CODES DE CORRECTION D'ERREURS POUR LA TRANSMISSION DE LA TÉLÉVISION NUMÉRIQUE[†] – PHASE 3

RAPPORT FINAL

par

Jean-Yves Chouinard et David Belcourt
Département de génie électrique et de génie informatique
Université d'Ottawa

pour

Industrie Canada
Contrat UB450-6-0052
Approvisionnement et services
Gouvernement du Canada

Mars 1997

[†] projet de recherche est financé en partie par le Programme des centres d'excellence de langue
e d'Industrie Canada.

CODES DE CORRECTION D'ERREURS POUR LA TRANSMISSION DE LA TÉLÉVISION NUMÉRIQUE† – PHASE 3

RAPPORT FINAL

par

Jean-Yves Chouinard et David Belcourt
Département de génie électrique et de génie informatique
Université d'Ottawa

pour

Industrie Canada
Contrat UB450-6-0052
Approvisionnement et services
Gouvernement du Canada

Mars 1997

†Ce projet de recherche est financé en partie par le Programme des centres d'excellence de langue française d'Industrie Canada.

Table des matières

Liste des figures

Chapitre 1

Introduction

1.1 Introduction à la phase 3 du projet

Plusieurs études ont été effectuées sur l'ajoût d'une *porteuse numérique* (ou signal numérique) au signal de télédiffusion analogique existant. Ces études remontent aussi loin qu'en 1976 alors que la BBC (British Broadcasting Corporation) tentait de développer un système audionumérique compatible avec le standard de télévision européen PAL ("Phase Alternation Line"). Pour des raisons techniques qui seront exposées ultérieurement, les normes de transmission européennes ne peuvent s'appliquer telles quelles au système nord-américain. Des modifications à ces techniques sont donc rendues nécessaires pour que leur implantation soit réalisable. Ceci nous amène à la troisième phase du présent projet qui consiste à *relocaliser* le signal numérique et à qualifier les répercussions et quantifier leurs effets sur le signal analogique NTSC.

1.2 Objectifs de la phase 3

Les objectifs qui sont posés pour le présent projet consistent en le développement d'une plate-forme de simulation pour l'étude des effets de l'insertion d'une porteuse numérique dans le spectre du signal NTSC. La superposition d'une telle porteuse offrirait la possibilité de transmettre un nouveau signal supportant des données numériques, tout en permettant la diffusion du signal NTSC déjà utilisé par les systèmes de télédiffusion nord-américains. La nature et les applications de ce signal numérique n'étant pas encore définies pour le moment, la seule contrainte qui saurait régir l'étude du système de superposition de signaux proposé est la conservation d'un signal compatible avec le standard NTSC déjà en place, c'est-à-dire en minimisant les effets perturbateurs des interférences mutuelles entre les deux signaux superposés.

À la conclusion de l'étude, nous aimerions connaître avec plus d'exactitude les caractéristiques de la *porteuse numérique* qui devrait être insérée. Ceci inclut la définition qualitative et quantitative de sa localisation dans le spectre NTSC, du type de modulation, de sa puissance de transmission, de la largeur de bande qu'elle occupe dans le spectre, de sa capacité de transmission, du rapport signal sur bruit et de la probabilité d'erreur de transmission, sans négliger les effets perturbateurs sur le signal NTSC.

1.3 Plan du rapport

L'essentiel du travail de recherche pour la phase 3 de ce projet, à savoir l'étude de l'insertion d'un signal numérique dans le signal NTSC, est présenté au chapitre 2. À la section 2.2, on présente la méthode d'insertion du signal numérique dans le signal NTSC. Il y est notamment question de la localisation de cette *porteuse numérique* dans la bande résiduelle du signal NTSC. Ensuite, à la section 2.3, on s'intéresse à la mise au point de la plate-forme de simulation elle-même du système de transmission de données dans le signal NTSC. Après quelques remarques relatives au langage de programmation, on décrit les *composantes du programme* de simulation lui-même.

Le chapitre 3 porte sur des techniques de codage innovatrices pour la transmission de la télévision numérique avec *multiple résolution* (multirésolution) et qui permettent une dégradation *progressive* du signal vidéo. On rappelle à la section 3.1 les résultats récents et marqués obtenus en théorie du codage avec l'invention des codes dits *Turbo* et des méthodes de décodage itératif. Ensuite à la section 3.2, on s'intéresse à la protection *hiérarchique* des signaux numériques. Enfin, on présente à la section 3.3 les avantages que présentent la méthode de modulation codée multiniveau dans le contexte d'une transmission numérique avec dégradation progressive.

Chapitre 2

Transmission de données dans la bande d'un signal NTSC

2.1 Introduction

En 1987, l'ACATS ("Advisory Committee on Advanced Television Service") se vit assigner le mandat de la recommandation de normes (ou "standards") pour la télévision avancée (ATV: "Advanced Television"). Subséquemment, en 1992, six systèmes ATV furent mis à l'épreuve. De ces six systèmes ATV, quatre utilisent la transmission numérique. Il est évident que l'intégration des systèmes numériques dans les systèmes de télédiffusion fait l'objet de recherches intensives dans les différents centres de recherche du monde entier.

Plusieurs types de systèmes de télédiffusion radioélectrique sont présentement à l'étude ou en utilisation. Ces systèmes reposent sur une variété de techniques de transmission, allant des systèmes entièrement analogiques comme le standard NTSC ("National Television Systems Committee") jusqu'aux systèmes purement numériques de la télévision haute définition TVHD (HDTV: "High Definition Television").

Même si le système HDTV semble occuper une grande part des recherches dans le domaine de la télévision avancée, il demeure que ce système possède plusieurs inconvénients lorsqu'il s'agit de l'intégrer avec le système déjà existant de télédiffusion. Puisqu'il est impensable d'instaurer un système qui rendrait des millions de récepteurs de télévision inutilisables et désuets, les chercheurs se tournent vers d'autres alternatives qui permettraient la coexistence du système standard NTSC et une nouvelle famille de systèmes où l'utilisation des données numériques ouvrirait de nouveaux horizons. Le présent chapitre porte justement sur l'étude d'un système analogique-numérique permettant cette coexistence entre les deux systèmes.

