), la partition de l'espace des chrominances obtenue selon la norme 2 est illustrée figure 62.

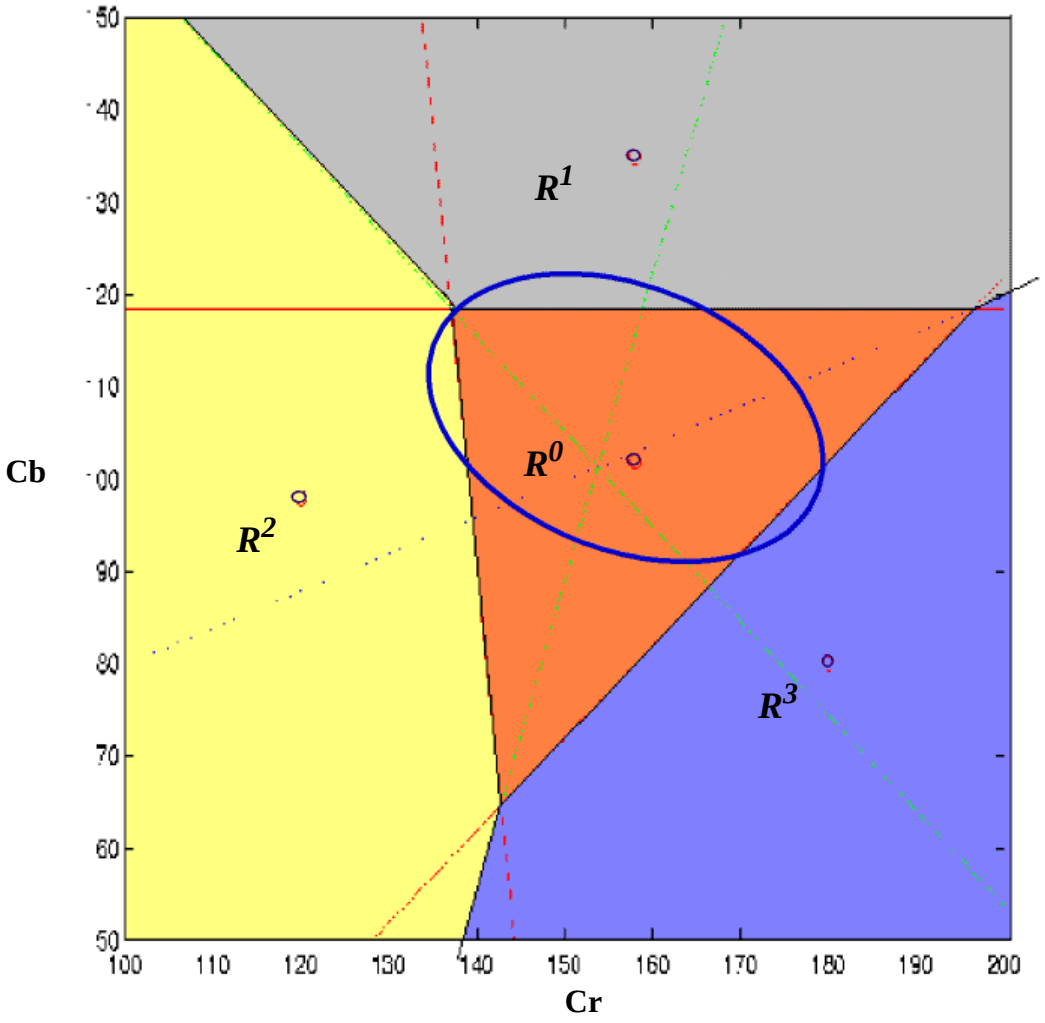
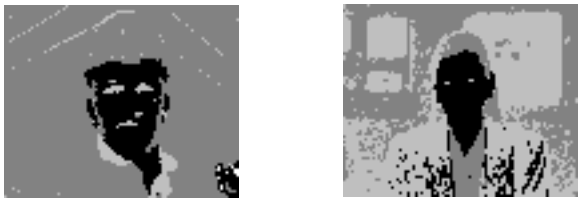


Figure 62: partition de l'espace des chrominances selon la norme 2

La figure 63 donne un exemple de partitionnement selon les classes pour la norme Euclidienne. Intuitivement, on comprend que le découpage selon cette norme conduit à une détection plus précise, moins sujette à des variations non linéaires.



a) Foreman **b) Akiyo**

Figure 63: partition selon la norme 2 d'images contenant un visage

Soulignons que la partition obtenue à l'aide de la règle du plus proche voisin est optimale lorsqu'on choisit, comme mesure de distorsion, l'erreur quadratique [59]. Et, inversement, si l'on se donne la partition, il est possible d'en déduire les meilleurs représentants. En effet, on montre que l'estimateur du centroïde est l'estimateur optimal. Cependant, si on ne connaît ni les représentants, ni la partition, il n'existe pas de solution optimale simple. Heureusement, l'algorithme

de Lloyd-Max [59], également appelé l'algorithme du "K-mean" [60], permet de construire un quantificateur quasi optimal. Cet algorithme itératif vérifie successivement les deux conditions d'optimalité.

ii. Approche itérative de Lloyd-Max

Dans la pratique, on souhaite réaliser la partition subjective idéale des chrominances de l'image révélant les régions de couleur peau de l'image, l'utilisation de l'algorithme de Lloyd-Max est un moyen efficace d'y accéder. Toutefois, comme nous le verrons, ce partitionnement orienté chrominance pose un problème particulier sur l'initialisation des représentants. Nous décrivons dans un premier temps l'algorithme de Lloyd-Max appliqué à la partition de l'espace des chrominances en 4 classes, puis nous traitons le problème de l'initialisation des représentants.

Algorithme de Lloyd-Max

1. Initialisation des 4 représentants.
2. Règle du plus proche voisin vis à vis des 4 représentants pour tous les pixels de l'image.
3. Mise à jour des représentants : chaque nouveau représentant est égal à la valeur moyenne des pixels de la classe concernée ce qui correspond à l'estimation du centroïde.
4. Itération des étapes 2 et 3 jusqu'à la convergence des classes.

Initialisation des représentants

Pour initialiser les représentants, on peut procéder comme suit, à l'aide de l'algorithme dit LBG (Linde, Buzo et Gray [61]) :

1. On ne considère initialement qu'une seule classe regroupant l'ensemble des pixels. Le représentant, $R^0(0)$, est choisi de manière à minimiser l'erreur quadratique moyenne. Il correspond, par conséquent, au centre de gravité des pixels.
2. A partir du représentant $R^0(0)$ on construit ensuite deux nouvelles classes de représentants respectifs $R^0(1) = R^0(0)$ et $R^1(1) = R^0(0) + \varepsilon$. ε est affecté d'une petite valeur mais le choix de cette valeur reste arbitraire.
3. Une partition en deux classes selon l'algorithme de Lloyd-Max est effectuée. Les représentants initiaux sont $R^0(1)$ et $R^1(1)$ précédemment calculés.
4. Les deux nouveaux représentants servent à leur tour de base à l'initialisation de quatre nouveaux représentants.
5. Les étapes 3 et 4 sont itérées, l'algorithme se termine lorsque le nombre de classes souhaité est atteint.

Toutefois, cette méthode d'initialisation n'est pas suffisamment robuste, et par ailleurs, elle ne prend pas en compte l'hypothèse de l'existence d'une région "peau".
Plus simplement, connaissant la valeur statistique du représentant "peau" et ayant fixé, après essais, le nombre de représentants à quatre, il devient possible de fixer empiriquement les autres représentants des classes de façon à centrer la classe "peau" sur la région statistique C^{ref}.

Résultats

En utilisant les représentants initiaux (fixés une fois pour toute) et en appliquant la règle du plus proche voisin, on peut déjà obtenir une segmentation grossière pour les normes 1 et 2 (figure 64 a et b). Si maintenant on affine l'estimation de ces représentants à l'aide de l'algorithme de Lloyd-Max on obtient une segmentation moins bruitée avec des régions plus clairement marquées dans l'image (figure 64 c).

a) b) c)

Figure 64: trois approches de partitionnement[1] : a) Norme1, b) Norme2, c) Itérative

Le choix de l'algorithme itératif de Lloyd-Max avec une initialisation des représentants sur la base de données statistiques s'avère donc efficace. Cependant, un inconvénient majeur de cette approche est que la complexité dépend du nombre d'itérations qui n'est pas connu d'avance. Un critère d'arrêt de l'algorithme doit donc être mis en oeuvre, soit il est basé sur la variation de l'erreur quadratique entre deux valeurs successives des représentants et dans ce cas l'algorithme s'arrête lorsque l'erreur est inférieure à un seuil fixé, soit, le nombre d'itérations est fixé selon un critère empirique. Dans le premier cas, l'algorithme n'est pas régulier et sa complexité n'est pas maîtrisée. Dans le deuxième cas, l'algorithme est régulier mais un compromis entre la précision des représentants et la complexité doit être trouvé.

I.2.2.c) Approche adaptative inter-image

L'approche itérative de Lloyd-Max bien que conduisant à une segmentation correcte, n'est pas satisfaisante du point de vu de la complexité. Afin, de satisfaire à la contrainte de faible complexité nous faisons l'hypothèse que deux images consécutives d'une vidéo sont fortement corrélées et ne diffèrent que par des variations locales de chrominances (et de luminance).
Les représentants optimum de l'image I_n sont alors similaires à ceux de l'image I_{n-1}. Les représentants de l'image I_{n-1} sont, par conséquent, une bonne estimation de ceux de l'image I_n, et une seule itération de l'algorithme de Lloyd-Max suffit pour l'adaptation des représentants de I_n. Notre approche conduit ainsi à un algorithme de complexité maîtrisée (une itération par image). La figure 65 montre que cette approche adaptative est près de 5 fois moins complexe que l'approche itérative.
En outre, nous avons pu vérifier sur un grand nombre de séquences la rapidité et la robustesse de la convergence. Nous avons pu constater, par exemple, que quand il faut trois itérations (sur une même image) à l'algorithme de Lloyd-Max pour converger, il faut trois images consécutives pour que la version adaptative converge (figure 66).
Par ailleurs, le choix de valeurs initiales (première image de la séquence) particulières pour les représentants conduit, dans la majorité des cas, à une segmentation exploitable dès la première image. La figure 68 illustre ce dernier point ainsi que la convergence des représentants dans l'image.

1. Les différents niveaux de gris correspondent à différentes classes.

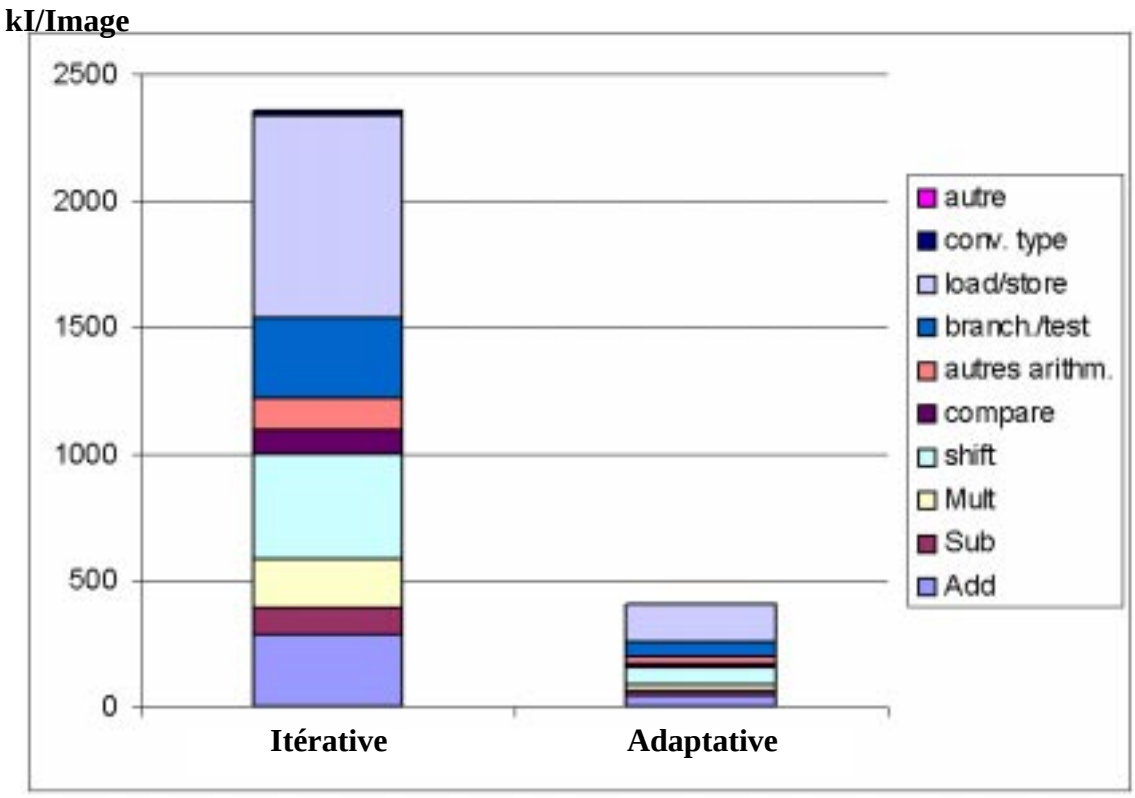


Figure 65: complexité de la méthode adaptative vs méthode itérative Lloyd-Max

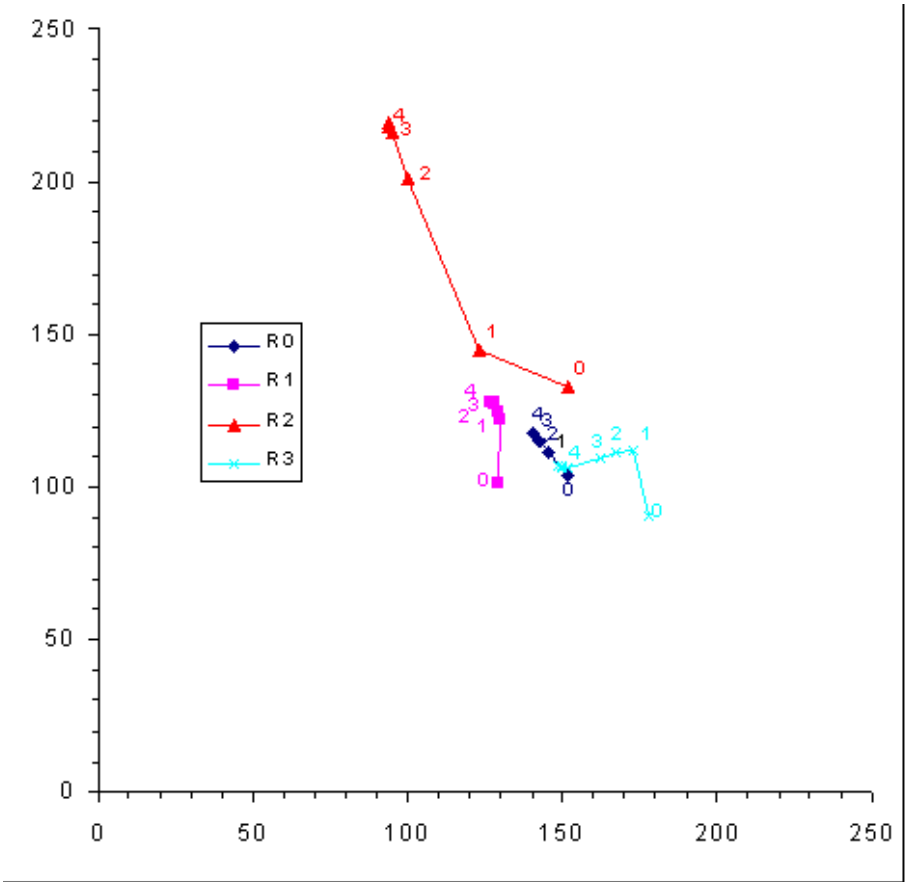


Figure 66: convergence des représentants dans l'espace des chrominances

image 1 image 2 image 3 image 4 image 5

image 21

a) Akiyo

image 1 image 2 image 3 image 21

b) Carphone

Figure 67: convergence de l'algorithme adaptatif sur différentes séquences

Les séquences ci-dessus ainsi que celles fournies en Annexe C.3 montrent l'efficacité de cette approche et illustrent également le caractère adaptatif de l'algorithme. Concrètement, on observe une convergence des classes sur les premières images de la séquence vidéo. L'estimateur des classes est, ensuite, continuellement adapté au cours du temps aux variations locales des chrominances inter image.

Bien sûr, à l'issue de cette segmentation, il reste à déterminer quelle classe correspond à la couleur peau afin de pouvoir construire le masque binaire.

I.2.2.d) Choix du représentant de la peau

La sélection de la classe correspondant à la couleur peau est faite par comparaison des représentants estimés avec le représentant statistique de la couleur "peau" de la figure 58. Dans le but de simplifier l'opération de sélection, la classe de référence, C^{ref}, est ramenée à un rectangle comme décrit sur la figure 68.