2.2 Système de transmission de données dans la bande résiduelle du signal NTSC

2.2.1 Système de transmission de données proposé

Le système retenu dans le cadre de cette étude sur un nouveau système ATV, consiste en un système analogique NTSC conventionnel auquel une porteuse est insérée pour permettre la transmission d'information supplémentaire sous forme numérique. Ce système aurait comme avantage de préserver l'intégrité du signal NTSC. Aucune modification des récepteurs existants pour la réception d'émission télévisées tel qu'on le connaît aujourd'hui ne serait donc nécessaire. De plus, en utilisant une technique différente que celle présentement utilisée, il serait possible de récupérer l'information numérique de la porteuse. Cette information pourrait, à son tour, être utilisée pour une application entièrement numérique. Cette approche permet d'envisager de nouvelles possibilités, sans avoir à recourir à des modifications des systèmes de réception déjà en place. Il est clair que la réalisation et la mise en oeuvre d'un système analogique-numérique, tel que le projet le propose, serait plus probable à court et moyen terme qu'un système entièrement numérique comme la télévision haute définition. La figure 2.1 illustre de manière schématique le spectre du signal NTSC conventionnel ainsi que les différents signaux ou composantes.

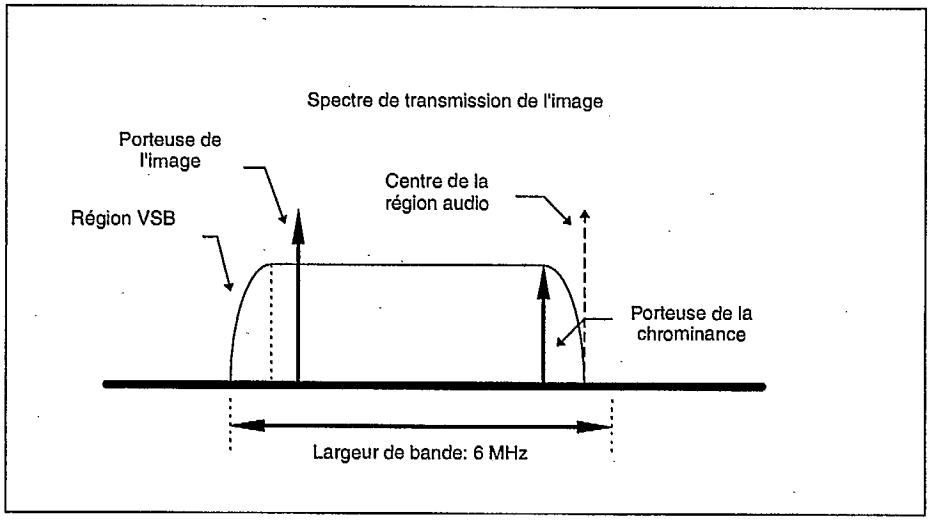

Figure 2.1: Représentation du spectre d'un signal NTSC.

2.2.2 Localisation du signal numérique à l'intérieur du signal NTSC

Des études faites par le Centre de recherche David Sarnoff pour un système de transmission de données compatible avec la norme NTSC démontrent qu'il est possible d'introduire une porteuse numérique dans la partie inférieure de bande résiduelle VSB. En procédant de cette façon, les effets d'interférence de la porteuse numérique sur le signal vidéo sont amoindris, la raison étant que la puissance transmise du signal numérique est fortement atténuée par le filtrage VSB. La figure 2.2 indique la localisation de la porteuse à l'intérieur du spectre NTSC. Ce système est connu sous le nom de *bande D* (i.e., "*D-channel*"). Des contraintes au niveau de la puissance de la porteuse numérique ainsi que du type de filtre requis doivent être prises en considération pour que le système soit fonctionnel. L'avantage résultant de cette nouvelle combinaison de signaux réside dans la possibilité de transmettre un signal audio entièrement numérique avec une capacité de 500 kbits/s à l'intérieur de la bande réservée au signal NTSC.

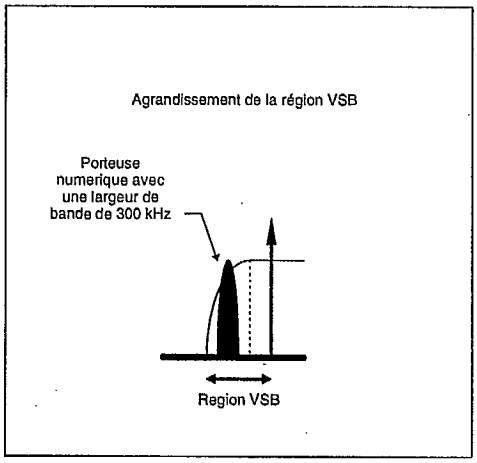

Figure 2.2: Localisation de la porteuse de données numérique dans le spectre du signal NTSC.

Pour cette troisième phase du projet de recherche, le signal numérique qui devra être insérée dans le canal NTSC consiste en un signal numérique modulé en phase MDP4, ou QPSK (modulation par déplacement de phase à quatre états ou "Quadriphase-shift keying"), sur une porteuse de bas niveau de puissance. Cette technique de superposition des signaux est déjà utilisée dans un système semblable qui repose sur le standard européen PAL.

Dans le système européen, l'insertion d'une porteuse numérique avec un débit de 728 kbits/s dans la bande FM audio permet de transmettre un signal audio numérique en stéréo. Mieux connue sous le nom de *NICAM-728* ("Near Instantenous Companded Audio Multiplex"), cette technique, quoique très utilisée en Scandinavie, en Europe continentale et au Royaume-Uni, n'est pas compatible avec le standard nord-américain NTSC, les raisons étant que les caractéristiques des filtres d'insertion et de recouvrement d'un signal numérique provoqueraient des interférences considérables sur les canaux adjacents. Ces filtres doivent avoir une largeur de bande qui excède de 100% la largeur du canal numérique. De plus, la technique européenne requiert l'insertion d'une porteuse de puissance moyenne dans le canal de télédiffusion radioélectrique, ce qui contribuerait à la dégradation des signaux analogiques NTSC. Or, l'objectif de cette étude est justement d'éviter que le signal NTSC original ne soit détérioré au point de ne plus être utilisable.

2.2.3 Plan du système de télécommunication

Le système développé par le centre de recherche David Sarnoff démontre que l'objectif du présent projet est possible. En comparant les systèmes *NICAM* et *D-channel* [Dig94], nous pouvons conclure avec certitude que les effets de cette porteuse numérique dépendent de l'endroit où elle se situe dans le spectre.