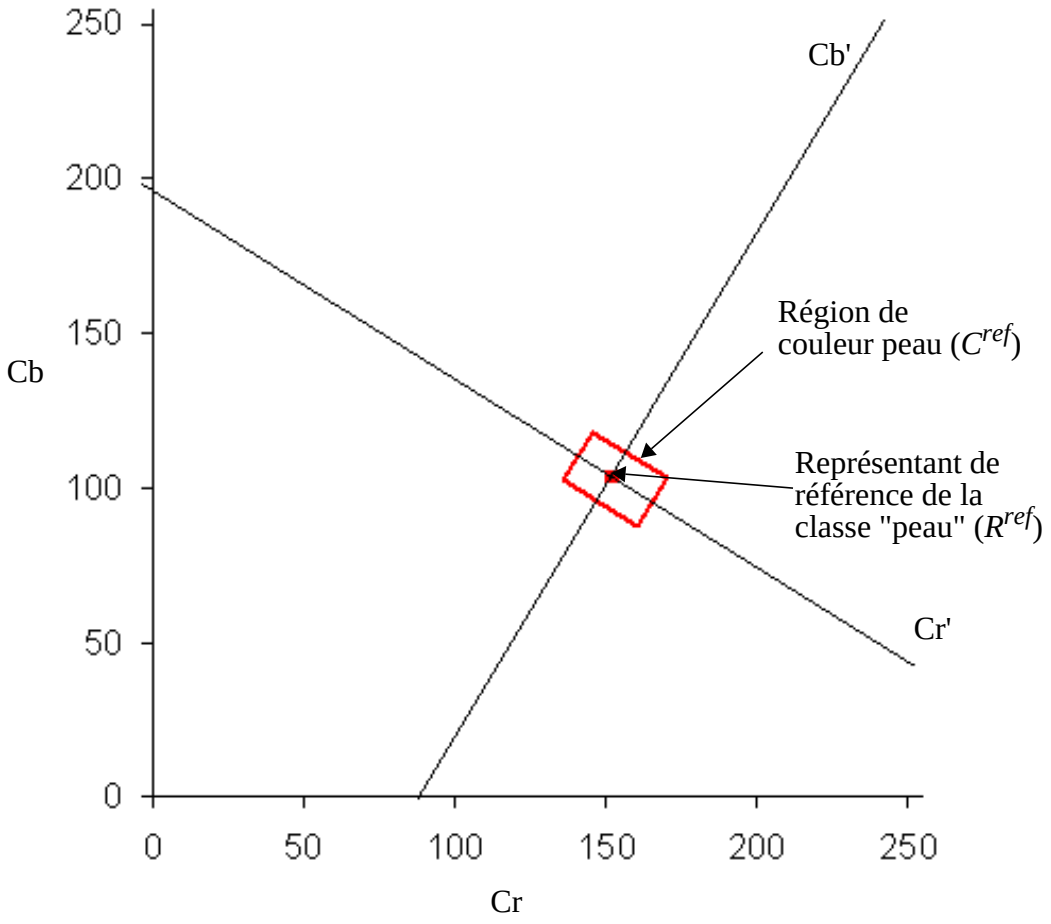


Figure 68: approximation de la région des couleurs peau

Le choix du représentant de la couleur "peau" est effectué en deux étapes :
1. on construit l'ensemble E_r des représentants appartenant à C^{ref}.

$$E_r = \{R^l, (l = [0, 3]) : R^l \in C^{ref}\} \qquad \text{(C.6)}$$

Cette étape correspond à un changement de repère selon l'équation (C.7) suivi d'un test sur les coordonnées des représentants dans le nouveau repère.

$$\tilde{R}^l = M \cdot C^l + T \qquad l = [0, 3] \qquad \text{(C.7)}$$

R^l représentant de la classe l dans le repère (Cr, Cb) des chrominances

$\tilde{R}^l$ représentant de la classe l dans le nouveau repère $(\tilde{C}r, \tilde{C}b)$

M matrice de rotation M

T vecteur de translation T

2. puis, on choisit le représentant de la peau parmi les représentants de l'ensemble E_r selon l'équation (C.8).

$$R^{peau} = \arg\ min_{R \in E_r}(\|R - R^{ref}\|_2) \quad \text{(C.8)}$$

R^{peau} est la classe de la couleur peau

Notons que ce test permet de sélectionner un représentant pour la couleur peau, mais aussi de prendre la décision de ré-initialiser les représentants des classes si aucune classe ne correspond à la couleur peau.
Cette méthode simple du choix d'un représentant de la classe "peau" permet de prendre en compte la forme de la région statistique. Par ailleurs, une approche possible (non testée) serait d'utiliser la distance de Mahalanobis.

I.2.2.e) Synoptique de la détection

La figure 69 résume les principes fonctionnels de notre algorithme de détection des régions de couleur peau.

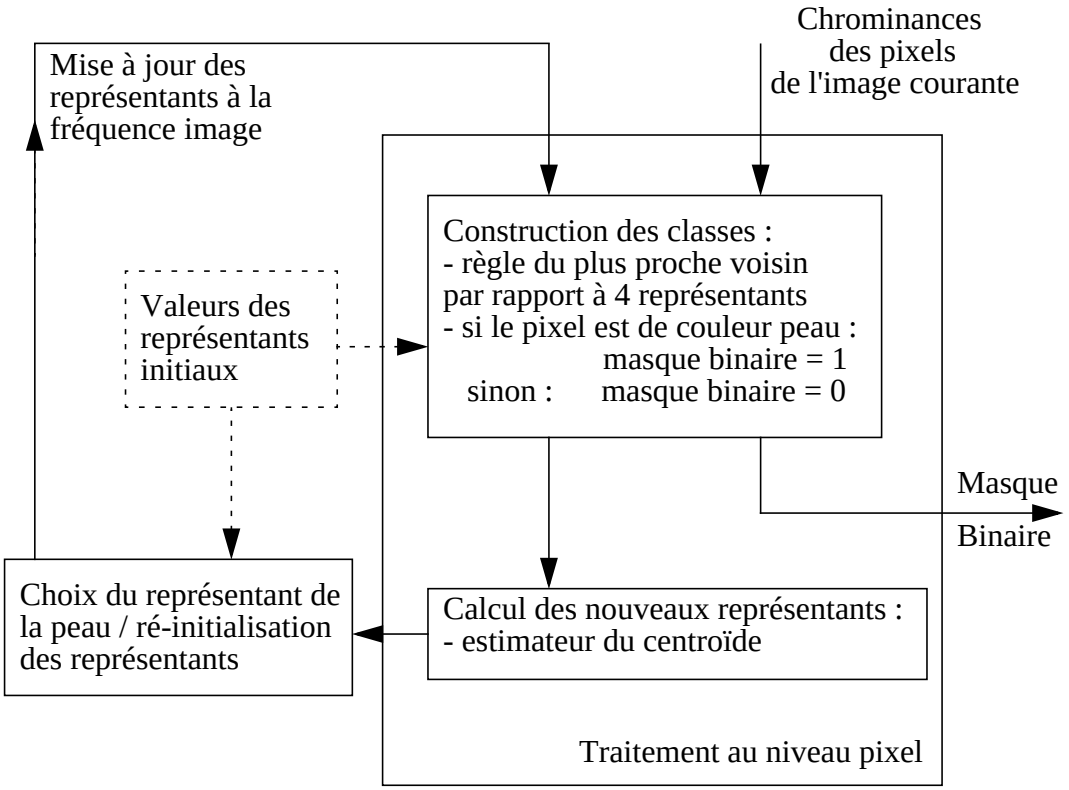


Figure 69: schéma fonctionnel de la détection de la couleur peau

I.2.3. Localisation du visage

Pour rappel, à l'issue de la détection nous obtenons le masque binaire des régions de couleur peau. Par exemple, pour la séquence Miss America nous obtenons le masque fourni en figure 70, les pixels, p_n, affectés d'un 1 appartiennent à une région de couleur peau et apparaissent en noir, ceux marqués par un 0 font partis des régions de moindre intérêt.

Figure 70: construction du masque binaire (image Miss America)

L'objectif de l'étape de localisation consécutive à la détection du visage n'est pas d'extraire le contour exact de l'objet visage, mais son contour approximé par une forme rectangulaire, comme une "boîte" circonscrite au visage.

La localisation du visage se déroule en deux phases :
- le filtrage non linéaire du masque binaire des régions de couleur peau,
- la corrélation du masque binaire filtré avec des fenêtres de taille et de forme pré-définies réalisée à l'aide d'un filtre adapté.

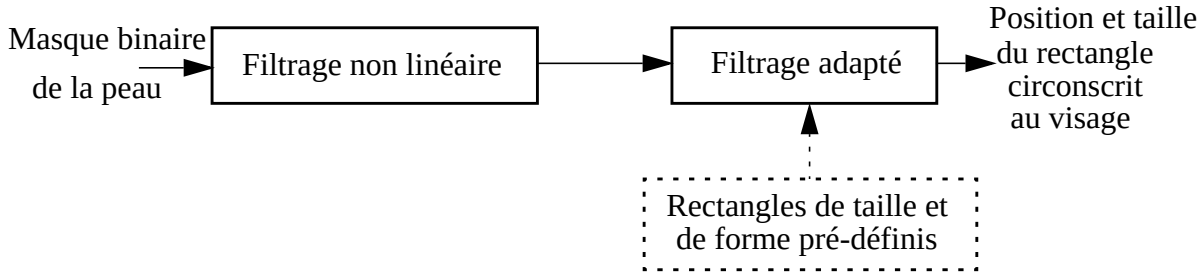


Figure 71: schéma fonctionnel de la localisation du visage

I.2.3.a) **Filtrage non linéaire**

Cette première phase a pour but de réduire la quantité d'information à traiter lors de la phase de localisation proprement dite.
Le filtrage non linéaire se déroule comme suit :
- l'image est subdivisée en blocs de taille 4*4 pixels, notés B_m. Au total, pour une image QCIF nous avons 18*22 blocs B_m, comme illustré ci-dessous.

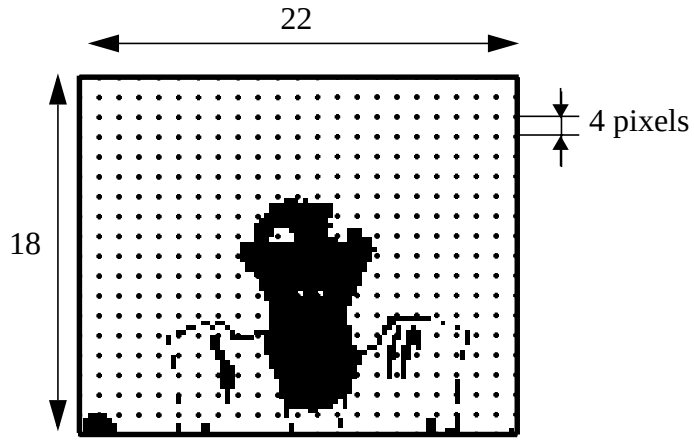


Figure 72: découpage de l'image pour l'étape de localisation

- Puis, on applique la procédure suivante :
soit S_m la sortie du filtre et τ le seuil du filtrage non linéaire, alors pour B_m chaque faire :

$$\begin{aligned} &\text{Si } \left(\sum_{p_n \in B_m} p_n \right) > (\tau \cdot card(B_m)) \quad , \text{ avec } (card(B_m) = 16) \\ &\text{Alors faire : } (S_m = 1) \\ &\text{Sinon faire : } (S_m = 0) \end{aligned} \qquad \text{(C.9)}$$

La région d'intérêt correspond bien sûr aux valeurs égales à 1. L'ensemble des S_m constitue le masque binaire filtré que l'on note M_F.

- à l'issue de ce filtrage le masque de la couleur peau est réduit à une image de taille 18*22 pixels
La figure 73 illustre le filtrage non linéaire sur une image de la séquence Miss America avec un seuil fixé à $\tau = 45\%$.

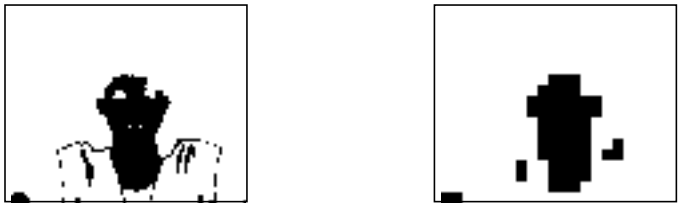


Figure 73: filtrage non linéaire du masque binaire (image Miss America)

Outre la réduction d'information, ce filtrage non linéaire filtre le bruit "haute fréquence" de l'image binaire et prépare la dernière étape.

I.2.3.b) Filtrage adapté ("matched filter")

La technique employée consiste à filtrer le masque binaire par une série de fenêtres de géométrie variable approximant la forme d'un visage, et à sélectionner la fenêtre, notée F_n, donnant le meilleur score. Cette opération de filtrage adapté fournit également la position de la fenêtre ainsi estimée.
A l'étape d'initialisation du filtrage, on fixe la taille minimale de la fenêtre, le nombre de fenêtres possibles ainsi que leur forme suivant la résolution de l'image.
Nous choisissons des fenêtres rectangulaires de facteur de forme : 1 (carré) et $\frac{L}{L+\Delta}$, où L est la largeur et $L + \Delta$ est la hauteur du rectangle.
Pour une image de résolution QCIF, on fixe la taille minimale à 4*6, le nombre de cadres possibles à 13 et Δ à 2.
On dispose donc dans ce cas du panel de fenêtres (ou filtres), F_n, suivant :

Tableau 25: panel de fenêtres utilisé pour le filtrage adapté (QCIF)

F_n	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}
facteur de forme	$\frac{4}{6}$	$\frac{6}{6}$	$\frac{6}{8}$	$\frac{8}{8}$	$\frac{8}{10}$	$\frac{10}{10}$	$\frac{10}{12}$	$\frac{12}{12}$	$\frac{12}{14}$	$\frac{14}{14}$	$\frac{14}{16}$	$\frac{16}{16}$	$\frac{16}{18}$

Remarque :
Les fenêtres sont choisies de manière à construire le rectangle le mieux ajusté au visage et de taille multiple d'un MB afin d'être exploitable, comme nous le verrons, dans un schéma de codage basé bloc de type H.263.

Le filtrage adapté appliquée au masque binaire filtré retourne le meilleur score ainsi que la position de meilleur score pour la fenêtre courante.
On note ce filtrage comme suit $MF_n(M_F) = (S_n, P_n)$, où S_n est le meilleur score et P_n est la position.
La localisation se déroule selon la procédure suivante pour une image en QCIF :

pour n : 0 à 12
faire :
$(S_n, P_n) = MF_n(M_F)$
si $(S_n > \alpha \cdot card(F_n))$
faire : $\begin{cases} \hat{P} = P_n \\ \hat{F} = F_n \end{cases}$
sinon sortir de la boucle

$\hat{P}$ et $\hat{F}$ sont, respectivement, la position optimale et le facteur de forme optimal du rectangle circonscrit au visage. α correspond au seuil du critère de sélection de la fenêtre.
Soulignons que les fenêtres sont ordonnées par surface croissante. Si la fenêtre F_n ne répond pas au critère, on teste la forme F_{n+1}. Soit, cette dernière satisfait au critère, auquel cas on itère, soit, elle ne satisfait pas au critère, auquel cas on retient la fenêtre F_n, ainsi que sa position P_n. Cette stratégie permet de trouver la fenêtre la mieux ajustée au visage. Par ailleurs, si la fenêtre F_0 ne satisfait pas au critère, alors il n'y a pas de visage.

La figure 74 ci-dessous illustre la construction du cadre circonscrit au visage de Miss America. D'autres illustrations peuvent être trouvées en annexe C.3.

Figure 74: rectangle circonscrit au visage (image Miss America)

I.3. Conclusions

En résumé, la figure 75 présente le diagramme fonctionnel de l'ensemble de l'algorithme.

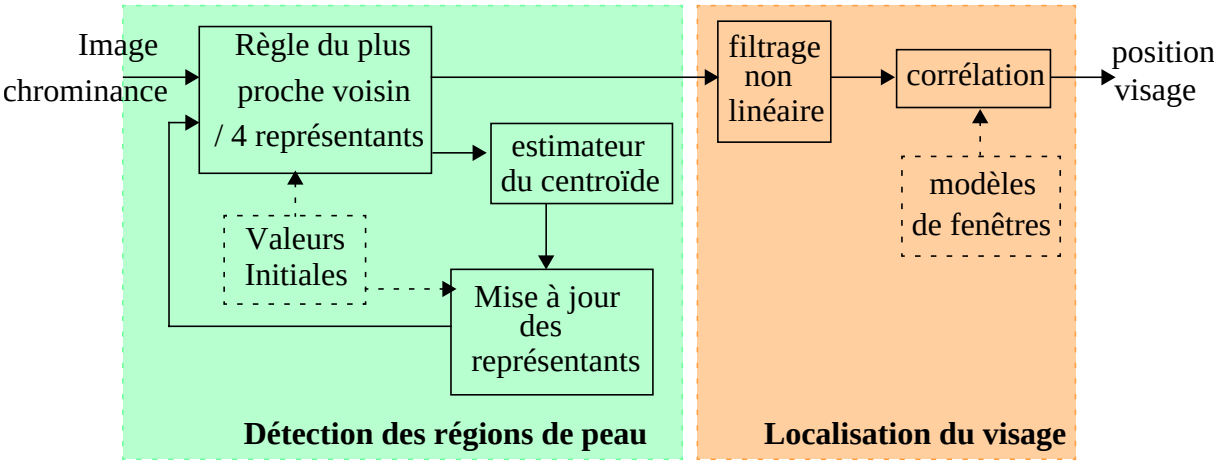


Figure 75: Diagramme fonctionnel de la localisation de visage

La complexité de l'approche adaptative proposée est nettement inférieure à celle de Lloyd-Max grâce à une technique itérative inter-image. En effet, nous obtenons une réduction d'un facteur 6 en nombre d'instructions et d'un facteur 7 en bande passante. De plus, cette réduction de la complexité ne se fait pas au détriment de la robustesse qui est équivalente d'après nos tests à l'approche itérative. L'efficacité de l'algorithme a été testée sur plusieurs séquences vidéos mises bout à bout. Aucune divergence n'a été observée.
Le traitement de la localisation, filtrage non linéaire suivi d'un filtrage adapté, conduit à une implémentation simple et efficace. Au total, le nombre d'opérations effectué par image pour la détection / localisation du visage est inférieur à 300 000 opérations/image, ce qui représente 1 % de la complexité d'un codeur H.263.
Après analyse de la complexité à l'aide de l'outil *gprof* sur station SUN (voir tableau 26), il apparaît que la majeure partie de cette complexité est localisée dans la construction des classes qui constitue le traitement au niveau pixel.

Tableau 26: temps CPU pour la détection/localisation du visage

Etapes	fonctions	% CPU
Détection du visage(77,6 %)	Accumulation des chrominances pour le calcul des valeurs moyennes	42,7 %
	Norme 2	26,1 %
	règle du plus proche voisin	8,5 %
	mise à jour des représentants / choix du représentant de la peau	0,3 %
Localisation du visage (22,4 %)	filtre adapté	15,7 %
	filtre non linéaire	6,7 %

Enfin, une illustration de la localisation du visage est fournie sur 15 images consécutives de la séquence Carphone en figure 76.

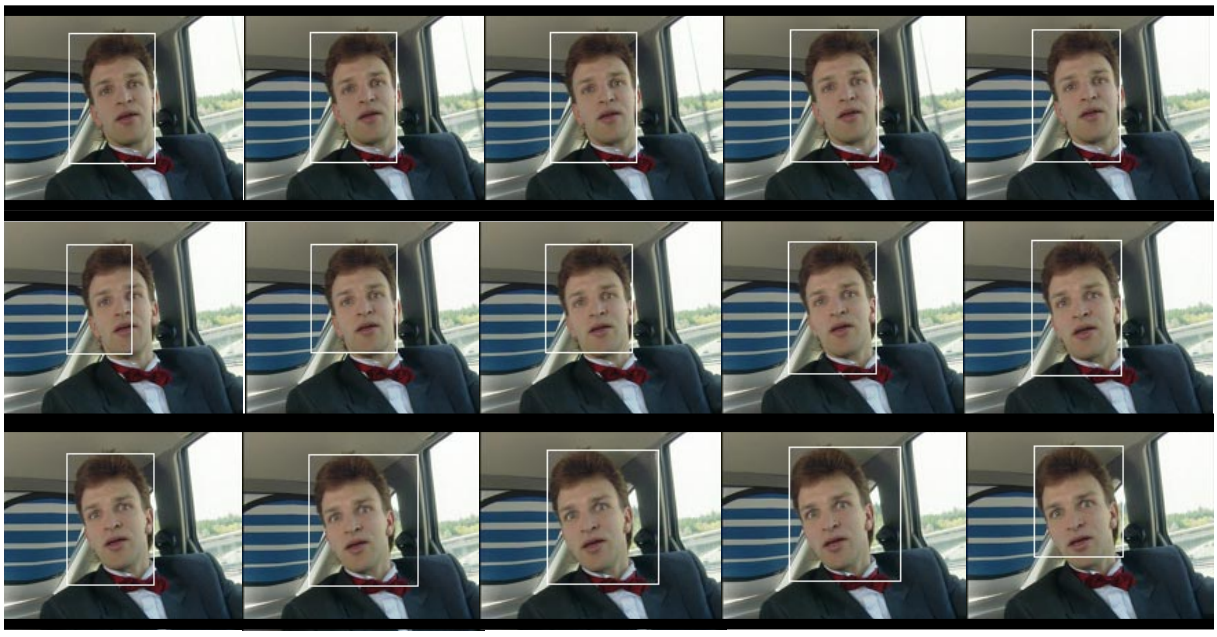


Figure 76: illustration de la localisation du visage sur 15 images de la séquence Carphone

II. Codeur orienté visage (COV)

II.1. Historique

Diverses approches de codage orientées objets sont proposées dans la littérature, essentiellement depuis l'apparition de la norme MPEG-4. Ces techniques de codage sont de deux sortes, elles peuvent soit être basées sur des normes existantes [62][63](MPEG-4, H.263), soit mettre en oeuvre des techniques de codage propriétaires [64][65][66]. L'inconvénient majeur de cette deuxième catégorie est que les codeurs peuvent moins facilement prétendre à une large utilisation du fait de leur non standardisation.

Le système proposé par H. Luo, A. Eleftheriadis et J. Kouloheris [67] fait partie de la première catégorie. Il est basé sur une modélisation statistique de l'image. Développé dans un premier temps dans le but de répondre aux exigences de segmentation de la norme MPEG-4, il permet de distinguer l'objet d'intérêt appartenant au premier plan, de l'arrière plan. Dans un deuxième temps, les auteurs sus-nommés ont songé à utiliser l'algorithme de segmentation comme prétraitement d'un codeur de type H.263, afin d'optimiser l'allocation de la bande passante du canal de transmission en fonction des objets codés. Ils ont en effet proposé d'augmenter la qualité sur la région du visage et d'optimiser le débit pour assurer une bonne synchronisation entre les lèvres et la voix. Toutefois, l'approche de H. Luo, A. Eleftheriadis et J. Kouloheris ne prend pas en considération la complexité relative de la détection par rapport à celle du codeur enjeu essentiel pour la téléphonie mobile. En effet, leur algorithme de segmentation peut traiter au maximum 15 images par seconde à la résolution QCIF (176*144 pixels) sur une plate-forme de type PC (pentium 200MHz). Utilisé conjointement avec un codeur H.263, les performances chutent à 8 ou 10 images par seconde selon leurs mesures. La complexité de leur approche est donc relativement importante mais surtout, elle est dominée par la segmentation qui représente plus de 50% de la complexité du codage, ce qui n'est pas acceptable pour des applications mobiles. De plus, une phase d'apprentissage est nécessaire afin de définir le modèle statistique initial de l'arrière plan. Du fait de cette contrainte, ce système n'est utilisable que pour le codage de scène où l'arrière plan est connu a priori. Le modèle statistique de l'arrière plan, une fois connu, est réactualisé à chaque image avec l'hypothèse que l'arrière plan est relativement fixe vis à vis de la caméra, ce qui est un fort inconvénient pour les applications mobiles. Leur système semble, a priori, être mieux adapté aux applications visiophoniques "indoor" de type internet ou salle de visioconférence, où seuls les interlocuteurs sont en mouvement.

II.2. Principes

Le système proposé consiste en un codeur H.263 classique auquel on a adjoint un pré-traitement, la détection de visage, permettant de mettre en oeuvre des stratégies de contrôle et de régulation du débit. Comme nous le verrons, le couplage de la détection du visage avec la compression vidéo H.263 conduit à une meilleure qualité visuelle, sans augmentation de la complexité globale du système de codage. De plus, ce système peut participer à l'amélioration de l'ergonomie du service de visiophonie. La figure 77 présente les principes du codeur orienté objet d'intérêt que nous appellerons COV pour *Codeur Orienté Visage*. Les lignes en pointillé sont les contrôles possibles dans le schéma de codage adopté.

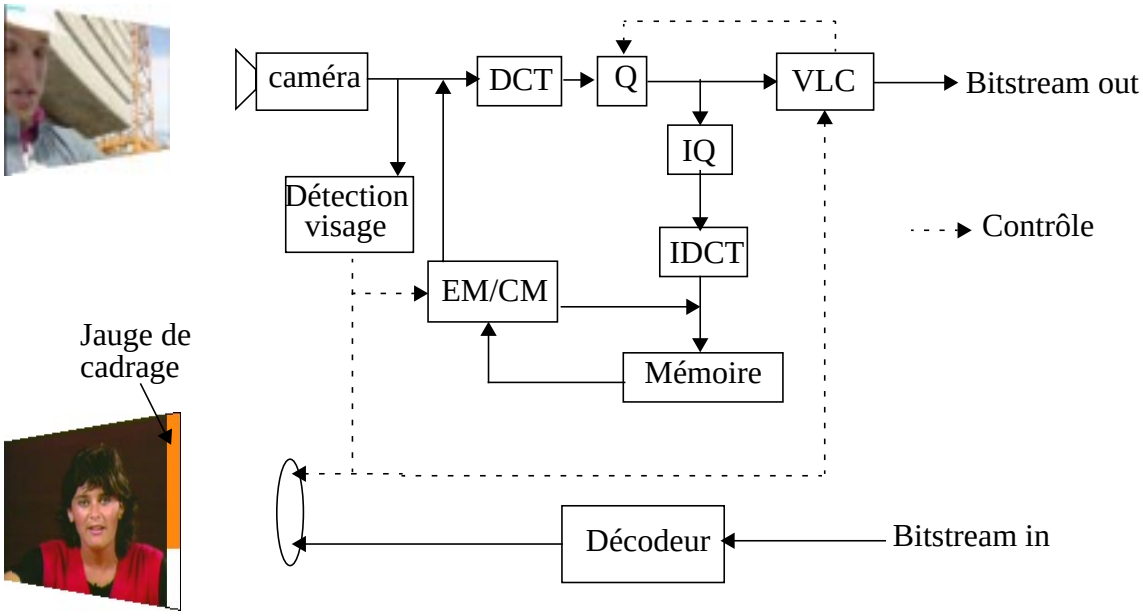


Figure 77: diagramme fonctionnel du codec H.263 orienté visage

II.3. Estimateur de mouvement orienté visage

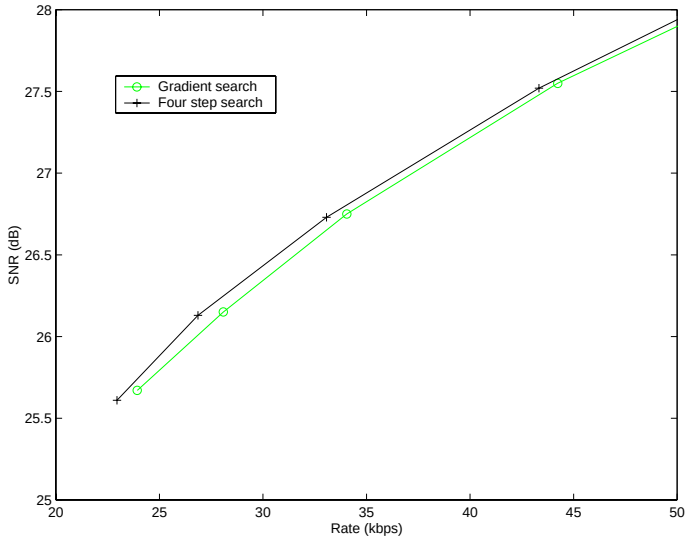
La détection de visage va permettre d'agir sur l'estimation de mouvement, de façon à coder "intelligemment" l'image. Cette stratégie aura à la fois des effets sur la qualité de l'image et sur la complexité globale du codeur.

II.3.1. Influence sur la qualité des estimations

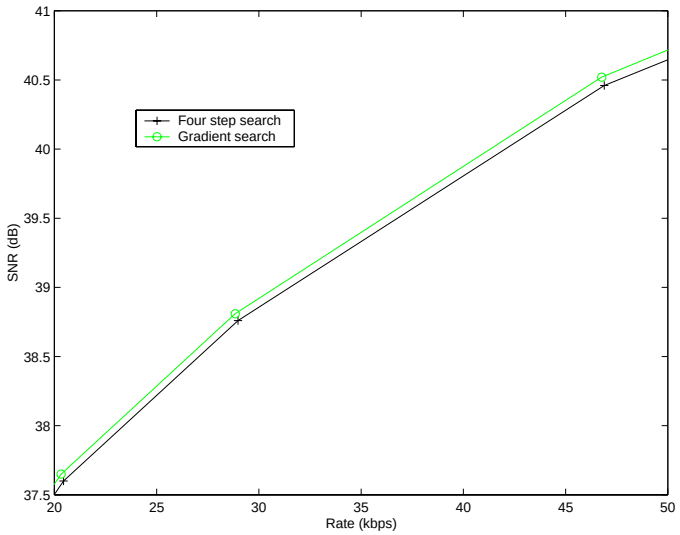
Comme nous l'avons vu dans le chapitre précédent, les performances des estimateurs de mouvement rapides dépendent de la stratégie de recherche employée. Mais la qualité de ces estimations est également fonction de la nature du mouvement et du degré de corrélation spatio-temporelle des MBs. En effet, on constate que selon la nature du mouvement (uniforme, aléatoire, ...) l'efficacité d'un estimateur de mouvement rapide est plus ou moins grande.
Dans une scène de visiophonie mobile, on est amené à distinguer trois types de mouvements directement liés à la nature des objets et à leur taille :
- le premier est celui du visage formé de plusieurs MBs et modélisé par un déplacement uniforme et des déformations locales,
- le deuxième type correspond aux objets de l'arrière plan animés de mouvements indépendants,
- et le troisième correspond au cas d'un arrière plan fixe.

Ainsi, dans le cas du visage il est souvent plus approprié de choisir un EM rapide prenant en compte le mouvement des MBs voisins dans l'estimation du mouvement du MB courant. C'est le cas de la recherche selon le gradient (GDS).
Pour l'arrière plan, plusieurs estimateurs de mouvement rapides sont possibles. Nous avons privilégié la méthode de recherche en "4 pas" (4SS) car elle est adaptée aux deux types d'arrière plan définis précédemment. En effet, la recherche du vecteur mouvement d'un MB par cette technique commence par un test de la valeur de distorsion obtenue pour un mouvement nul. Si cette valeur est inférieur à un certain seuil (fixé empiriquement) alors on décide que le MB est immobile et on évite ainsi des calculs inutiles, sinon on poursuit la recherche du vecteur mouvement dans la fenêtre de recherche avec une stratégie tenant compte des mouvements de forte amplitude.

La figure 78 compare la méthode de recherche selon le gradient à la recherche en "4 pas" pour une séquence comportant des mouvements rapides et aléatoires (Coastguard) et pour une séquence typique de visiophonie où seul est présent un visage (Miss America). Dans le premier cas (Coastguard), on vérifie bien que la méthode 4SS est légèrement meilleure que la méthode GDS, alors que dans le deuxième cas (Miss America), la méthode GDS donne de meilleurs résultats en terme de rapport signal à bruit. Toutefois, il faut reconnaître que les différences sont minimes, ce que l'on peut expliquer du fait que la qualité de la vidéo à des débits très faibles est limitée par la forte compression et n'est plus dominée par la précision des estimations (cf. paragraphe B.II.3.2).



a)



b)

Figure 78: SNR vs technique d'EM a) Coastguard, b) Miss America

II.3.2. Amélioration du compromis qualité / complexité

Le véritable avantage d'un contrôle de l'estimation de mouvement est une plus grande flexibilité sur le compromis qualité / complexité de l'estimateur. En effet, l'information de localisation du visage autorise plusieurs stratégies de codage et la possibilité d'utiliser deux estimateurs de mouvements différents. Une stratégie particulièrement efficace est d'utiliser la méthode GDS pour effectuer l'estimation de mouvement sur l'objet "visage", et la méthode 4SS pour l'estimation du mouvement des MBs de l'arrière plan. Le tableau 27 donne la complexité du codeur pour 3 exemples de configuration par rapport à un codeur utilisant une recherche exhaustive. La qualité de codage sur les séquences Carphone et Foreman sont fournies dans les tableaux 28 et 29 (15Hz@40kbits/s).

Tableau 27: complexité en fonction de la stratégie d'EM

	FS	FS/4SS	GD/4SS	GD/CSA
Nb images P	185	185	185	185
Complexité globale	**77.45**	**53.54**	**34.66**	**34.02**
Arithmétique	39.48	31.25	21.73	21.43
inc., or, and, ...	0.63	0.36	0.26	0.26
Test, branch.	11.01	6.19	3.32	3.21
L / S	21.61	12.96	7.94	7.76
Conversion	0.41	0.52	0.44	0.44
Autres	4.31	2.26	0.97	0.92

Tableau 28: SNR en fonction de la stratégie d'EM : séquence Carphone

	FS	4SS	FS/4SS	GD/4SS	GD/CSA
SNR total	185 images codées				
Y	30.49	30.4	30.04	30.06	29.78
Cr	36.53	36.53	36.36	36.35	36.14
Cb	35.86	35.87	35.43	35.49	35.32
SNR visage	185 visages détectés				
Y	29.94	29.72	29.88	29.92	29.7
Cr	35.14	34.8	35.06	35.07	34.88
Cb	35	34.98	34.96	35.01	34.77

Tableau 29: SNR en fonction de la stratégie d'EM : séquence Foreman

	FS	4SS	FS/4SS	GD/4SS	GD/CSA
SNR total	124	123	124	124	123
Y	29.52	29.15	28.85	28.8	28.03
Cr	35.89	36	36.13	36.15	35.71
Cb	35.44	35.39	35.39	35.25	34.71
SNR visage	124	123	124	124	123
Y	29.56	29.21	29.79	29.73	29.25
Cr	34.36	33.8	34.58	34.65	34.17
Cb	35.51	35.39	35.73	35.47	35.05

On observe que l'utilisation d'une stratégie basée sur la méthode du gradient alliée à la technique de recherche des vecteurs mouvement en "4 pas" entraîne une réduction de la complexité de 55 % par rapport à la méthode FS (recherche exhaustive), pour une qualité d'image équivalente. Par ailleurs, la qualité subjective obtenue en comparaison de celle obtenue avec la méthode 4SS est meilleure pour un surcoût de complexité de seulement 7 % (cf. tableau 19 du paragraphe B.II.3.2 et tableau 27). Notons que cette approche permet de sauter moins d'images pour les vidéo comportant beaucoup de mouvement comme la séquence Foreman (123 images codées pour la méthode 4SS, 124 images codées pour l'approche GD/4SS).

II.4. Contrôle du débit orienté visage

La connaissance de la position du visage du locuteur permet également de mettre en place une stratégie d'allocation de bits orientée visage. Ce contrôle de débit, comme pour la méthode classique, se situe à deux niveaux : le niveau image et le niveau macrobloc.

II.4.1. Au niveau image

La technique classique de régulation du débit au niveau image est, comme nous l'avons vu dans le chapitre B, de sauter des images quand le débit est jugé insuffisant. Cette approche n'est pas satisfaisante dans le cas de la visiophonie. En effet, si le nombre d'images non codées devient important, la vidéo est saccadée ce qui conduit généralement à une perte de compréhension dans la communication et rend le service inacceptable.
Le nombre d'images sautées dépend bien entendu du débit disponible mais également, et surtout, de la quantité d'information à transmettre. On fera ici la distinction entre l'information concernant le locuteur et celle concernant l'arrière plan. L'information à transmettre pour décrire l'image du locuteur est indispensable, en revanche, celle de l'arrière plan n'est pas primordiale et une réduction de la fréquence de rafraîchissement est possible sans que cela nuise à la communication audiovisuelle. Aussi, nous proposons un nouveau schéma de régulation où la fréquence de l'arrière plan est divisé par deux. Dans ce schéma, les bits économisés serviront à rehausser la qualité sur le visage.

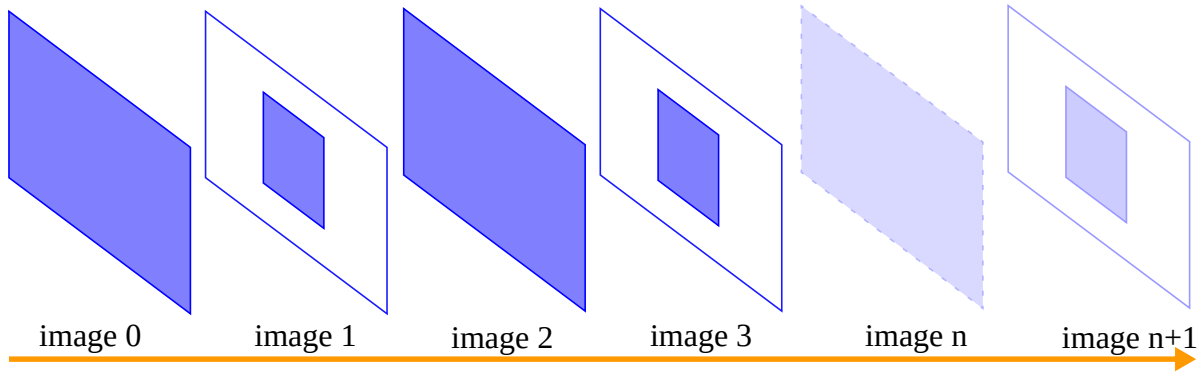


Figure 79: stratégie de codage orientée visage

Plus précisément, dans ce schéma de codage, la première image est codée entièrement, la seconde n'est codée que sur l'objet "visage", la troisième est à nouveau codée entièrement et ainsi de suite (figure 79). Pour une fréquence image cible de 15 Hz, le visage est donc codé à 15 Hz alors que l'arrière plan est codé à une fréquence moitiée, soit 7.5 Hz.