Puisque la porteuse numérique peut être contenue dans un canal à bande étroite, soit de l'ordre de 300 kHz, il est possible d'envisager plusieurs endroits où cette porteuse peut-être localisée à l'intérieur de la bande NTSC. Le système *D-channel* propose que cette porteuse soit placée dans la bande inférieure VSB du signal vidéo. Cependant, il semble tout aussi plausible qu'elle se puisse se situer entre la bande audio et la bande couleur (c'est-à-dire de chrominance) avec des performances similaires. L'une des facettes de cette étude vise à déterminer le (ou les) endroit(s) dans le spectre du signal NTSC où le canal numérique causera le moins d'interférences.

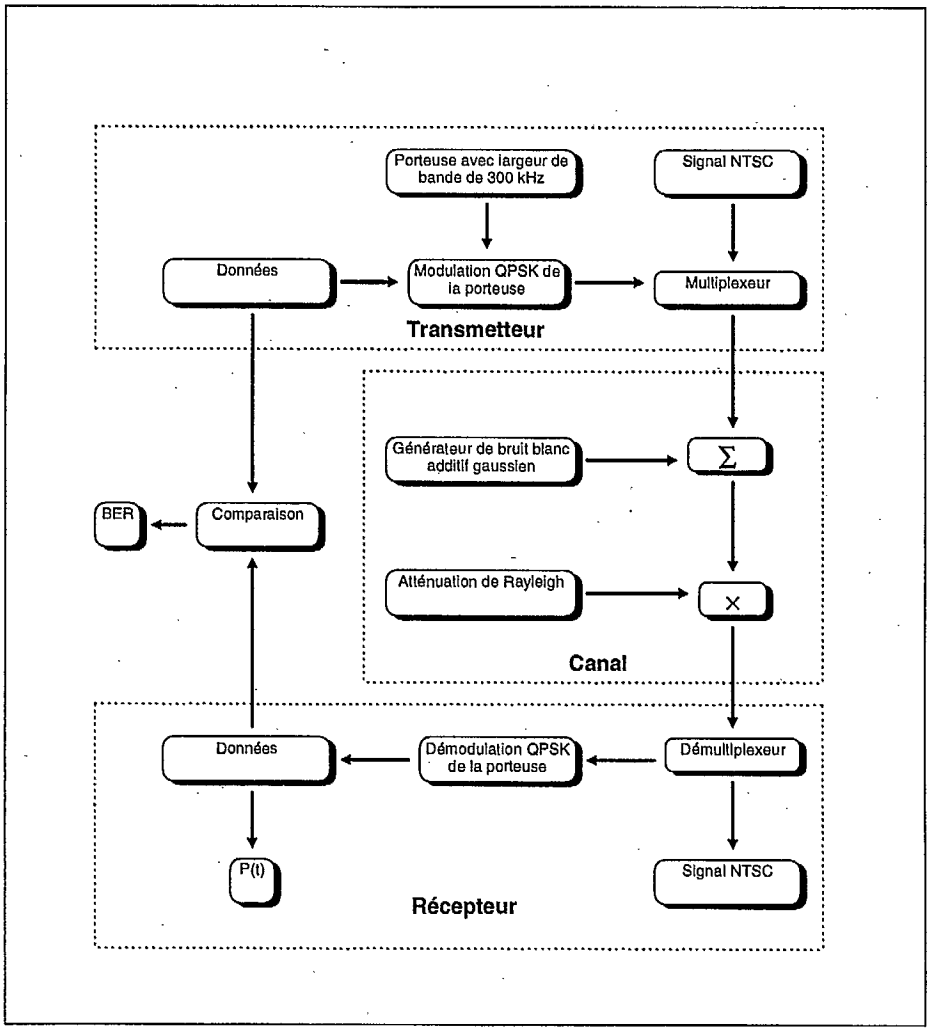

Figure 2.3: Diagramme bloc du système de télécommunication NTSC avec porteuse numérique.

La figure 2.3 montre sous la forme d'un diagramme bloc le système que l'on entend utiliser. Sur cette figure, on remarque que le canal est caractérisé par un bruit blanc

additif à bande limitée ainsi que de l'affaiblissement de signal de type Rayleigh. Comme le montre la figure 2.4, l'ajoût de canaux adjacents permettrait d'évaluer l'interférence co-canal causée par d'autres utilisateurs NTSC avec ou sans porteuse numérique. Ces canaux interférants sont donc du même type que le canal principal.

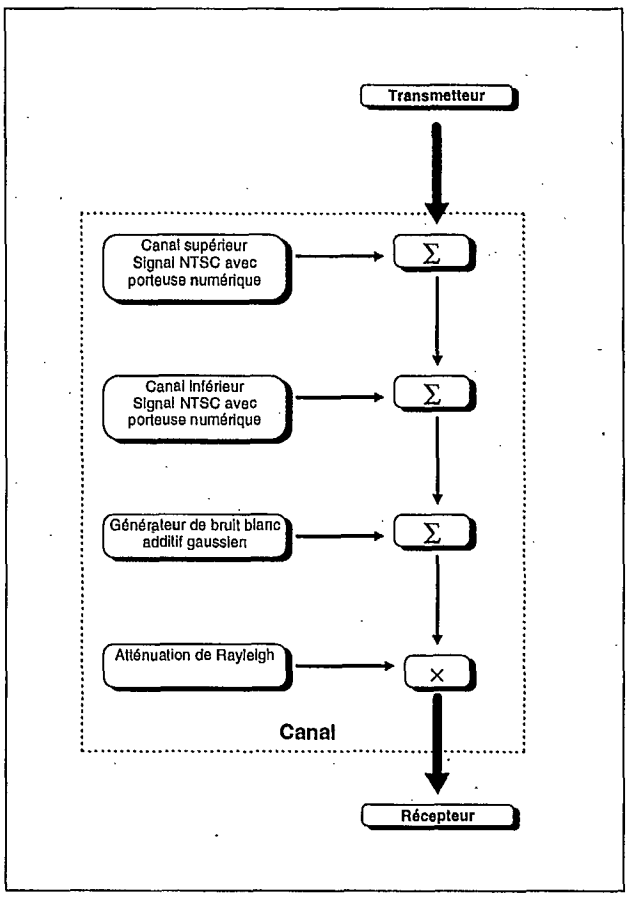

Figure 2.4: Modèle du canal avec bruit blanc additif, porteuses numériques et interférences NTSC.

On remarque qu'à la sortie du système, chaque bit est comparé avec son bit de départ. Ainsi, nous pouvons estimer la probabilité d'erreur par bit P_b (ou BER: "Bit Error Rate"). On mesure la puissance de chacune des bits et on détermine l'énergie par bit E_b, et ensuite, connaissant la valeur de la densité spectrale de la puissance du bruit N_0, le rapport signal sur bruit $\frac{E_b}{N_0}$ (SNR: "Signal to Noise Ratio").

D'autres types de modulation peuvent être utilisés pour la transmission de l'information numérique. Des systèmes à porteuses multiples ou à niveaux multiples peuvent aussi faire l'objet de recherches.

2.3 Simulation du système NTSC avec porteuse numérique

2.3.1 Langage de programmation

Le langage choisi pour la programmation des simulations du système de télécommunication est le langage *"C"*. Ce langage évolué nous permettra l'élaboration d'un *code plus efficace* (ici programme) que certains autres langages pour lesquels la programmation se fait via une interface graphique. En utilisant un programme efficace, nous réduisons considérablement le temps de calcul demandé par les simulations. Cependant, d'autres outils de programmation, par exemple les logiciels *MatLab* ou *Simulink*, sont utilisés pour la représentation de graphiques ou l'élaboration de sous-systèmes moins complexes du point de vue du temps de calcul requis. De plus, le langage de programmation *"C"*, par sa flexibilité et sa popularité, est plus facile et rapide à modifier pour accommoder de nouvelles contraintes de systèmes.

2.3.2 Étapes de conception

La *plate-forme de simulation* qui est prévue pour simuler le système de télécommunication pour ce projet nous permettra d'obtenir la probabilité d'erreur de transmission en fonction du rapport signal sur bruit de l'information numérique. Avec ces résultats, il nous sera possible de quantifier et d'étudier les performances de transmission. En modifiant les caractéristiques des filtres numériques, la puissance émise dans le canal numérique ainsi que les caractéristiques de modulation QPSK (MDP4), nous devrions être en mesure de *pouvoir contrôler le rendement* du canal numérique. Il est inévitable que l'augmentation du débit d'information et la diminution de la probabilité d'erreur se fera au détriment d'une dégradation du signal NTSC. Or, un des aspects de cette étude est de déterminer jusqu'à quel point le signal NTSC peut être distorsionné tout en maintenant un seuil minimal de fiabilité de la liaison.

La première étape de programmation consiste à élaborer un code (programme) qui permettra la modulation et la démodulation de données en utilisant une porteuse et la modulation QPSK. Une fois ce code de programmation au point, il s'agira d'incorporer cette porteuse dans le spectre NTSC. Nous utiliserons à cette fin une représentation en bande de base pour les premières étapes de la programmation. Si cette manière de procéder s'avère efficace et productive, il peut s'avérer possible de pouvoir modifier le programme pour que la simulation soit effectuée aux fréquences d'opération réelles. À l'aide d'un signal NTSC échantillonné et numérique, nous pourrons simuler le canal analogique dans lequel la porteuse numérique doit être insérée. L'étape la plus délicate à ce point est la modélisation des filtres qui sont utilisés pour insérer et récupérer la porteuse. Ces filtres sont de type *passe bande étroite*. Typiquement, ce genre de filtre possède une bande passante étroite et centrée autour d'une fréquence particulière. En ce qui concerne la modélisation de ces filtres, elle doit être faite en considérant les limites de leur réalisation pratique.

2.3.3 Calcul des probabilités d'erreur et du rapport signal sur bruit

La figure 2.3 illustre comment se fait le calcul de la probabilité d'erreur de transmission: nous nous attarderons au calcul de la probabilité d'erreur par symbole P_s et par bit P_b (BER). Nous souhaitons déterminer le nombre de symboles (ou alternativement de bits) en erreur sur le nombre total de symboles (ou bits selon le cas) de données transmis. Selon l'ordre de modulation (déterminé par le nombre de signaux dans la constellation), le nombre de symboles possibles ne sera pas le même. Les taux d'erreurs par symbole et par bit sont obtenus des expressions suivantes:

$$\begin{aligned} P_s &= \frac{N_{s_e}}{N_s} \\ P_b &= \frac{N_{b_e}}{N_b} \end{aligned}$$

où P_s et P_b représentent respectivement les probabilités d'erreur par symbole et par bit, N_{s_e} et N_{b_e} le nombre de symboles et de bits en erreur, et enfin N_s et N_b le nombre total de symboles et de bits transmis.

Un autre critère d'importance est le calcul du rapport signal sur bruit. À la figure 2.3, on remarque que la puissance de chaque bit est mesurée: on en détermine l'énergie correspondant à chaque bit, soit E_b. Sachant la densité spectrale de la puissance du bruit N_0 et sa largeur de bande B on calcule le rapport signal sur bruit $SNR = \frac{S}{N}$:

$$\frac{S}{N} = \frac{E_b}{N_0}\left[\frac{R_b}{B}\right]$$

où la variable R_b indique le taux de transmission des bits, ou débit binaire. Le même calcul peut être effectué au niveau des symboles en considérant cette fois-ci l'énergie par symbole E_s et le débit R_s exprimé en symboles par unité de temps.

2.3.4 Description détaillée de la programmation

En raison de contraintes d'espace mémoire, et afin de faciliter la programmation des simulations, nous séparons les différentes étapes en plusieurs sous-programmes, le principe étant de pouvoir analyser la performance de chaque bloc et ainsi permettre les modifications requises. Or, pour chaque bloc, l'information résultante est inscrite dans un fichier. Ainsi, au bloc suivant, nous avons tout simplement à récupérer cette information, la traiter et la transposer dans un nouveau fichier, et ainsi de suite. En procédant de la sorte, nous évitons les problèmes de synchronisation des étapes. Les blocs étant indépendants les uns par rapport aux autres, il est possible d'isoler, ou de *contourner* un des blocs, pour vérifier le bon fonctionnement d'un bloc en particulier.

La première étape consiste à générer une série de symboles de façon aléatoire. Pour la modulation QPSK, nous avons quatre symboles différents (soit $\boxed{00}$, $\boxed{01}$, $\boxed{11}$ et $\boxed{10}$). Pour ce faire, nous utilisons la fonction de génération de nombres aléatoire **rand** dans une boucle répétitive. Cette fonction permet de générer une séquence de nombres pseudo aléatoires suivant une loi uniforme. En effectuant une répartition égale pour les quatre cas, on obtient des symboles ayant une répartition de la probabilité égale entre les quatre symboles possibles, c'est-à-dire équiprobable.