II.4.2. Au niveau macrobloc

Au niveau MB, nous avons choisi d'utiliser la régulation définie par la recommandation TMN8 mais en l'adaptant à la taille de l'objet codé : image complète ou visage seul.

En outre, d'autres possibilités d'allocation orientée objet sont possibles dans un schéma de codage H.263 suivant les options sélectionnées. Nous en proposons une :
la régulation TMN8 est basée sur une modélisation de la répartition des bits disponibles sur tous les MBs de l'image ne faisant pas de distinction entre les MBs (appartenance au visage ou non). Une amélioration possible serait une stratégie utilisant deux modèles, un pour les MBs du visage et l'autre pour les MBs de l'arrière plan.

$$B = B_{visage} + B_{arrireplan} \qquad (C.10)$$

avec $\begin{cases} B_{visage} : \text{nombre de bits alloué aux MBs du visage} \\ B_{arri'ere\ plan} : \text{nombre de bits alloué aux MBs de l'arrière plan} \\ B : \text{nombre total de bits disponible par image} \end{cases}$

La division en deux débits cibles, notés B_{visage} et $B_{arrière\ plan}$, offre la possibilité de rehausser la qualité de codage du visage (au détriment de l'arrière plan, bien entendu) en choisissant :

$$\left(\frac{B_{visage}}{N_{visage}} > \frac{B_{arriereplan}}{N_{arriereplan}}\right) \text{ où } \begin{cases} N_{visage} \text{ est le nombre de MBs appartenant à l'objet visage} \\ N_{arriereplan} \text{ est la nombre de MBs appartenant à l'arrière plan} \end{cases}$$

De plus, cette division en deux cas a évité une chute de la qualité sur le visage dans le cas où l'arrière plan possède beaucoup de mouvement. En effet, les deux régions étant traitées séparément, les bits nécessaires au codage des MBs d'erreur de prédiction importante (fort mouvement de l'arrière plan) ne seront pas "pris" aux MBs du visage.

III. Apports pour le mobile

III.1. Régulation du débit

Ce nouveau schéma de régulation prend mieux en compte la nature de la vidéo et offre une plus grande flexibilité avec la capacité d'absorber des variations brutales de débit, par exemple, lorsque celui-ci tombe en dessous de 20 kbit/s la qualité de l'image reste acceptable alors que dans le cas d'un codage classique la vidéo n'est plus regardable (figure 80). De plus, il devient possible de baisser à tout moment la fréquence image de l'arrière plan (jusqu'à ne plus transmettre que le visage) et/ou de réduire le nombre de bits alloués à l'arrière plan. On définit ainsi un codeur que l'on peut qualifier de "scalable". En effet, cette régulation s'apparente à une décomposition graduelle d'objet et de qualité.

a) **b)**

Figure 80: Séquence Carphone codée à 15 kbit/s
a) H.263 classique, b) H.263 orienté visage

III.2. Schéma de scalabilité

Une approche possible serait de définir différents niveaux de "scalabilité", par exemple les niveaux suivants :
L0 : niveau de base.
L1 : premier niveau de "scalabilité". Insertion d'image du visage afin d'augmenter la fréquence image de celui-ci. Ce niveau conduit à une meilleure intelligibilité de la vidéo-communication.
L1 : deuxième niveau de "scalabilité". Insertion d'image de l'arrière plan afin d'augmenter la qualité et la fréquence des images.
La figure 81 présente ces 2 niveaux de "scalabilité".

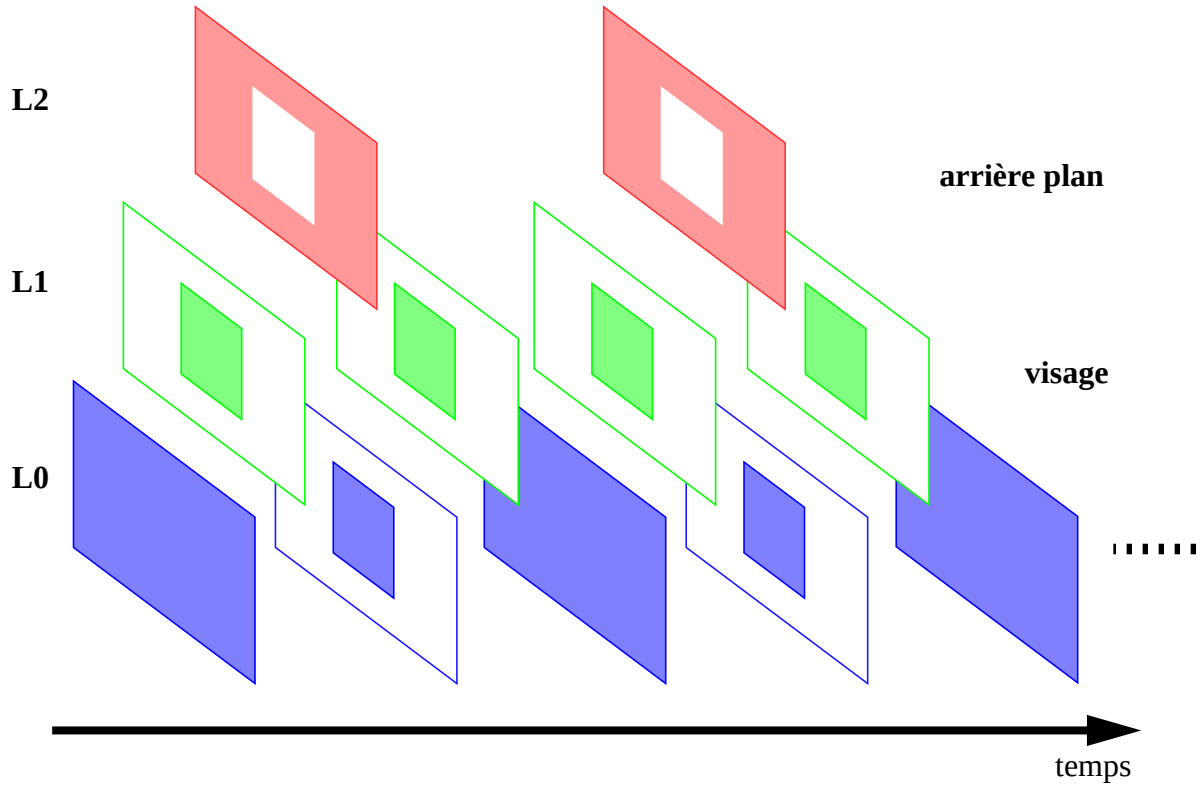


Figure 81: scalabilité d'objet et temporelle

III.3. Performances du COV

III.3.1. Complexité

Cette étude de complexité utilise la même méthodologie d'évaluation que celle appliquée pour le codeur standard H.263 au chapitre B.
Le tableau 30 donne les complexités de différentes réalisations de codeur H.263 : un codeur de référence avec une recherche exhaustive des vecteurs mouvement, deux codeurs avec des techniques de recherche rapides, recherche en quatre pas (4SS) et recherche selon le gradient (GDS), et le dernier implante le codeur COV tel que proposé.
Les mesures sont effectuées sur la séquence Carphone codée à 15 Hz pour un débit de 40 kbit/s.

Tableau 30: complexité du COV par rapport aux codeurs "classiques"

Complexité / Profil		FS	4SS	GDS	COV
Million Instructions/image		**77.45**	**32.33**	**31.69**	**24.21**
Complexité arithmétique*		***41.23***	***17.39***	***17.39***	***13.53***
	Add*	14.9	6.41	6.28	4.47
	Sub*	7.25	0.96	0.89	0.85
	Mult*	0.73	0.66	0.66	0.42
	Div*	0.05	0.05	0.05	0.03
	Shift*	12.71	9.38	9.3	7.55
	Autres(inc,...)	0.63	0.51	0.51	0.2
	Cmp* (abs)	5.6	0.27	0.21	0.22
Branch. Test		9.26	5.11	5.05	3.87
Load / Store		21.61	7.73	7.55	5.64
Conversion de type		0.41	0.34	0.33	0.31
Autres instructions		4.31	0.91	0.87	0.67
Bande passante		**28.09**	**15**	**14.76**	**10.37**
	Lecture	24.05	11.23	11.01	7.73
	Ecriture	4.04	3.77	3.75	2.64

Tableau 31: pourcentage du temps CPU du COV par rapport aux codeurs "classiques"

	EM/CM	DCT/IDCT	Q/IQ	VLC	Détection	Autres
FS	84.21	10.97	2.79	1.91	0	0.12
4SS	48.18	34.52	10.24	6.91	0	0.15
GDS	47.53	37.47	7.78	6.99	0	0.23
COV	46.45	30.94	10.22	5.3	6.71	0.38

Commentaires :
Sur la séquence Carphone, nous observons que les stratégies de codage mises au point pour le codeur orienté visage entraînent une réduction supplémentaire de la complexité par rapport au codeur utilisant une recherche rapide des vecteurs mouvement. Une réduction d'environ un facteur 2.5 était obtenue pour les méthodes 4SS et GDS. Pour le COV, une diminution de la complexité d'un facteur 3.2 est atteint. Ceci s'explique par le schéma de régulation pour lequel

l'arrière plan n'est codé qu'une image sur deux, évitant ainsi de prédire et de coder de nombreux MBs.
Le tableau 31 indique, comme pour les autres codeurs, que la complexité du COV est dominé par le traitement d'estimation de mouvement. Soulignons que le traitement de détection du visage, bien que représentant 6 % de la complexité du codeur, a pour effet de faire baisser la complexité globale du codeur, ce qui est très satisfaisant au regard des fortes contraintes du terminal.

III.3.2. Qualité

Le tableau 32 donne le rapport signal à bruit obtenu sur la séquence Carphone pour les différents codeurs.

Tableau 32: Rapport signal à bruit sur le visage (COV vs "classiques")

	Y	Cr	Cb
FS	29.94	35.14	35
4SS	29.87	35.17	34.97
GDS	29.86	35.2	35.05
COV	**32.37**	**37.22**	**37.74**

La stratégie adoptée d'allocation des bit orientée visage conduit à un gain significatif en qualité sur le visage. Cette amélioration est particulièrement visible pour les séquences contenant un arrière plan en mouvement comme les séquences Carphone et Foreman. Par contre, les séquences de visiophonie avec un arrière plan fixe (Miss America, Akiyo, Claire), nécessitant que peu de bits pour le codage de l'arrière plan, tirent moins partie de notre stratégie. Le tableau 33 résume les performances en qualité sur les visages du codeur COV par rapport à un codeur classique utilisant une recherche exhaustive des vecteurs mouvement pour les deux types de séquences visiophoniques.

Tableau 33: Amélioration apportée par le COV

SNR	Carphone, Foreman	Akiyo, Claire, Miss America
Y	+ 1.95 dB	+ 0.81 dB
Cr	+ 1.71 dB	+ 0.91 dB
Cb	+ 2.14 dB	+1.05 dB

Nous avons choisi de fournir le rapport signal à bruit uniquement sur le visage car cela correspond mieux à la qualité subjective de la vidéo. En effet, bien que le rapport signal à bruit sur toute l'image soit plus faible avec le codeur COV (du à l'arrière plan), on observe une nette amélioration de la qualité subjective de la vidéo comme l'illustre les figures 82 et 83 sur les séquences Carphone et Miss America (d'autres exemples sont fournis en annexe C.7).

Figure 82: qualité subjective sur la séquence Carphone (COV vs FS)

Figure 83: qualité subjective sur la séquence Miss America

Commentaires :

- Sur l'image 100 de la séquence Carphone des défauts de contours de blocs apparaissent aux frontières du rectangle circonscrit au visage car pour cette image seule la zone délimitée par le rectangle est codée (figure 82). Cependant, nous observons en général que les défauts liés à la stratégie de codage sont peu perceptibles à la fréquence image de 15 Hz. En outre, ils peuvent être réduit par un filtrage des contours des MBs, comme celui spécifié dans l'annexe J de la norme H.263.
- En comparant l'image 100 pour le codeur COV au codeur "classique", on note qu'un objet de l'arrière plan n'est pas à la même position. Ceci est du à la fréquence image de 7.5 Hz de l'arrière plan pour le COV. Toutefois, le regard étant focalisé sur le visage de premier plan, ce

retard passe inaperçu.
- Nous observons que, dans le cas d'un codage "classique", les traits du visage sont flous, ce qui nuit à l'intelligibilité de la communication, alors qu'avec l'approche proposée les traits du visage apparaissent plus clairement.
- Pour une séquence de visiophonie où l'arrière plan est fixe (séquence Miss America, figure 83), nous constatons que les qualités subjectives du codeur COV et du codeur 'classique' sont identiques.

En conclusion, notre codeur orienté visage permet de rehausser le niveau d'intelligibilité de la vidéo-communication, et fournit une qualité correcte même pour un débit très faible inférieur à 40 kbit/s.

III.4. Amélioration de l'ergonomie

Pour qu'un service de visiophonie sur mobile soit acceptable, il ne suffit pas que les contraintes de débit, de consommation, de qualité soit respectées, il faut également que l'ergonomie du terminal soit adaptée au service.

- Le problème, ici, soulevé est celui du positionnement du locuteur par rapport à la caméra de son terminal.
Généralement, l'utilisateur d'un service de visiophonie possède un retour de sa propre image afin de pouvoir se positionner dans le champ de la caméra et ainsi ne pas se retrouver hors cadre. Ce retour est habituellement réalisé par l'affichage dans un angle de l'écran d'une petite fenêtre contenant sa propre image. Cette image iconisée est souvent appelée PIP pour 'Picture In Picture'.
Cependant, l'affichage d'une PIP sur un écran de mobile réduit la surface de l'image utile déjà fortement contrainte du fait de la taille du terminal (figure 84).

Figure 84: PIP en taille réelle (écran 2'')

Le codage orientée visage proposée dans cette thèse offre la possibilité de réaliser un système simple d'assistance au cadrage utilisant l'information de position du locuteur fournie. Un diagramme fonctionnel du système proposé est fourni en figure 85 ainsi qu'une illustration d'une vidéo-communication en figure 86. Il fonctionne comme suit : une jauge est incrustée sur le côté droit de l'écran, si elle est pleine (entièrement verte), alors le locuteur est correctement positionné, en revanche si elle est vide (entièrement rouge), alors le locuteur est hors champ de la caméra de son terminal.

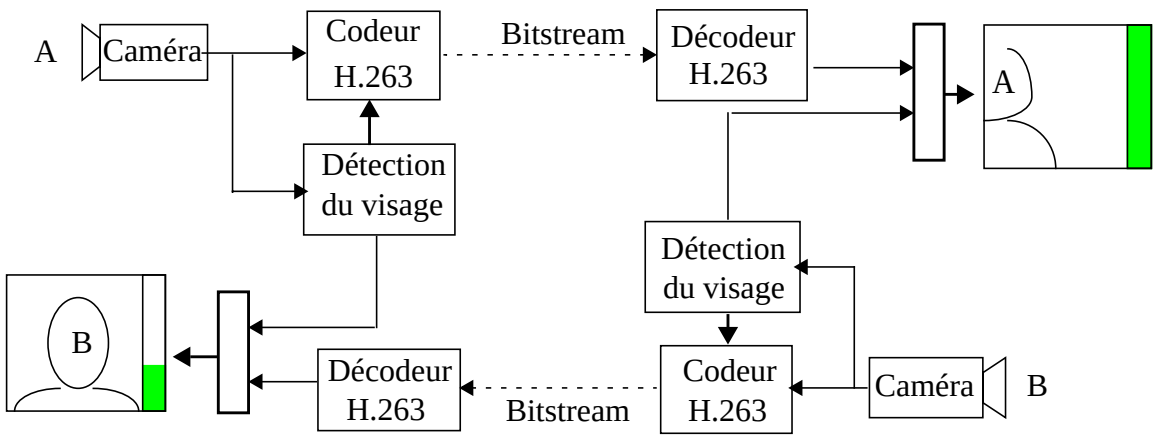


Figure 85: schéma fonctionnel de l'assistance au cadrage

Figure 86: illustration sur une vidéo-communication entre Foreman et Carphone

III.5. Résumé : codage orienté visage

L'approche de codage orienté visage proposée met en oeuvre un algorithme adaptatif de détection du visage robuste et de faible complexité. Les principales caractéristiques de cette détection du visage sont :

- Au niveau technique :
  - convergence inter-image et adaptativité au cours du temps,
  - grande robustesse,
  - facilité d'implémentation : régularité et faible complexité.
- Au niveau usage :
  - aucun apprentissage du système et donc pas de latence,
  - aucune contrainte pour l'utilisateur.

Les informations de positions et de taille du visage fournies par la détection du visage sont exploitées par le codeur vidéo dans ses stratégies de codage qui sont résumées ci-après :

- schéma de régulation : fréquence moitiée sur l'arrière plan,
- stratégie d'estimation de mouvement : mise en oeuvre de deux estimateurs de mouvement rapides,
- schéma de scalabilité temporelle et au niveau des objets,
- assistance au cadrage

Cette approche satisfait aux contraintes de complexité et de qualité de service d'une application de visiophonie mobile posées au début de cette thèse.
La figure 87 illustre, sur 15 images consécutives de la séquence Carphone, la qualité de service obtenue par l'utilisation du codeur COV.

- a) COV -

- b) Classique -

Figure 87: illustration de la qualité du codeur COV (a) vis à vis d'un codeur "classique" (b) sur la séquence Carphone

IV. Architecture

IV.1. Partitionnement HW/SW de la détection de visage

L'algorithme de détection / localisation du visage se décompose, comme nous l'avons vu précédemment, en deux principaux blocs de traitement : la détection des régions de couleur peau et la localisation d'une région correspondant au visage. Chacun de ces traitements peut être divisé en deux sous-traitements : pour la détection, la construction du masque binaire et le sélection de la classe "peau", pour la localisation, le filtrage non linéaire et le filtrage adapté. Le schéma de la figure 88 donne le partitionnement structurel.