L'une des contraintes à prendre en considération est la fréquence d'échantillonnage utilisée pour effectuer les simulations. Or, le signal NTSC utilisé pour les simulations se présente sous la forme d'un fichier ASCII. L'information contenue dans ce fichier est l'amplitude du signal échantillonné avec une fréquence de 14.32 MHz ($14.32 = 4 \times 3.58$ MHz). Il s'agit par la suite de synchroniser l'insertion de l'échantillon de la porteuse avec l'échantillon du signal NTSC. Tout au long de la programmation, une attention particulière est portée à la fréquence d'échantillonnage.

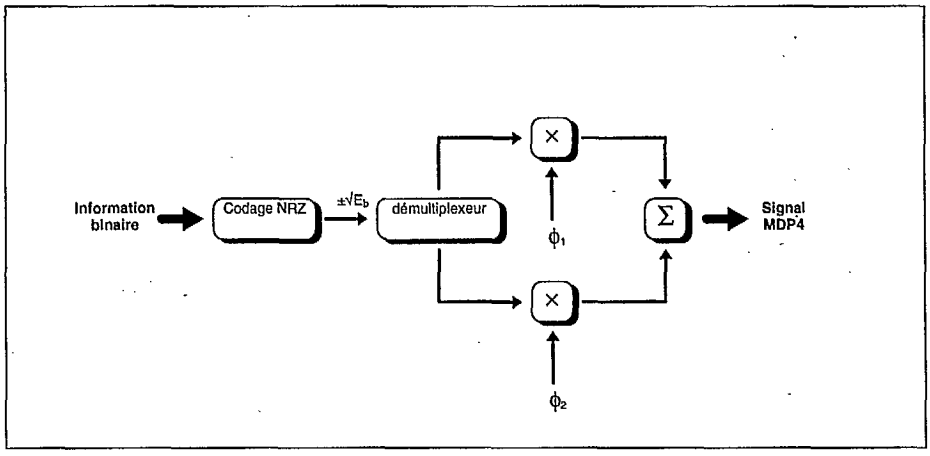

Figure 2.5: Modulateur par déplacement de phase à quatre états MDP4 (QPSK).

La figure 2.5 montre, sous la forme d'un bloc diagramme, les étapes de la modulation QPSK au transmetteur. Sur ce diagramme, on peut voir que la porteuse modulée est elle-même composée de deux composantes, soit une composante en phase (soit $\phi_1(t) = A\cos(\omega_c t)$) et l'autre en quadrature ($\phi_2(t) = A\sin(\omega_c t)$). Une des variables d'importance est l'énergie qui résulte de l'amplitude de chaque bit avant le codage de ligne *de non retour à zéro* NRZ (i.e. a_1 et a_2). Lors de la programmation du bloc de modulation QPSK, nous faisons en sorte qu'il soit possible de modifier ces variables avec aisance. Comme le montre

la figure 2.5, il s'agit alors de prendre chaque symbole sous forme binaire et de leur assigner des niveaux logiques en utilisant le codage de non-retour à zéro. En traitant les bits en parallèle (2 bits $\Longleftrightarrow$ 4 symboles), nous les mélangeons avec les deux composantes de la porteuse soit: $\phi_1(t)$ et $\phi_2(t)$.

Puisqu'il s'agit d'une simulation numérique, la variable usuelle t du domaine de temps continu est remplacée par son équivalent discret k. Enfin, on effectue une sommation des deux composantes dans le plan complexe des signaux. Au niveau de la programmation de ce bloc, les opérations mathématiques et logiques s'effectuent tel qu'illustré par le diagramme bloc. L'opération de mélange des signaux consiste donc en deux multiplications et une sommation dans le plan complexe alors que le codage de ligne NRZ assigne les valeurs $\pm\sqrt{E_b}$ selon que la valeur logique de l'information binaire est un ou zéro. Au niveau de la durée de chaque échantillon, nous devons considérer la fréquence d'échantillonnage du signal NTSC (soit 14.32 MHz), celle-ci étant notre point de référence.

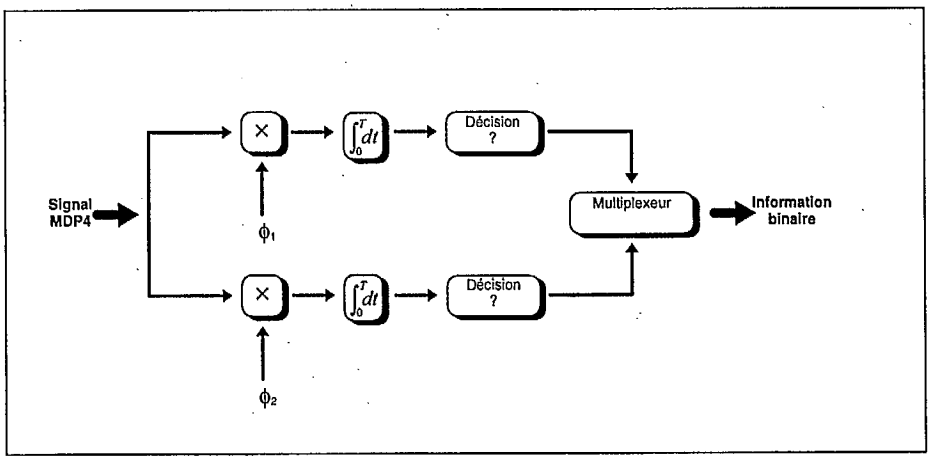

Figure 2.6: Démodulateur MDP4 (QPSK).

La démodulation de ce signal s'effectue dans un bloc séparé. Ceci permet d'inclure une étape pour modéliser un canal de télédiffusion. La figure 2.6 donne un aperçu d'un récepteur à démodulation cohérente. Comme dans le cas du transmetteur, les opérations mathématiques et logiques peuvent être reproduites sans difficulté. Puisque l'information à démoduler se retrouve dans un fichier (comme c'est le cas pour tous les blocs du système simulé), on peut extraire l'information à même le fichier source.

Pour faciliter et simplifier la tâche de la programmation, nous supposons que la synchronisation est établie dès le départ et maintenue par la suite. En d'autres mots, le premier symbole de la chaîne est supposé toujours connu. On obtient ainsi une référence de phase nulle.

La simulation s'effectue dans le domaine du temps discret et les opérations effectuées en temps continu, tel une intégrale temporelle, se traduisent par une sommation temporelle. Or, la démodulation QPSK cohérente demande que le signal soit intégré. Le but de cette opération est de déterminer dans quel hémisphère polaire (quadrant) du plan complexe se trouve le symbole à démoduler. Selon que le résultat de l'intégrale (la sommation dans notre cas) est positif ou négatif, nous sommes en mesure de déterminer si la valeur des composantes en phase et en quadrature sont du niveau logique un ou zéro afin de déterminer lequel des symboles, à savoir $\boxed{00}$, $\boxed{01}$, $\boxed{11}$ ou $\boxed{10}$, a été le plus vraisem-

blablement transmis par la source. Pour programmer cette intégrale, nous effectuons une sommation sur N échantillons (N étant le nombre d'échantillons par symboles).

Pour que le transmetteur soit complet, il doit comporter une étape de combinaison de la porteuse numérique avec un signal NTSC existant. Cela consiste à effectuer une addition vectorielle des deux signaux complexes. En théorie, si l'échantillonnage est suffisamment élevé, nous devrions nous rapprocher suffisamment du cas analogique. D'où, ici, l'importance que les deux signaux (NTSC et QPSK) possèdent la même fréquence d'échantillonnage. La fonction mathématique n'est pas complexe à instaurer dans le programme, mais le conditionnement du signal QPSK demande une grande attention. Il en est de même pour le récepteur, où une étape du procédé consiste à récupérer la porteuse insérée.

À cette étape du projet, il est difficile de définir les caractéristiques du filtre passe-bande ainsi que les opérations nécessaires pour le simuler. En théorie, pour une réponse en fréquence donnée par sa fonction de transfert $H(f)$, on détermine sa réponse impulsionnelle $h(t)$ par sa transformée inverse de Fourier. Nous savons que le signal à la sortie d'un tel système est donné par $y(t) = h(t) \otimes x(t)$, c'est-à-dire la convolution du signal d'entrée $x(t)$ avec la réponse impulsionnelle du canal $h(t)$.

Le modèle de canal doit également inclure l'addition de bruit blanc Gaussien. Pour générer ce bruit on utilise, comme dans le cas du générateur de symboles, la fonction **rand**. En modifiant la distribution des variables aléatoires on obtient une composante de bruit plus ou moins importante. Finalement, une fois toutes ces étapes exécutées, nous devons obtenir un fichier ou se retrouve l'information originale transmise mais corrompue par le canal de transmission. La dernière étape de la programmation consiste prendre l'information à la sortie de la chaîne de télécommunications et de la comparer avec l'information originale émise par la source telle qu'inscrite dans le fichier de départ. Ce dernier bloc effectue la lecture de l'information dans les deux fichiers (i.e., celle d'origine et celle démodulée), compare chaque symbole (ou les bits selon le cas) un à un et fait le décompte correspondant du nombre de symboles ou de bits en erreur.

2.3.5 Simulations

L'ordre dans lequel les diverses étapes du programme de simulation du système de transmission de données superposé au signal NTSC sont accomplies est le suivant:

1. génération des symboles pseudo aléatoires;
2. modulation des symboles en phase à l'aide de la modulation QPSK;
3. ajoût de la porteuse numérique dans le signal NTSC standard;
4. simulation du canal de télédiffusion avec atténuation de Rayleigh et addition de bruit Gaussien;
5. récupération de la porteuse (signal) numérique;
6. démodulation QPSK et récupération de l'information; et
7. comparaison de l'information originale transmise et celle reçue.

Description de la représentation graphique des résultats

Cette section est inclue pour faciliter la compréhension des graphiques des résultats. Contrairement à la représentation typique des spectres en bande de base, les figures inclues dans le rapport ont une allure différente. Deux raisons expliquent ces différences. La première est que les signaux traités sont des échantillons temporels. La conséquence sur le spectre est la répétition du spectre de base à des intervalles donnés pas la fréquence d'échantillonnage. En théorie, tant et aussi longtemps que la fréquence d'échantillonnage respecte le théorème de Nyquist, le fait de traiter un signal discret au lieu d'un signal continu ne pose aucune difficulté. La seconde raison qui explique les résultats graphiques est qu'ils originent du logiciel *MatLab*. L'exécution de la commande de transformation de Fourier (FFT) produit un résultat qui se répartit entre les fréquences 0 et f_s. Or, dans les graphiques, la partie positive du spectre se retrouve dans la première moitié (celle de gauche) et la partie négative du spectre se retrouve dans la seconde moitié (celle de droite). La figure 2.7 montre ces étapes.