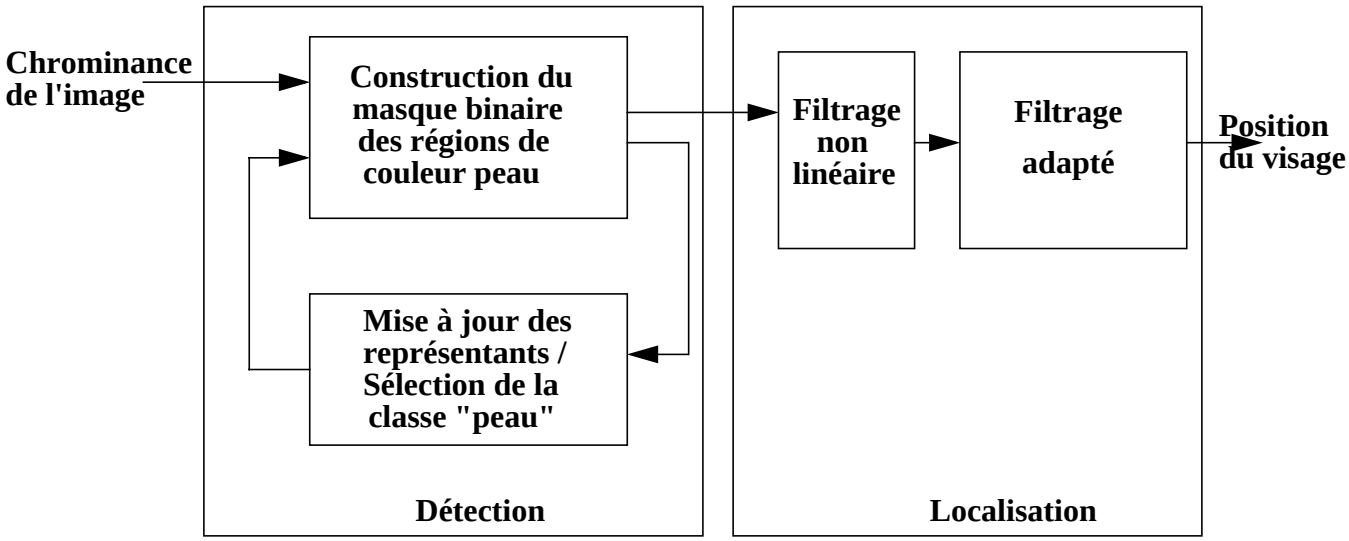


Figure 88: partitionnement structurel de la détection / localisation de visage

IV.1.1. Détection de la couleur peau

Nous donnons ci-après des éléments sur l'implémentation du traitement de la détection :

- La construction du masque binaire effectue un traitement régulier au niveau pixel. Deux opérations principales sont réalisées : la mesure de distance (Norme 2) du pixel courant par rapport à 4 représentants et l'affectation du pixel à la classe dont le représentant est le plus proche. Lors de l'affectation du pixel courant à une classe, on effectue également la construction du masque binaire : si la classe du pixel est celle de la peau alors la valeur '1' est donnée au pixel sinon la valeur '0' lui est donnée. A l'issue de ce traitement, le masque binaire servant à la localisation du visage est construit.

- La mise à jour des représentants est réalisée par le calcul de la valeur moyenne des éléments des classes. Cette étape nécessite, pour chaque classe, l'accumulation des valeurs des chrominances (Cr et Cb) de ses éléments, puis la division des deux accumulations par le nombre d'éléments la composant. Les images sont de résolution QCIF en 4:2:0. Le nombre de pixels a traiter est donc de 88*72 = 6336. Les chrominances sont codées sur 8 bits, chaque accumulateur est de taille 21 bits, estimée sur la base des valeurs moyennes des représentants et pour éviter tout débordement.

- Le sélection est faite selon la méthode décrite en C.I.2.1.d.

• Une description VHDL de ces différents blocs a été réalisée. Le diagramme des dépendances inter bloc est fourni sur la figure 89. Le tableau 34 en résume les résultats de synthèse pour une cible FPGA Altera Flex10k [68].

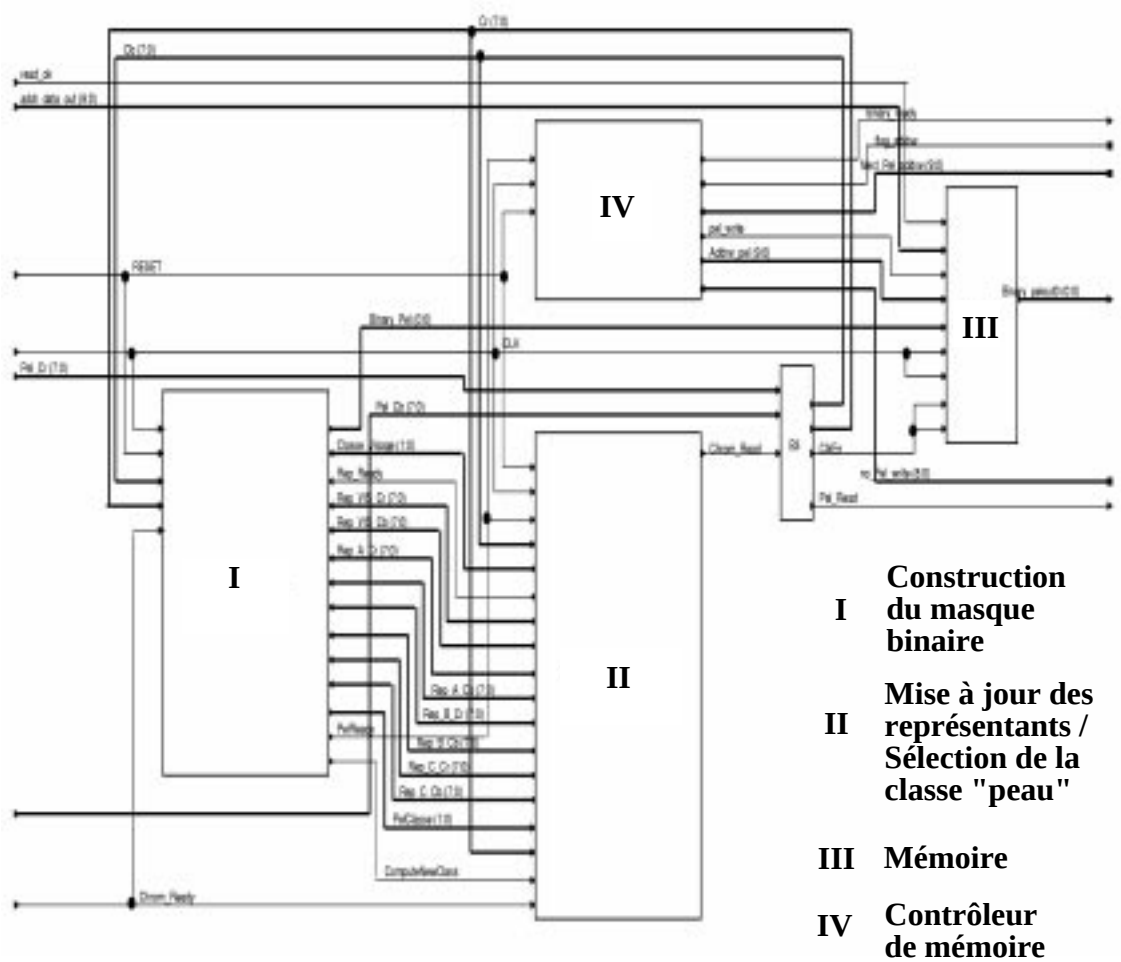


Figure 89: diagramme de la détection

Tableau 34: synthèse FPGA (nombre de cellules / surface)

Détection de la couleur peau			
3282 / 1980			
Construction du masque binaire		**Mise à jour des représentants / Sélection de la classe "peau"**	
369 / 405.5		1995 / 1545.5	
Calcul de distance	**Règle du plus proche voisin**	**Diviseur**	**Sélection de la classe peau**
89 / 211.5	84 / 88	150 / 166	512 / 608.5

Nous avons également réalisé une synthèse FPGA utilisant les options d'optimisation des outils synopsys, celle-ci conduit à 2038 cellules, au lieu de 3282 cellules, pour l'ensemble de la détection des régions de couleur peau.

A l'issue de cette première analyse, on s'aperçoit qu'il n'est pas astucieux de réaliser toute la détection en "matériel". En effet, les deux traitements principaux de la détection n'ont pas les mêmes contraintes temps réel : la construction du masque binaire est un traitement au niveau pixel effectuant un grand nombre de calculs de distance (380160), et tire avantage d'une implémentation en "matériel", alors que la mise à jour des représentants et le choix du représentant de la peau n'interviennent qu'une fois par image, ce qui ne justifie pas une implémentation "matérielle". Une implantation en "logiciel" de la deuxième étape de la détection est donc préférable.

IV.1.2. Localisation du visage

Pour ce bloc de traitement, on effectue également deux étapes principales : le filtrage non-linéaire et le filtrage adapté. La première étape est régulière et traite des blocs de pixels indépendants. Ainsi, pour chaque bloc, la somme des pixels du bloc est calculée puis un seuillage est appliqué afin de construire une image sous-échantillonnée et filtrée du masque binaire. Comme pour l'étape de construction du masque binaire, cette étape effectuée un grand nombre de fois tirerait partie d'une réalisation en matériel.
La seconde étape n'est quant à elle pas adaptée à une réalisation "matérielle" du fait de la non régularité des opérations : balayage du masque binaire par des fenêtres de dimensions multiples. Nous préconisons une implantation sous forme logicielle.

IV.1.3. Mémoire du masque binaire

L'algorithme de détection / localisation du visage nécessite la mémorisation du masque binaire des régions de couleur peau qui, pour une image QCIF, est de 6336 pixels. Chaque pixel du masque nécessitant 1 bit pour son codage, deux solutions sont envisageables. La première est la création sur le SoC d'une mémoire de dimension 6336*1bit utilisable uniquement pour ce traitement. La deuxième est d'encoder plusieurs pixels du masque dans un même mot et de façon à utiliser des mémoires standards. Dans ce cas, un parcours astucieux de l'image regroupant les pixels d'un même bloc, permet d'associer un bloc 4*4 à un mot de 16 bits. Une mémoire standard de 396*16 bits est alors suffisante.

Remarque :
Afin d'accélérer le traitement et éviter de mémoriser un masque binaire de 6336 pixels, le sous-échantillonnage de l'image peut être fait avant la détection des régions de couleur peau.

IV.2. Implantation sur la plate-forme IVT

L'IVT est un SoC de STMicroelectronics dédié à la visiophonie [69], il implémente le codec de la recommandation H.263+ sur une architecture multi-processeurs dédiés où les traitements les plus coûteux sont réalisés en matériel. Le choix du partitionnement matériel / logiciel est fait selon les besoins en puissance de calcul et en flexibilité des traitements effectués par le codec (figure 90). Le schéma bloc de l'architecture de l'IVT est fourni en figure 91. Quatre co-processeurs re-programmables sont mis en oeuvre pour :
- fixer la stratégie de codage (MSQ),
- effectuer la recherche des vecteurs mouvement,
- construire le 'bitstream' et effectuer le codage à longueur variable (BSP),
- et prédire les MBs.

Dans une étude menée dans le cadre de cette thèse, nous avons examiné dans quelle mesure il serait possible d'implanter, sur cette plate-forme, les stratégies de codage orientées visage présentées au chapitre précédent. Nous proposons l'implantation suivante :

- l'information de position de visage est exploitée par le MSQ de façon à orienter sa stratégie d'allocation des bits (contrôle de la prédiction et de la quantification), et par l'estimateur de mouvement de manière à modifier la technique de recherche suivant la nature du MB (visage ou arrière plan).

- l'information de position du visage est quant à elle implémentée sur la plate-forme supportant l'IVT. Cette plate-forme dont le schéma fonctionnel est donné en figure 92 est composée d'un FPGA, de mémoires SDRAM et FLASH, d'un processeur hôte, le ST20, d'un IVT et de divers périphériques (entrée caméra, sortie écran, interface ISDN,…). Le partitionnement envisagé pour la détection / localisation du visage est également fournie sur la figure 92.

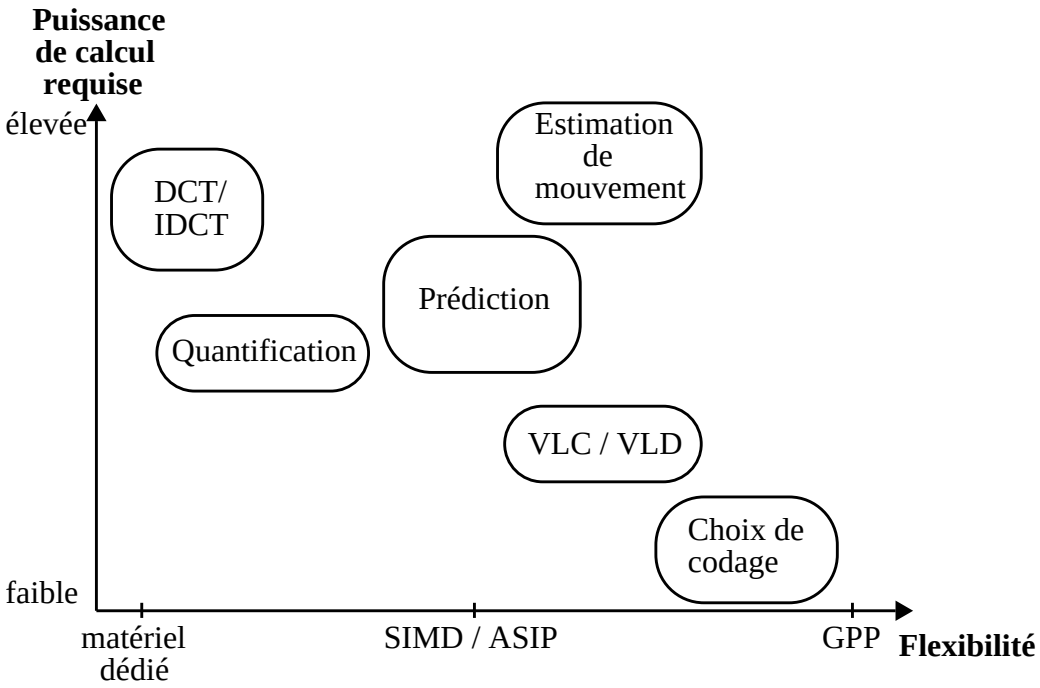


Figure 90: partitionnement retenu sur l'IVT[1]

1. tiré de [69]

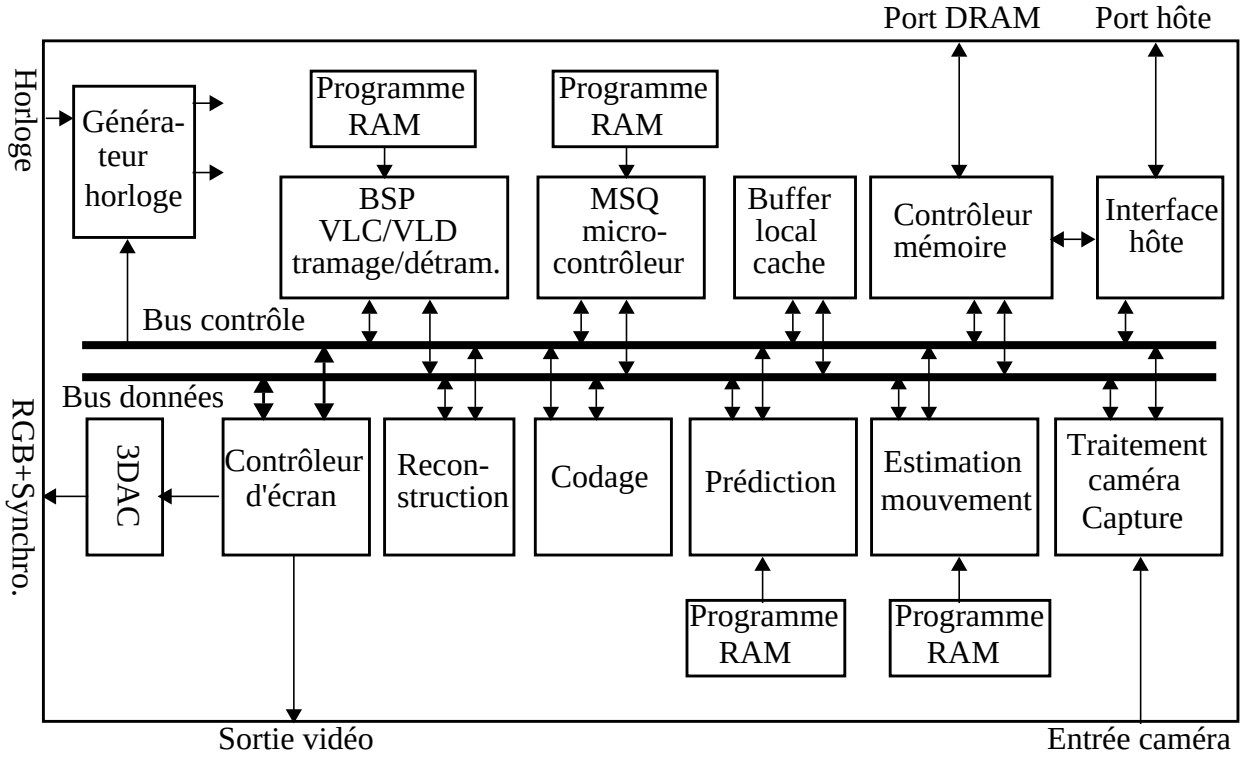


Figure 91: diagramme fonctionnel de l'IVT[1]

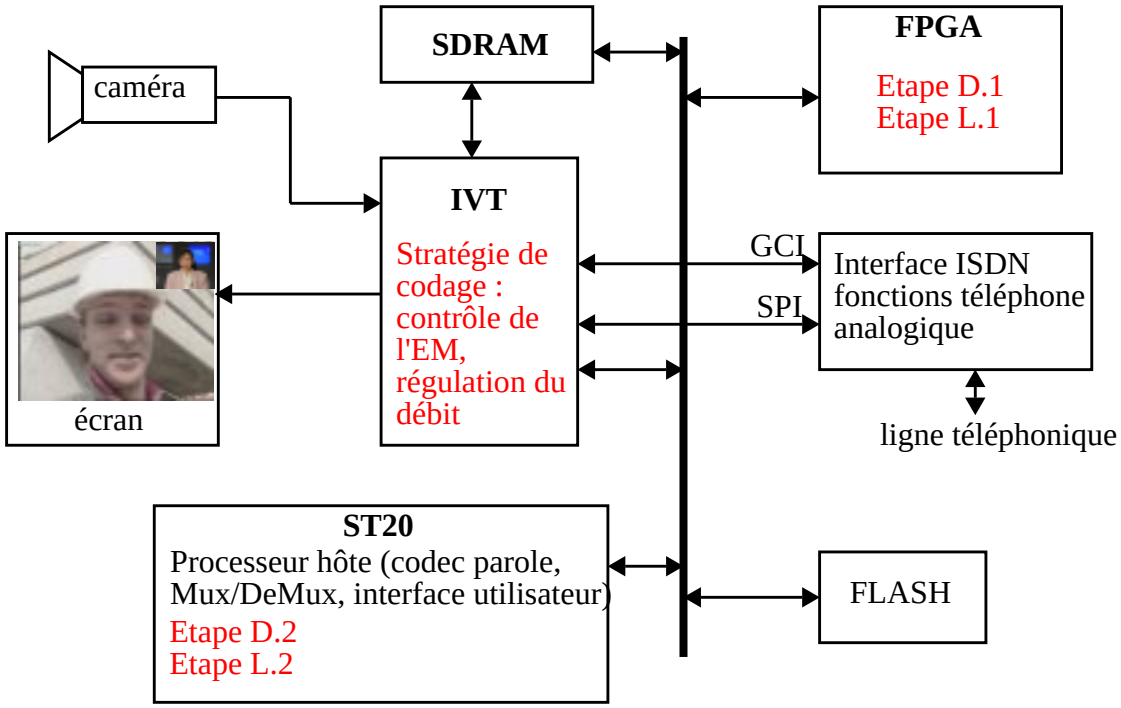


Figure 92: adaptation du schéma fonctionnel de la plateforme IVT au codeur COV

1. tiré de [69].