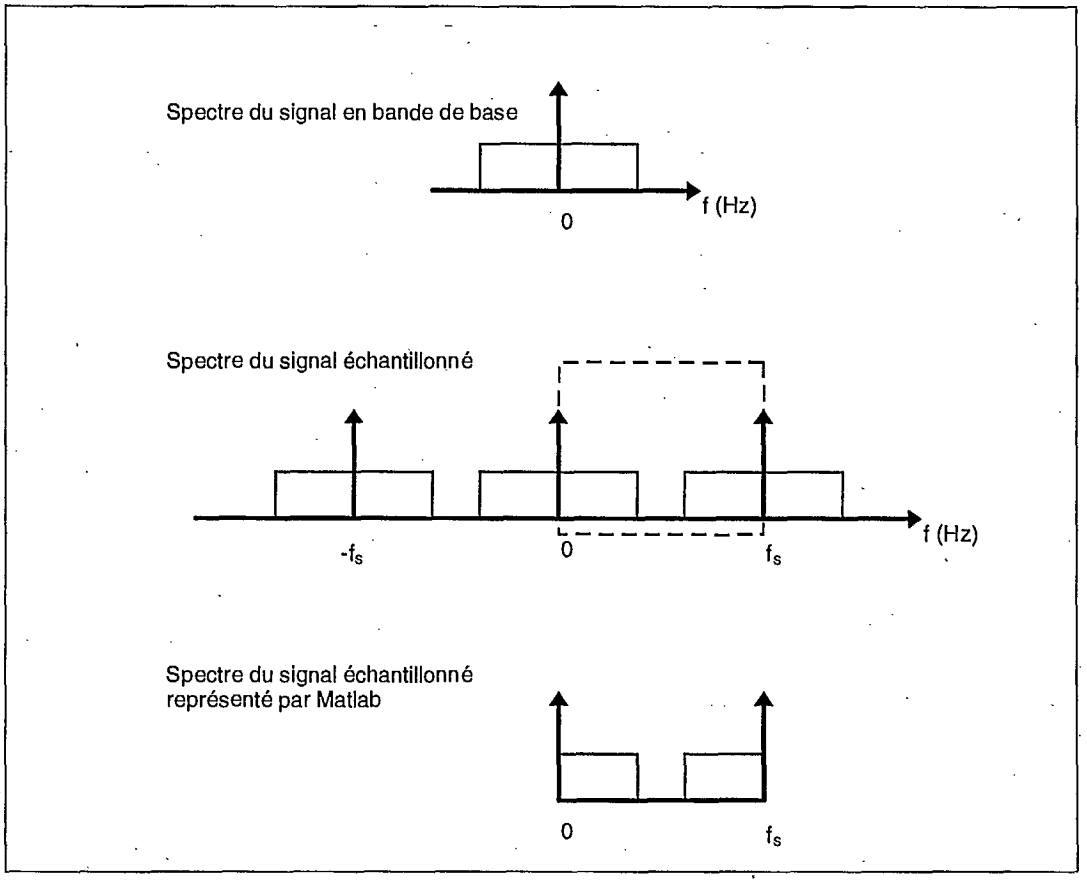

Figure 2.7: Spectre d'amplitude du signal vidéo NTSC à la sortie du filtre VSB.

La figure 2.8 donne un exemple de signal modulé avec la modulation QPSK représenté en fonction du temps t. Cette figure, comme les suivantes, ont été produites à l'aide du logiciel *MatLab*.

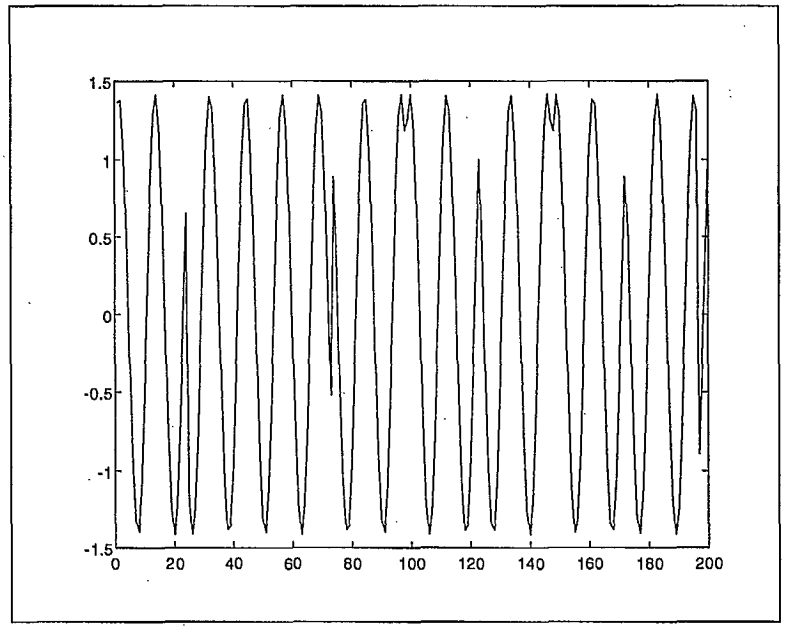

Figure 2.8: Exemple de signal modulé avec la modulation QPSK (en fonction du temps).

La figure 2.9 montre le spectre d'amplitude du signal vidéo NTSC à l'entrée du filtre VSB.

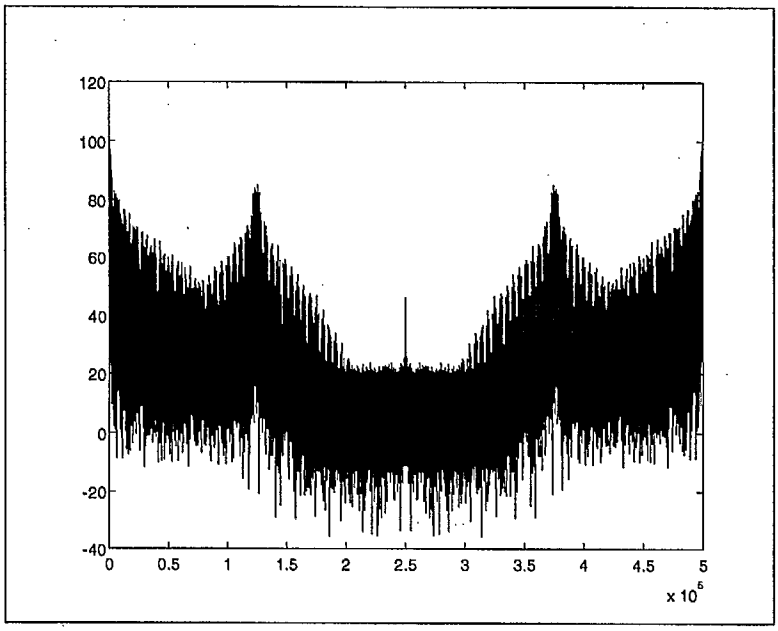

Figure 2.9: Spectre d'amplitude du signal vidéo NTSC à l'entrée du filtre VSB (échelle d'amplitude en dB).

Le spectre d'amplitude du signal de données numériques est montré à la figure 2.10.

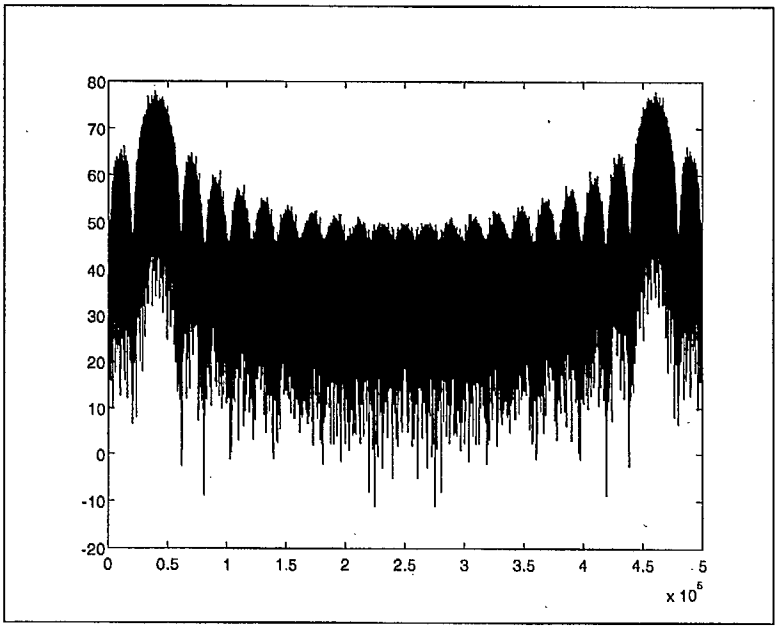

Figure 2.10: Spectre d'amplitude de la porteuse numérique à 1 MHz.