- Le pré-traitement de détection / localisation est réparti comme suit :

- la construction du masque binaire (étape D.1) est réalisée sur le FGPA selon le découpage en blocs de 16*16 pixels.
- l'accumulation des chrominances selon leur classe d'appartenance est également implémentée sur le FPGA afin de limiter les transferts entre le FPGA et le ST20.
- dès qu'un bloc a été traité, l'étape de filtrage non linéaire (étape L.1) est effectuée sur ce bloc. Le résultat est une valeur binaire représentant l'appartenance ou non du bloc à une région de couleur peau qui est écrit en mémoire. L'avantage du rapprochement des étapes D.1 et L.1 est d'éviter la mémorisation des 6336 pixels du masque binaire. L'information contenue dans le masque binaire utile à la localisation du visage ne nécessite alors plus que la mémorisation de 396 pixels (1 bit), ce qui serait fait sur le processeur hôte (ST20).

- le ST20 prend en charge les étapes moins contraintes et mieux adaptées à une réalisation sous forme logicielle, à savoir, la mise à jour des représentants et la sélection du représentants de la peau pour la prochaine image (étape D.2), et le filtrage adapté (étape L.2).

- Les stratégies de codage :

Comme nous l'avons vu, nous utilisons deux stratégies de codage orientées visage : le contrôle de l'estimation de mouvement et la régulation du débit. Ces deux stratégies sont implantées sur le MSQ. Les deux techniques d'estimation de mouvement sont implantés dans la mémoire programme de l'estimateur de mouvement.

En conclusion, l'implantation concrète du codeur COV sur la plate-forme IVT, bien que tout à fait réaliste, n'a pas été réalisée, car elle aurait nécessité un investissement important des deux parties (STMicroelectronics et France Télécom R&D) afin d'adapter la plate-forme à nos attentes, ce qui dépassait le cadre de cette thèse. En effet, une adaptation matérielle (taille du FGPA) et l'utilisation d'un processeur hôte plus performant (le ST20 est surchargé, l'utilisation, par exemple, d'un ST40 offrirait les ressources nécessaires à la validation du codeur).

V. bibliographie

[1] ‘MPEG-4 video group – generic coding of audio-visual objects’, ISO/IEC JTC1/SC29/ WG11 14496-2, International standard, Atlantic City, 1998.

[2] 'MPEG-7 Applications document V.10', ISO/IEC JTC1/SC29/WG11 N3934, Pise, Janvier 2001.

[3] T. Kanade, ‘Region segmentation : signal vs semantics’, Proc. 4th Int. Joint Conf. on Pattern recognition, pp 95-105, Kyoto, Novembre 1978.

[4] R. Chellappa, C.L. Wilson et S. Sirohey, ‘Human and machine recognition of faces : a survey’, Proceedings of the IEEE, Vol. 83, No.5, pp 705-740, Mai 1995.

[5] G. Chow et X. Li, 'Towards a system for automatic facial feature detection', Pattern Recognition, Vol. 26, No. 12, pp 1739-1755, 1993.

[6] J.W. Roach et J.K. Aggarwal, 'Computer tracking of objects moving in space', IEEE Trans. on Patter Analysis and Machine Intelligence, Vol. 1, No. 2, pp 127-135, 1979.

[7] V. Govindaraju, D.B. Sher, R.K. Srihari et S.N. Srihari, 'Locating human faces in newspaper photographs', Proc. Conf. on Comp. Vision and Pattern Recognition, pp 549-554, 1989.

[8] J.B. Waite et W.J. Welsh, 'An application of active contour models to head boundary location', Proc. British Machine Vision Conf., pp 407-412, Oxford, 1990.

[9] I. Craw, D. Tock et A. Bennett, 'Finding face features', Proc. 2nd European Conf. on Comp. Vision, pp 92-96, 1992.

[10] T.F. Cootes, C.J. Taylor, D.H. Cooper et J. Graham, 'Active shape models – their training and application', Comp. Vision and Image Understanding, Vol. 61, No. 1, pp 38-59, 1995.

[11] A. Jacquin et A. Eleftheriadis, 'Automatic location tracking of faces and facial features in video signal', Int. Work. On Automatic Face and Gesture Recognition, pp 142-147, Zurich, 1995.

[12] T. Kanade, ‘Picture processing system by computer complex and recognition of human faces’, PhD thesis, Department of Information Science, Kyoto University, Novembre 1973.

[13] G.J. Klinker, S.A. Shafer et T. Kanade, 'A physical approach to color image understanding', International Journal of Computer Vision, 4, pp 7-38, 1990.

[14] Y. Sumi et Y. Ohta, 'Detection of face orientation and facial components using distributed appearance modelling', Proc. Int. Work. On Automatic Face and Gesture Recognition, pp 254-259, Zurich, 1995.

[15] A. Zelinksy et J. Heinzmann, 'Real time visual recognition of facial gestures for human computer interaction', Proc. 2nd Int. Conf. on Automatic Face and Gesture Recognition, pp 351-356, Vermont, 1996.

[16] T. Leung, M. Burl et P. Perona, 'Finding faces in cluttered scenes using labelled random graph matching', Proc. 5th Int. Conf. on Comp. Vision, pp 637-644, Cambridge, 1995.

[17] H.P. Graf, E. Cosatto, D. Gibbon, M. Kocheisen et E. Patajan, 'Multimodal system for locating heads and faces', Proc. 2nd Int. Conf. on Automatic Face and Gesture Recognition, pp 88-93, Vermont, 1996.

[18] M.C. Burl and P. Perona,'Recognition of planar object classes', Comp. Vision and Pattern Recognition, San Fransisco, Juin 1996.

[19] K.C. Yow et R. Cipolla, 'Finding initial estimates of human face location', Department of Engineering, University of Cambridge, 1995.

[20] B. Schiele et A. Waibel, 'Gaze tracking based on face color', Proc. Int. Work. On Automatic Face and Gesture Recognition, pp 344-349, Zurich, 1995.

[21] K.C. Yow et R. Cipolla, 'Enhancing human face detection using motion and active contours', Proc. 3rd Asian Conf. on Comp. Vision, Vol. 1, pp 515-522, Hong Kong, 1998.

[22] K. Aas, 'Detection and recognition faces in video sequences', Norsk Regnesentral, 1998.

[23] G.W. Cottrell et M.K. Fleming, 'Face recognition using unsupervised feature extraction', Proc. Int. Conf. on Neural Networks, pp 322-325, Paris, 1990.

[24] G. Burel et D. Carel, 'Detection and localization of faces on digital images', Pattern Recognition Letters, Vol. 15, No. 10, pp 963-967, 1994.

[25] K.K. Sung et T. Poggio, 'Example based learning for view based human face detection', Technical report A.I. Memo 1521, CBLC paper 112, MIT, 1994.

[26] S.H. Lin, S.Y. Kung et L.J. Lin, 'Face recognition / detection by probabilistic decision based neural network', IEEE Trans. on Neural Networks, Vol. 8, No. 1, pp 114-132, 1997.

[27] E. Osuna, R. Freund et F. Girosi, 'Training support vector machines : an application to face detection', Proc. Conf. Comp. Vision and Pattern Recognition, pp 130-136, Puerto Rico, 1997.

[28] R. Feraud, O. Bernier et D. Collobert, 'A constrained generative model applied to face detection', Neural Processing Letters 5, pp 73-81, 1997.

[29] D. Valentin, H. Abdi, A.J. O'Toole et G.W. Cottrell, 'Connectionist models of face processing : a survey', Pattern Recognition, Vol. 27, No. 9, pp 1209-1230, 1994.

[30] H.A. Rowley, S. Baluja et T. Kanade, 'Rotation invariant neural network based face detection', CMU-CS-97-201, Carnegie Mellon University, 1997.

[31] R. Vaillant, C. Monrocq et Y. Le Cun, 'Original approach for the localisation of objects in images', IEEE Proceedings on Vision, Image and Signal Processing, Vol. 141, No. 4, pp 245-250, Aout 1994.

[32] S. Oka, M. Kitabata, Y. Ajioka et Y. Takefuji, 'Grouping complex face parts by nonlinear oscillations', Keio University, 1998.

[33] A.J. Colmenarez et T.S. Huang, 'Face detection with information based maximum discrimination', University of Illinois, 1997.

[34] B. Moghaddam et A. Pentland, 'Probabilistic visual learning for object representation', MIT, 1995.

[35] M.M. Fleck, D.A. Forsyth et C. Bregler, 'Finding naked people', Proc. 4th European Conf. on Comp. Vision, Vol. 2, pp 593-602, Cambridge, 1996.

[36] R. Kjeldsen et J. Kender, 'Finding skin in color images', Proc. 2nd Int. Conf. on Automatic Face and Gesture Recognition, pp 312-318, Vermont, 1996.

[37] Q. Chen, H. Wu et M. Yachida, 'Face detection by fuzzy pattern matching', Proc. 5th Int. Conf. on Computer Vision, pp 591-596, Cambridge, 1995.

[38] Y. Dai et Y. Nakano, 'Face texture model based on SGLD and its application in face detection in a color scene', Pattern recognition, Vol. 29, No. 6, pp 1007-1017, 1996.

[39] C. Gomila et F. Meyer, 'Automatic video object generation tool: segmentation and tracking of persons in real time', Annales des Télécommunications, Vol. 57, No. 5-6, 2000.

[40] M. Hunke et A. Waibel, 'Face locating and tracking for human computer interaction', School of computer science, Carnegie Mellon University.

[41] J. Yang, W. Lu et A. Waibel, 'Skin color modeling and adaptation', ACCV, 1998.

[42] D. Comaniciu et P. Meer, 'Robust analysis of feature spaces: color image segmentation', Department of electrical and computer engineering.

[43] J.-C. Terrillon et S. Akamatsu, 'Comparative performance of different chrominance spaces for color segmentation and detection of human faces in complex scene images', Vision Interface, Canada, Mai 1999.

[44] F. Smeraldi, O. Carmona et J. Bigün, 'Real tima head tracking by saccadic exploration and gabor decomposition', Proceedings of the 5th Int. Workshop on Advanced Motion Control (AMC), Coimbra, pp 684-687, 1998.

[45] F. Meyer et S. Beucher, 'Morphological segmentation', Journal Visual Com. Image Representation, Vol.1, No. 1, pp 21-46, Septembre 1990.

[46] K. Sobottka et I. Pitas, 'Looking for faces and facial features in color images', Department of informatics, University of Thessaloniki.

[47] M. Kampmann et J. Ostermann, 'Automatic adaptation of a face model in a layered coder with an object based analysis-synthesis layer and a knowledge based layer', Signal Processing: Image Communication, Vol. 9, pp 201-220, 1997.

[48] M. Kampmann, 'Segmentation of a head into face, ears, neck and hair for knowledge based analysis-synthesis coding of videophone sequences', http://www.tnt.uni-hannover.de/~kampmann

[49] R. Feraud, O. Bernier, J.E. Viallet et M. Collobert, ‘A fast and accurate face detector for indexation of face images’, 4th Int. Conf. on Automatic Face and Gesture Recognition, Grenoble, 2000.

[50] R.J. Qian et T.S. Huang, 'Object detection using hierarchical MRF and MAP estimation', Proc. Conf. Comp. Vision and Pattern Recognition, pp 186-192, Puerto Rico, 1997.

[51] S. Kawato et J. Ohya, ‘Automatic skin color distribution extraction for face detection and tracking’, 5th Int. Conf. on Signal Processing, Vol.2, pp 1415-1418, Beijin, Aout 2000.

[52] S. Roux et E. Petit, ‘Enhanced H.263 codec for mobile visiophony’, Proc. Int. Conf. on Visualization, Image and Image Processing (IASTED/VIIP), Marbella, Septembre 2001.

[53] S. Roux et E. Petit, ‘Codeur H.263 amélioré pour la visiophonie mobile’, 7iéme journée d’étude et d’échange : COmpression et REprésentation des Signaux Audiovisuels (CORESA), Dijon, Novembre 2001.

[54] S. Roux et E. Petit, ‘Procédé et dispositif de transmission d’une vidéo comportant un visage, en particulier dans un système de visiophonie mobile’, Demande de brevet France Télécom R&D, B 01/0462 FR – FZ/DD, Avril 2001.

[55] B.G. Batchelor, 'Classification and data analysis in vector spaces', Pattern recognition, Ple-

num Press, NY, pp 67-116, 1978.

[56] C.W. Therrien, 'Decision, esimation and classification', Wiley, NY, 1989.

[57] Bow, Sing Tze, ' Pattern recognition', Marcel Dekker, NY, 1984.

[58] J.T. Tou et R.C. Gonzalez, 'Pattern recognition principles', Addison Wesley, Reading, MA, 1974.

[59] N. Moreau, 'Techniques de compression des signaux', Masson, 1994.

[60] J. Mac Queen, 'Some methods for classification and analysis of multivariante data', Proc. 5th Berkeley Symp. on probability and statistics, University of California Press, Berkeley, 1967.

[61] Y. Linde, A. Buzo et R.M. Gray, 'An algorithm for vector quantizer design', IEEE Trans. Com., Vol. 28, Janvier 1980.

[62] V.A. Christopoulos, P. De Muynck et J. Cornelis, 'Contour simplification for segmented still image and experimental results', Signal Processing: Image Communication, Vol. 14, pp 335-357, 1999.

[63] J. Hartung, A. Jacquin, J. Pawlyk, J. Rosenberg, H. Okada et P.E. Crouch, ‘Object oriented H.263 compatible video coding platform for conferencing applications’, IEEE Journal on selected area in communications, Vol. 16, No. 1, pp 42-54, Janvier 1998.

[64] C. De Vleeschouwer et B. Macq, 'Content based and perceptual bit allocation using matching pursuits', Signal Processing: Image Communication, Vol. 16, pp 611-626, 2001.

[65] M. Hötter, 'Optimization and efficiency of an object oriented analysis-synthesis coder', IEEE Trans. on Circuits and Systems for Video tech., Vol. 4, No. 2, pp 181-194, Avril 1994.

[66] P. Salembier et al., 'Segmentation based video coding system allowing the manipulation of objects', IEEE Trans. on Circuits and Systems for Video Tech., Vol. 7, No. 1, pp 60-74, Février 1997.

[67] H. Luo, A. Eleftheriadis et J. Kouloheris, ‘Statistical model based video segmentation and its application to very low bit rate video coding’, Signal Processing : Image Communication, Vol. 16, pp 33-352, 2000.

[68] Altera Flex 10k, ‘Embedded Programmable Logic Device Family V4.1’, Mars 2001, http://www.altera.com

[69] M. Harrand et al., ‘A single chip CIF 30 Hz, H.261, H.263 and H.263+ video encoder / decoder with embedded display controller’, IEEE Journal of Solid State Circuits, Vol. 34, No. 11, pp 1627-1633, Novembre 1999.

CONCLUSION

L'objectif de cette thèse était d'étudier la faisabilité du service de visiophonie sur les terminaux mobiles de prochaine génération. Pour cela, nous avons défini une méthodologie de conception de systèmes embarqués basée sur une modélisation de haut niveau d'abstraction.

Comme nous l'avons montré, cette méthodologie de conception s'articule autour de deux axes, algorithmique et architectural, et repose sur une spécification au niveau système. Tout d'abord avec une description algorithmique en langage C et à l'aide d'outils de "profiling", nous avons obtenu des éléments de complexité relatif à une réalisation logicielle sur circuit intégré programmable (DSP, RISC, GPP). Puis, l'utilisation d'une description système SDL alliée à la mise en oeuvre de l'outil ArchiMate d'exploration d'architecture, a abouti à une répartition adéquate matérielle / logicielle des blocs de traitement du système. Ces deux approches complémentaires induisent un cycle de conception itératif rapide entre la spécification système et la description architecturale matérielle / logicielle. De plus, à chaque itération des mesures de performance ont été établies :
- au niveau algorithmique : estimations de la complexité, de la bande passante ("profiling" du modèle C) et de la qualité vidéo sur l'application de visiophonie,
- au niveau architectural : réalisation, dans l'outil ArchiMate, d'une mesure du taux d'activité pour l'estimation de la dissipation d'énergie des bus de données (opérateur de mesure du taux d'activité co-développé en collaboration avec la société Valiosys/Arexsys). Dans notre contexte de visiophonie mobile, nous avons testé cette méthodologie sur un codeur de type H.263 (recommandation UIT-T).

- Au niveau algorithmique nous avons identifié les options de codage normalisées (de la recommandation H.263+) les mieux appropriées à la visiophonie mobile :
- le mode de prédiction avancé (AP) ou le filtrage dans la boucle de codage (H.263 annexe J) efficaces pour réduire les effets blocs souvent présent à très faible débit,
- les options de résistance aux erreurs (découpage en ‘slice’, partitionnement en groupe de données indépendantes, codage à longueur variable réversible,…).

Puis, l’étude de la complexité du codeur H.263 a permis d'identifier le bloc de traitement le plus critique, l'estimateur de mouvement, ce qui a motivé une étude comparative de différents estimateurs de mouvement rapides aboutissant au choix de deux d'entre-eux :
- la méthode en "4 pas" bien adaptée à la nature de la vidéo, notamment pour la prédiction du mouvement aléatoire de l'arrière plan,
- et la méthode du gradient bien adapté à la prédiction du mouvement du visage.

Ces deux estimateurs de mouvement se distinguent par leur efficacité, leur faible complexité moyenne, leur simplicité de mise en oeuvre, et conduisent à une complexité moyenne de 250 MOPS pour le codage vidéo à 15Hz d'une séquence de résolution QCIF.

Par ailleurs, afin d'optimiser ce codeur pour un fonctionnement à très bas débit (< 40 kbit/s), nous avons proposé un codeur tenant compte de l'information sémantique contenue dans la vidéo, dans l'esprit de la norme MPEG-4. Nous avons ainsi réalisé, à la fois un algorithme adaptatif pour l'extraction du visage, et un nouveau schéma de codage pour la compression vidéo à très bas débit :

- l'algorithme de détection du visage proposé tire parti de l'hypothèse de localisation de la couleur "peau" dans une région étroite de l'espace des chrominances. Cette nouvelle approche adaptative testée sur plusieurs séquences classiques de visiophonie est robuste (convergence rapide et adaptation aux variations locales de chrominance) et de faible complexité (1 % de la complexité d'un codeur H.263).

- le schéma de codage H.263 a donc été modifié à deux niveaux :
    - au niveau de l'estimateur de mouvement par la mise en oeuvre de deux estimateurs de mouvement rapides contrôlés par la détection du visage (méthode du gradient sur le visage et méthode en "4 pas" sur l'arrière plan),
    - au niveau de la régulation du débit par l'utilisation d'une deuxième fréquence image pour coder l'arrière plan et pour allouer plus de "bits" pour le codage du visage.

Ce codeur orienté visage conduit à une qualité vidéo subjective nettement supérieure à celle d'un schéma de codage "classique", et ce, sans augmenter la complexité globale du codage.

• Au niveau architectural, nous avons tout d'abord spécifié en langage SDL le codeur H.263 dans le but d'explorer la répartition "matérielle / logicielle". En effet, la description SDL des blocs de traitement du codeur est le point d'entrée de l'outil ArchiMate.

La première implémentation des blocs ciblait uniquement une réalisation sous forme "matériel" car sous ArchiMate nous ne disposions pas des IPs "coeur de processeur". Cette implémentation a mis en évidence certaines limitations de l'outil ArchiMate. Celui-ci a effectivement généré un modèle "matériel" en C++, malheureusement, ce modèle n'a pu être compilé sous *gnu g++* pour des raisons liées à la déclaration de tableaux de grande taille. Pour des applications orientées flot de données, il faudrait pouvoir mettre en oeuvre des modèles "matériels" de mémoires afin de pallier ce problème.

Sur la base des résultats de l'étude algorithmique, un partitionnement "matériel / logiciel" a été établi au niveau d'ArchiMate. Dans ce partitionnement, seul le calcul de distorsion de l'estimation de mouvement est réalisé en "matériel", le reste des traitements est réalisé en logiciel sur une cible multi-processeurs. Ce partitionnement offre la plus grande flexibilité pour l'implantation de différentes versions du codeur vidéo (autres profils,...), et, en particulier, si on souhaite rapidement tester de nouveaux schéma de codage (par exemple le codeur orienté visage). Ainsi, nous avons fait le choix d'isoler l'estimateur de mouvement, que nous avons modélisé sous ArchiMate pour une cible "matérielle". Cette modélisation fait apparaître les flots de données entre les blocs constituant la recherche du mouvement : mémoire des images, séquenceur de la recherche du vecteur mouvement d'un MB dans une fenêtre de recherche, mémoire du MB, mémoire de la fenêtre de recherche et opérateur de calcul de distorsion. Le modèle "matériel" généré par ArchiMate a cette fois ci été simulé entièrement, produisant une simulation réaliste du flot de données. Fort de cette modélisation, nous avons pu tester la fonction de mesure du taux d'activité co-développée avec la société Valiosys/Arexsys. Nous avons ainsi obtenu des in-

formations sur la consommation des bus du bloc de l'estimation de mouvement pour différentes architectures (comme par exemple l'utilisation d'une mémoire interne ou externe pour les images).

D'autre part, cette étude a conduit à la recommandation de quelques évolutions des fonctionnalités de l'outil ArchiMate :

- la taille du bus ne doit pas être figée (aujourd'hui, bus 32 bits uniquement) :
afin d'obtenir la flexibilité nécessaire à la spécification de l'architecture d'un système et de satisfaire aux contraintes de conception, il faut pouvoir ajuster la largeur des bus.

- la sélection d'un mode de codage des données doit être possible :
l'outil devrait proposer au concepteur différents types de représentation de données comme par exemple la représentation en complément à deux, en signe - valeur absolue, en base 'n', ..., ou encore des modes de codage comme le codage en bus inversé,...

- l'implémentation des protocoles de communication inter-blocs doit être adaptable à la nature des données :
nous proposons les deux possibilités suivantes :
  - les données et les signaux de contrôle partagent le même bus (architecture type Von Neumann),
  - les données et les signaux de contrôle utilisent deux bus distincts (architecture type Harvard).

Nous avons donc testé cette méthodologie à l'aide de simulations sur un outil commercial en vue d'effectuer un prototypage "matériel". De plus, pour une validation complète du système, la méthodologie de conception, schématisée sur la figure 93, intègre une étape supplémentaire de prototypage matériel.
Par ailleurs, il apparaît aujourd'hui que la conception de tels systèmes va nécessiter de plus en plus l'utilisation d'outils d'exploration d'architectures facilitant la manipulation des IPs (coeurs de processeur, opérateurs dédiés, ...) ainsi que leur interfaçage, et conduisant au développement supplémentaire d'un nombre réduit de blocs "matériels" spécifiques. D'autre part, pour les SoCs embarqués dans les terminaux mobiles, une architecture essentiellement composée de processeurs programmables (DSP + RISC), semble mieux adaptée par rapport à une implémentation application-spécifique. En effet, une telle architecture apporte plus de flexibilité (adaptable à l'évolution rapide des applications et des normes), permet de réduire le "time-to-market", notamment par l'utilisation de langages de haut niveau et de compilateurs efficaces, et offre la possibilité de supporter un OS ou encore une machine virtuelle type Java. Simuler dans son ensemble une telle architecture relève encore aujourd'hui de la gageure, mais sera sans doute un enjeu important pour les années à venir.

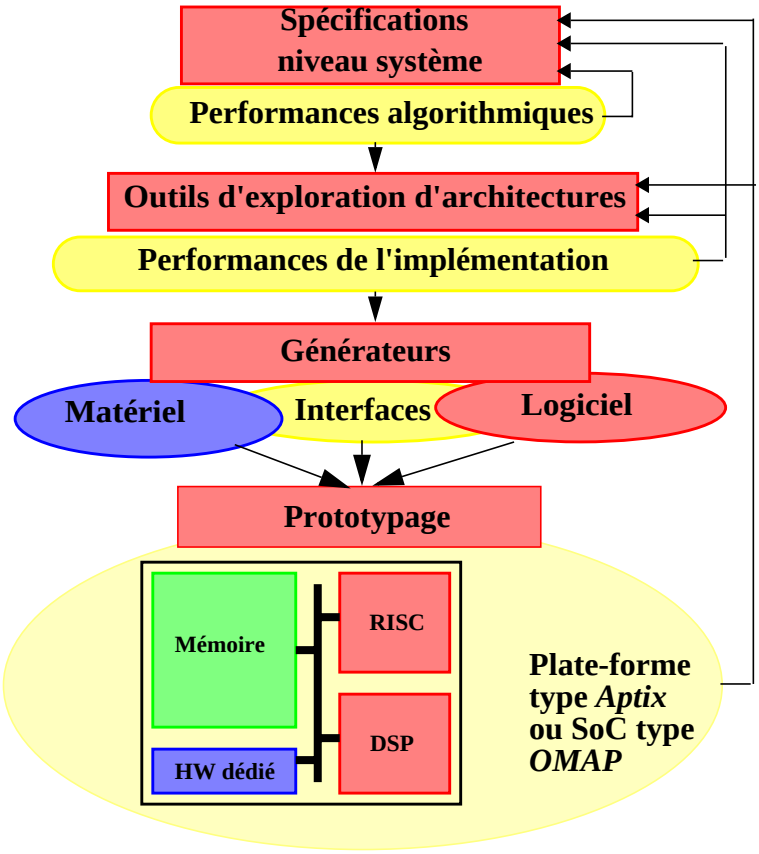
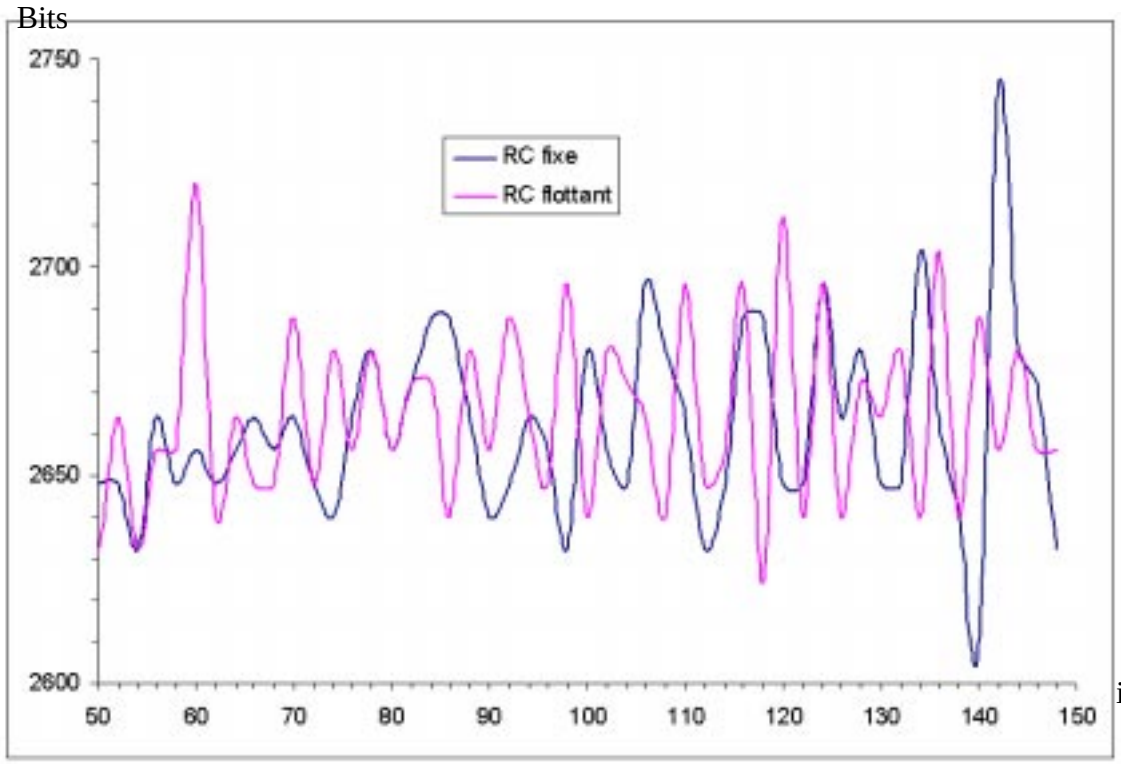


Figure 93 : Schéma de conception de systèmes embarqués

Annexe B.1

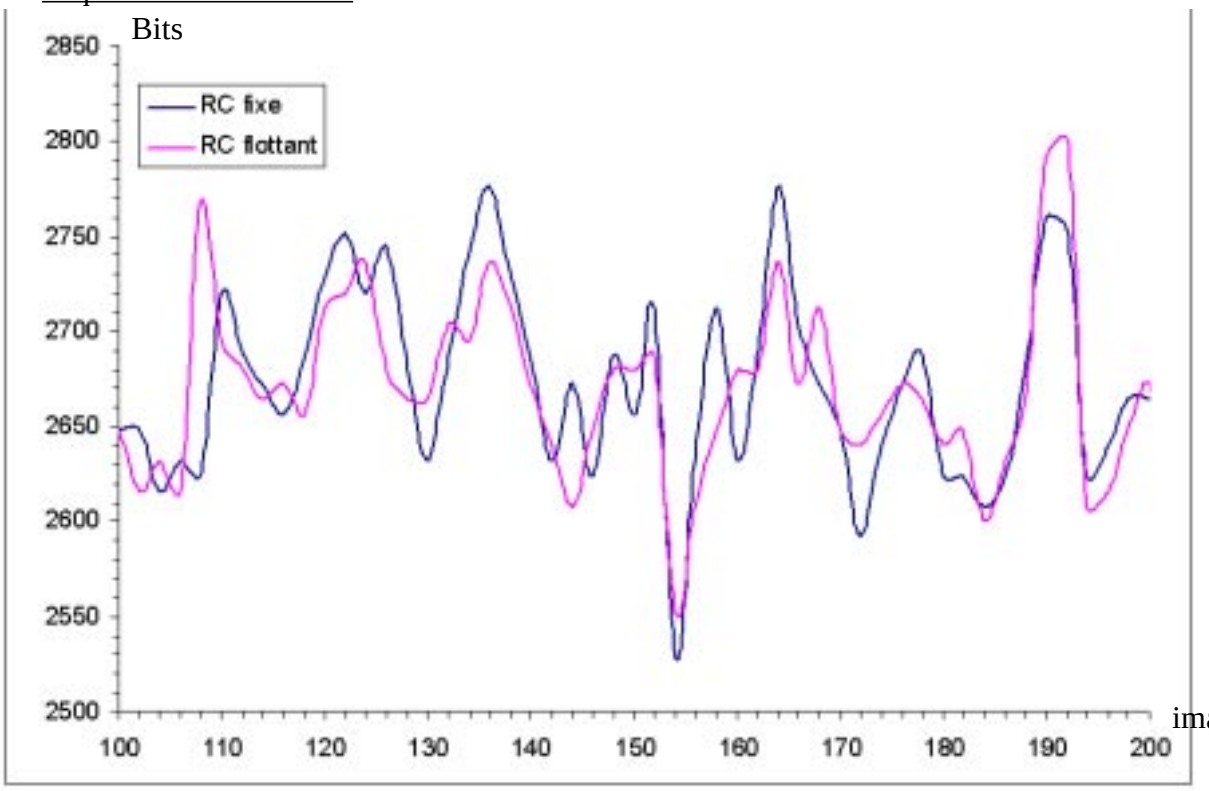
Recommandation TMN8 : implantation virgule fixe vs virgule flottante

Séquence Carphone



image

Séquence Miss America



image

Annexe B.2

Qualité en fonction du profil pour différentes séquences vidéo

Miss America, Akiyo et claire: scéne de visiophonie avec arrière plan fixe.
Foreman: visiophonie avec mouvement important du visage et déplacement de la caméra.
Coastguard: vidéo sans visage avec un mobile à déplacement rapide, un autre à déplacement lent et un travelling de caméra.
Container: vidéo sans visage avec un mobile à déplacement lent.
Foot: vidéo sans visage avec une activité de mouvement importante.

Séquence (Nombre d'images)		**Sans option**	**UMV**	**AP**	**L.1**	**L.2**	**L.3**	**MP4SP**
Miss America (75)	images codées	73	73	73	73	73	73	73
	Y	39.9	40.01	40.15	40.09	39.8	40.87	39.87
	Cr	39.59	39.72	39.75	39.92	39.51	40.69	39.5
	Cb	39.63	39.72	39.76	39.88	39.62	40.18	39.54
Akiyo (150)	images codées	146	146	146	146	146	146	146
	Y	38.78	38.74	39.06	38.53	38.47	40.34	38.75
	Cr	42.07	42.01	42.25	42.09	41.75	43.59	41.94
	Cb	40.25	40.18	40.54	40.33	39.89	42.44	40.19
Claire (150)	images codées	146	147	146	148	146	146	147
	Y	39.68	39.71	40.01	39.6	39.48	40.86	39.68
	Cr	42.68	42.79	42.86	42.73	42.64	44.04	42.7
	Cb	40.45	40.47	40.66	40.21	40.25	41.74	40.36
Foreman (130)	images codées	125	125	124	125	125	124	125
	Y	29.45	29.94	30.03	29.11	29.64	30.58	29.7
	Cr	36.07	37.32	36.26	37.85	37.12	37.57	36.82
	Cb	35.57	36.32	35.53	37.15	36.35	36.85	36.18

Séquence (Nombre d'images)		**Sans option**	**UMV**	**AP**	**L.1**	**L.2**	**L.3**	**MP4SP**
Container (150)	images codées	144	144	144	144	144	144	144
	Y	33.71	33.71	33.73	33.34	33.46	35.43	33.65
	Cr	38.91	38.92	39.49	39.38	38.76	40.1	38.68
	Cb	39.52	39.54	39.49	40.09	39.31	40.53	39.36
Coas-tguard (150)	images codées	146	146	146	146	145	146	146
	Y	27.47	27.61	27.75	27.07	27.41	28.5	27.46
	Cr	39.07	41.48	39.87	41.71	71.06	71.97	40.42
	Cb	37.98	38.67	38.54	39.51	38.57	39.52	38.86
Foot (75)	images codées	46	49	39	33	46	22 *	37
	Y	23.66	23.66	23.81	23.94	23.69	23.68	23.84
	Cr	31.66	31.48	31.26	33.6	31.54	31.32	31.17
	Cb	28.63	28.45	28.33	31.45	28.52	28.03	28.19

(*) Dans ce cas la régulation de débit est mise en échec. Le débit obtenu est de 30 kbit/s au lieu de 40 kbit/s.

Annexe B.3

Complexité en fonction de l'estimateur de mouvement

Estimations réalisées sur 149 images de la séquence Miss America (30 Hz) codées à 15 Hz avec un débit cible de 40 kbit/s (régulation du débit TMN8). Le nombre d'images codées est de 72 P et 1 I pour toutes les techniques présentées.

complexité / profil		FS spirale	3SS	4SS	GDS	2DLOGS
Million Instructions/image		**77.54**	**31.29**	**31.12**	**31.7**	**31.52**
complexité arithmétique*		***41.46***	***17.4***	***17.08***	***17.37***	***17.3***
	Add*	14.96	6.29	6.16	6.27	6.24
	Sub*	7.3	0.98	0.79	0.91	0.85
	Mult*	0.73	0.66	0.66	0.65	0.66
	Div*	0.05	0.05	0.05	0.05	0.05
	Shift*	12.74	9.14	9.3	9.27	9.33
	Autres(inc.,...)	0.63	0.31	0.51	0.51	0.51
	Cmp* (abs)	5.68	0.28	0.12	0.22	0.17
Branchement / test		9.25	4.92	5.03	5.06	5.06
Load / Store		21.72	7.43	7.35	7.57	7.48
Conversion de type		0.41	0.34	0.34	0.33	0.34
Autres instructions		4.07	0.89	0.81	0.86	0.83
Bande passante		**28.24**	**11.96**	**11.66**	**11.83**	**11.8**
lecture		24.17	11.07	10.85	10.97	10.97
écriture		4.07	0.89	0.81	0.86	0.83

Annexe B.4

Qualité en fonction de l'estimateur de mouvement pour différentes séquences vidéo

Séquence (Nombre d'images)		**FS**	**3SS**	**4SS**	**GDS**	**2DLOG**
Miss America (75)	images codées	73	73	73	73	73
	Y	39.87	39.81	39.78	39.86	39.81
	Cr	39.54	39.49	39.43	39.53	39.47
	Cb	39.57	39.58	39.62	39.63	39.59
Akiyo (150)	images codées	146	146	146	146	146
	Y	38.78	38.82	38.82	38.79	38.82
	Cr	42.31	42.32	42.32	42.3	42.32
	Cb	40.59	40.65	40.65	40.7	40.65
Claire (150)	images codées	146	146	146	146	146
	Y	39.65	39.64	39.61	39.64	39.62
	Cr	42.81	42.65	42.74	42.62	42.75
	Cb	40.39	40.37	40.37	40.31	40.38
Foreman (130)	images codées	125	124	124	124	124
	Y	29.52	28.99	29.15	29.3	29.23
	Cr	35.89	34.88	36	35.97	36.05
	Cb	35.44	34.68	35.39	35.37	35.5
Container (150)	images codées	144	144	144	144	144
	Y	33.81	33.79	33.78	33.78	33.8
	Cr	39.03	39.04	38.99	39.02	39.09
	Cb	39.62	39.6	39.66	39.62	39.69

Séquence (Nombre d'images)		**FS**	**3SS**	**4SS**	**GDS**	**2DLOG**
Coas-tguard (150)	images codées	146	145	144	143	144
	Y	27.59	27.44	27.52	27.48	27.52
	Cr	39.02	39.69	39.68	40.34	39.84
	Cb	38.31	38.37	38.68	38.6	38.61
Foot (75)	images codées	43	40	40	39	40
	Y	23.72	23.69	23.69	23.73	23.71
	Cr	31.68	31.87	31.94	31.88	31.97
	Cb	28.66	28.66	28.71	28.7	28.67

Annexe B.5

Résultat de simulation de l'exemple ping pong

- Signaux p1top2 et p2top1

p1top2 = B’00000000000000000000000000000000’
p2top1 = B’00000000000000000000000000000000’
date : 0
p1top2 = B’00000000000000000000000000000001’
date : 2
p1top2 = B’00000000000000000000000000001010’
date : 12
p2top1 = B’00000000000000000000000000001000’
date : 30
p1top2 = B’00000000000000000000000000000001’
date : 38
p1top2 = B’00000000000000000000000000001001’
date : 48
p2top1 = B’00000000000000000000000000000000’
date : 56
p2top1 = B’00000000000000000000000000000111’
date : 66
p1top2 = B’00000000000000000000000000000001’
date : 74
p1top2 = B’00000000000000000000000000001000’
date : 84
p2top1 = B’00000000000000000000000000000000’
date : 92
p2top1 = B’00000000000000000000000000000110’
date : 102
p1top2 = B’00000000000000000000000000000001’
date : 110
p1top2 = B’00000000000000000000000000000111’
date : 120
p2top1 = B’00000000000000000000000000000000’
date : 128
p2top1 = B’00000000000000000000000000000101’
date : 138
p1top2 = B’00000000000000000000000000000001’
date : 146
p1top2 = B’00000000000000000000000000000110’
date : 156
p2top1 = B’00000000000000000000000000000000’
date : 164
p2top1 = B’00000000000000000000000000000100’
date : 174
p1top2 = B’00000000000000000000000000000001’
date : 182
p1top2 = B’00000000000000000000000000000101’

date : 192
p2top1 = B’00000000000000000000000000000000’
date : 200
p2top1 = B’00000000000000000000000000000011’
date : 210
p1top2 = B’00000000000000000000000000000001’
date : 218
p1top2 = B’00000000000000000000000000000100’
date : 228
p2top1 = B’00000000000000000000000000000000’
date : 236
p2top1 = B’00000000000000000000000000000010’
date : 246
p1top2 = B’00000000000000000000000000000001’
date : 254
p1top2 = B’00000000000000000000000000000011’
date : 264
p2top1 = B’00000000000000000000000000000000’
date : 272
p2top1 = B’00000000000000000000000000000001’
date : 282
p1top2 = B’00000000000000000000000000000001’
date : 290
p1top2 = B’00000000000000000000000000000010’
date : 300
p2top1 = B’00000000000000000000000000000000’
date : 308

- Distance de Hamming sur le signal "to_block2"

p_p1 1 1
date : 2
p_p1 3 4
date : 12
p_p1 3 7
date : 38
p_p1 1 8
date : 48
p_p1 1 9
date : 74
p_p1 2 11
date : 84
p_p1 2 13
date : 110
p_p1 2 15
date : 120
p_p1 2 17
date : 146
p_p1 3 20
date : 156
p_p1 3 23

date : 182
p_p1 1 24
date : 192
p_p1 1 25
date : 218
p_p1 2 27
date : 228
p_p1 2 29
date : 254
p_p1 1 30
date : 264
p_p1 1 31
date : 290
p_p1 2 33
date : 300

- Distance de Hamming sur le signal "to_block1"

p_p2 1 1
date : 30
p_p2 1 2
date : 56
p_p2 3 5
date : 66
p_p2 3 8
date : 92
p_p2 2 10
date : 102
p_p2 2 12
date : 128
p_p2 2 14
date : 138
p_p2 2 16
date : 164
p_p2 1 17
date : 174
p_p2 1 18
date : 200
p_p2 2 20
date : 210
p_p2 2 22
date : 236
p_p2 1 23
date : 246
p_p2 1 24
date : 272
p_p2 1 25
date : 282
p_p2 1 26
date : 308

ANNEXE C.1

Complexité de la détection

Le tableau ci-dessous donne les complexités de différentes techniques de construction de 4 classes dans l'espace des chrominances. Règle du plus proche voisin avec la norme 1, règle du plus proche voisin avec la norme 2 et algorithme itératif. Les estimations sont données en kilo-instructions par image pour 381 images de la séquence Carphone.

		Norme 1	Norme 2	Itératif
Kilo Instructions / image		**1293.1**	**1300.4**	**6188.6**
complexité arithmétique*		***253.6***	***329.6***	***1532.5***
	Add*	145.9	171.2	744.8
	Sub*	50.7	50.7	279.6
	Mult*	57	107.7	508.2
	Shift	228.7	228.7	1088.3
	Autres (inc,...)	57	57	314.5
Branchement / test		265.1	208.7	1101.8
Load / Store		418.8	457.2	2102.7
Conversion de type		6.4	6.4	35.4
Autres instructions		63.5	12.8	13.4
Bande passante (kB/image)		**859.1**	**968.9**	**4407.6**
	lecture	804.7	890	4002.6
	écriture	54.4	78.9	405

Remarque : l'instruction "shift" n'est pas contabilisée dans les instructions arithmétiques car elle correspond essentiellement à un calcul d'indice de tableau.

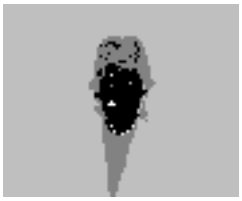
ANNEXE C.2

Qualité de la détection en fonction de la méthode retenue

La qualité de la segmentation orientés visage est illustrée sur 4 séquences vidéo typique de scène de visiophonie.

Règle du plus proche voisin utilisant la norme 1


Carphone


Claire


Grandma


Silent

Règle du plus proche voisin utilisant la norme 2


Carphone


Claire


Grandma

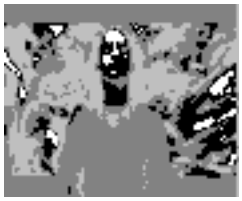

Silent

Algorithme itératif


Carphone


Claire


Grandma


Silent

ANNEXE C.3

Convergence de l'algorithme adaptatif de détection

La convergence de l'algorithme adaptatif est illustrée ci-dessous sur trois séquences.

Séquence Claire

image 1 image 2 image 3

image 21

Séquence Foreman

image 1 image 2

image 21

Séquence Miss America

image 1 image 2 image 3

image 21

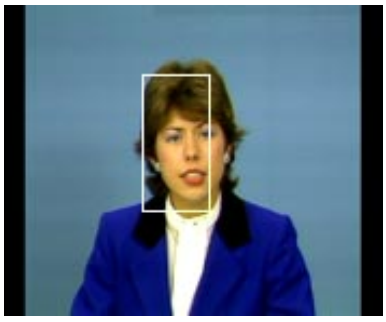
ANNEXE C.4

Rectangle circonscrit au visage

Les 4 images ci-dessous donnent le résultat de la construction du rectangle circonscrit au visage sur 4 scènes de visiophonie.

Akiyo

Carphone



Claire

Foreman

ANNEXE C.5

Complexité de la détection / localisation

Les estimations sont données en kilo-instructions par image pour 381 images de la séquence Carphone.

		Norme 1	Norme 2	Itératif	Adaptatif
Kilo Instructions / image		**1820.9**	**1934.7**	**6992.2**	**1526.8**
complexité arithmétique		***351.9***	***463.6***	***1697***	***289.5***
	Add	232.9	275.3	876.3	182.7
	Sub	55.6	55.2	286	55.7
	Mult	63.4	133.1	534.8	51.1
	Shift	398.6	420.6	1344.4	348.9
	Autres (inc,...)	76.6	74.9	340.6	89.1
Branchement / test		395	351.9	1283.9	332.7
Load / Store		499.7	573	2240.6	449.9
Conversion de type		26.4	24.4	59.2	8.9
Autres instructions		72.7	24.4	26.5	7.8
Bande passante (kB/image)		**941**	**1098.1**	**4555.7**	**650.3**
	lecture	885.8	1013.1	4144.5	527.6
	écriture	55.2	85	411.2	122.7

ANNEXE C.6

Qualité du codeur COV pour différentes séquences de visiophonie

Le rapport signal à bruit est donné pour la région du visage.

Séquence	SNR	FS	COV
Miss America	Y	36.38	37.25
	Cr	37.04	38.18
	Cb	37.22	38.09
Akiyo	Y	35.19	35.78
	Cr	39.49	40.13
	Cb	39.02	40.19
Claire	Y	34.3	35.28
	Cr	37.6	38.56
	Cb	36.2	37.32
Foreman	Y	29.56	31.03
	Cr	34.36	35.69
	Cb	35.51	37.05

ANNEXE C.7

Qualité subjective du COV

Les trois séquences vidéo ci-aprés illustrent l'intérêt d'utiliser un codeur orienté visage plutôt qu'un codeur "classique" (avec une recherche exhaustive des vecteurs mouvement) pour le codage visiophonique. En effet, pour les séquences comportant peu de mouvement (Akiyo et Claire), la qualité subjective est identique, pour les séquences comportant beaucoup de mouvement (Foreman), la qualité subjective est sensiblement améliorée.

Séquence Akiyo (image 109)

COV

FS

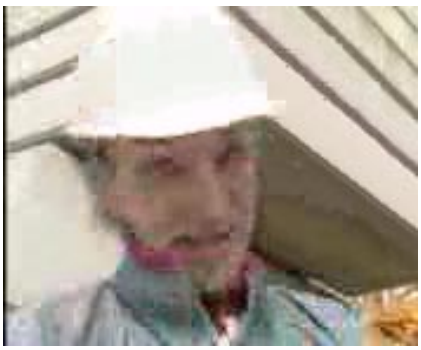
Séquence Foreman

COV

image 120

image 121

FS



Du fait de la rapidité de mouvement de Foreman les contours du rectangle circonscrit au visage apparaissent trés marqués sur l'image 120. Cependant, l'amélioration de la qualité sur le visage est telle que la qualité subjective offerte par le COV et nettement meilleure que celle obtenue avec un codeur "classique".

Séquence Claire (image 125)

COV

FS

Glossaire

2D-LOG : two Dimensions LOGarithmic search
3SS : three Step Search
4SS : four Step Search
ACE : Advanced Coding Efficiency
AIC : Advanced Intra Coding mode
AIVLC : Alternative Inter VLC
ALU : Arithmetic and Logic Unit
AMR : Adaptive MultiRate codec
AP : Advanced Prediction mode
ARTS : Advanced Real Time Simple
ASEI : Additional Supplemental Enhancement Information
ASIC : Applied Specific Integrated Circuit
ASIP : Applied Specific Instruction set Processor
ATM : Asynchronous Transfer Mode
CCIR : Commité Consultatif International de Radiodiffusion
CIF : Common Intermediate Format
CM : Compensation de Mouvement
COV : Codeur Orienté Visage
CP : Core Profile
CPMVM : Continuous Presence Multipoint and Video Multiplex
CPU : Central Processor Unit
CSA (CS) : Cross Search Algorithm
DCT : Discrete Cosine Transform
DF : Deblocking Filter mode
DPS : Data Partitioned Slice
DRAM : Dynamic Random Access Memory
DSP : Digital Signal Processor
EM : Estimation du (Estimateur de) mouvement
ERPS : Enhanced Reference Picture Selection
FDD : Frequency Division Duplex
FEC : Forward Error Correction mode
FPFA : Field Programmable Function Array
FPGA : Field Programmable Gate Array
FS : Full Search

GDS : Gradient Descent Search

GMC : Global Motion Compensation

GOB : Group Of Block

GOV : Group Of Video objects

GPRS : General Packet Radio Service

GSM : Global System for Mobile communications

HDL : Hardware Description Language

HW : HardWare

IP : Intellectual Property

ISD : Independent Segment Decoding

ISO/IEC : International Standardization Organization / International Electro-technical Commission

IRAM : Intelligent RAM

JPEG : Joint Picture Expert Group

MB : MacroBloc

MIPS : Millions d'Instruction Par Seconde

MMACS : Million de Multiplication-Accumulations par Seconde

MOPS : Millions d'Opérations Par Seconde

MP3 : MPEG Layer 3

MPEG : Moving Picture Expert Group

MQ : Modified Quantization mode

NTSC : National Television Standard Committee

OBMC : Overlapped Block Motion Compensation

OSA (OS) : Orthogonal Search Algorithm

PAL : Phase Alternating Line

PDA : Personal Digital Assistant

PIP : Picture In Picture

PCB : Printed Circuit Board

Q/IQ : Quantification / Inverse Quantification

QCIF : Quarter Common Intermediate Format

RISC : Reduced Instruction Set Computer

RNIS : Reseau Numérique à Intégration de Service

RPR : Reference Picture Resampling

RPS : Reference Picture Selection

RRU : Reduced Resolution Update mode

RTL : Register Transfer Level

RTOS : Real Time Operating System

SAC : Syntax based Arithmetic Coding

SA-DCT : Shape Adaptive Discrete Cosine Transform

SDL : System Description Language

SEI : Supplemental Enhancement Information

SIMD : Simple Instruction Multiple Data

SMS : Short Message Service

SNR : Signal to Noise Ratio

SoC : System on a Chip

SP : Simple Profile

SRAM : Static Random Access Memory

SS : Slice Structured mode

SSP : Simple Scalable Profile

SW : SoftWare

TDD : Time Division Duplex

TMN : Test Model Near term

UIT-T (ITU-T) : Union Internationale des Télécommunications – groupe des Télécommunications

UMTS : Univeral Mobile Telecommunication System

UMV : Unrestricted Motion Vector mode

VHDL : Very large scale integrated circuit High Description Language

VLC : Variable Length Coding

VLIW : Very Long Instruction Word

VLSI : Very Large Scale Integrated circuit

VO : Video Object

VOL : Video Object Layer

VOP : Video Object Plane

VOP : Video Object Plane

VS : Visual Object Sequence

VTR : Video Tape Recorder

WAP : Wireless Application Protocol

Résumé

Cette thèse traite de la faisabilité de l'intégration d'un service de visiophonie sur les terminaux mobiles de prochaine génération au travers d'une méthodologie de conception des systèmes multimedia embarqués.
Nous commençons par situer le contexte de l'étude, les réseaux 3G, les terminaux mobiles, les processeurs pour le codage vidéo, ainsi que les normes de codage des groupes ISO/MPEG et UIT-T.
Puis, nous proposons une méthodologie de conception au niveau système prenant en compte les contraintes de l'embarqué, en particulier, l'autonomie du terminal que nous appliquons au codeur vidéo H.263 (recommandation UIT-T). Cette méthodologie s'articule autour de deux axes: algorithmique (spécification système et analyse de performance) et architectural (partitionnement matériel / logiciel et analyse de l'efficacité de l'implémentation).
Face aux contraintes de la visiophonie embarquée, nous proposons un nouveau schéma de compression vidéo exploitant l'information sémantique de l'image vidéo, dans l'esprit de la norme MPEG-4 (codage objet). Nous proposons ainsi, à la fois un algorithme adaptatif pour l'extraction du visage, et un nouveau schéma de codage pour la compression vidéo à très bas débit.

Algorithm and architecture for embedded multimedia system

Abstract

This thesis deals with visiophony service implementation feasibility for next generation mobile terminal through an embedded multimedia systems conception methodology.
We first set up our research context : 3G networks, mobile terminals, processors for video coding, and also ISO/MPEG and IUT-T coding standards. We then propose a system level conception methodology taking into account embedded constraints like terminal autonomy. We apply it to H.263 video coder. This methodology is built along two levels : algorithmic (system specification and performance analysis) and architectural (hardware / software partitioning and implementation efficiency analysis).
To cope with embedded visiophony constraints, we proposed a new video compression scheme taking advantage of video semantic information, as suggested by MPEG-4 object coding approach. In this way, we both propose an adaptive algorithm for face detection and a new coding scheme for very low bit rate video compression.

Spécialité : Microélectronique

Mots clés : Visiophonie mobile, détection du visage, méthodologie de conception pour les systèmes embarqués.

ISBN : 2-913329-83-7
ISBNE : 2-913329-84-5