La fonction de transfert $H(f)$ du filtre VSB est montrée à la figure 2.11 sur une échelle linéaire d'amplitude et en décibels sur la figure 2.12.

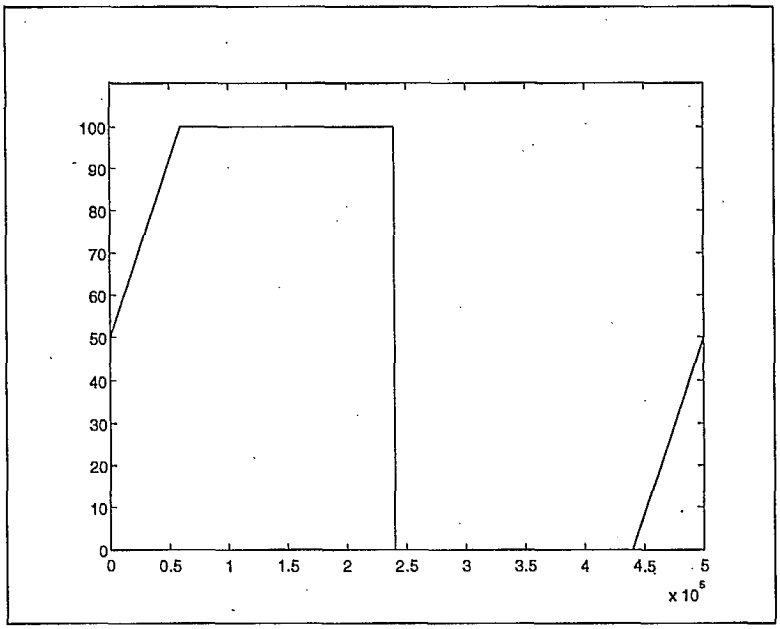

Figure 2.11: Fonction de transfert $H(f)$ du filtre VSB (amplitude linéaire).

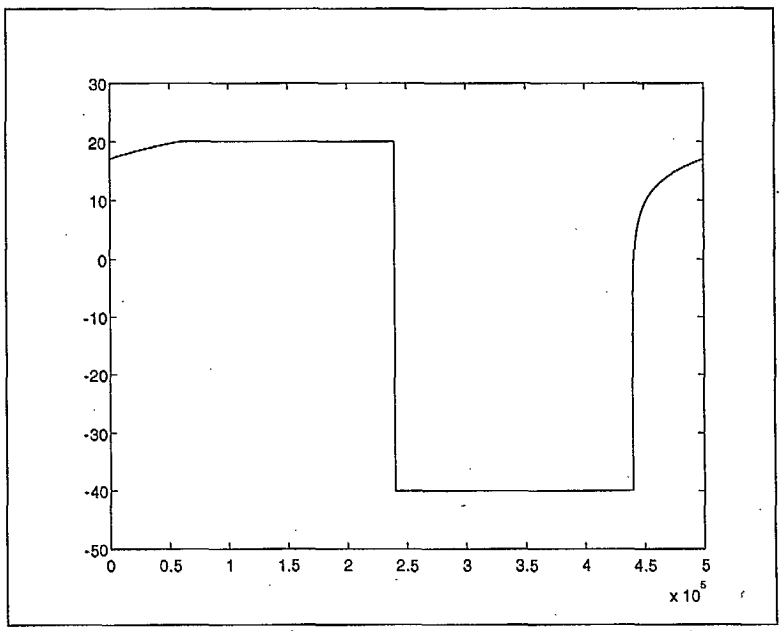

Figure 2.12: Fonction de transfert $H(f)$ du filtre VSB (amplitude en dB).

La figure 2.13 montre, sur une échelle linéaire, la fonction de transfert du filtre passe bande.

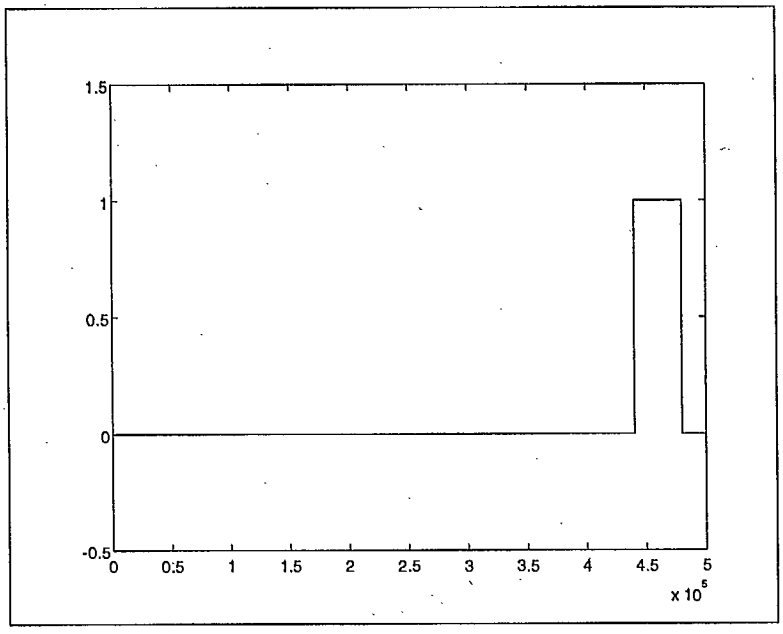

Figure 2.13: Fonction de transfert du filtre passe bande (échelle d'amplitude linéaire).

La figure 2.14 montre le spectre d'amplitude du signal vidéo NTSC à la sortie du filtre VSB: l'échelle d'amplitude est en décibels (dB).

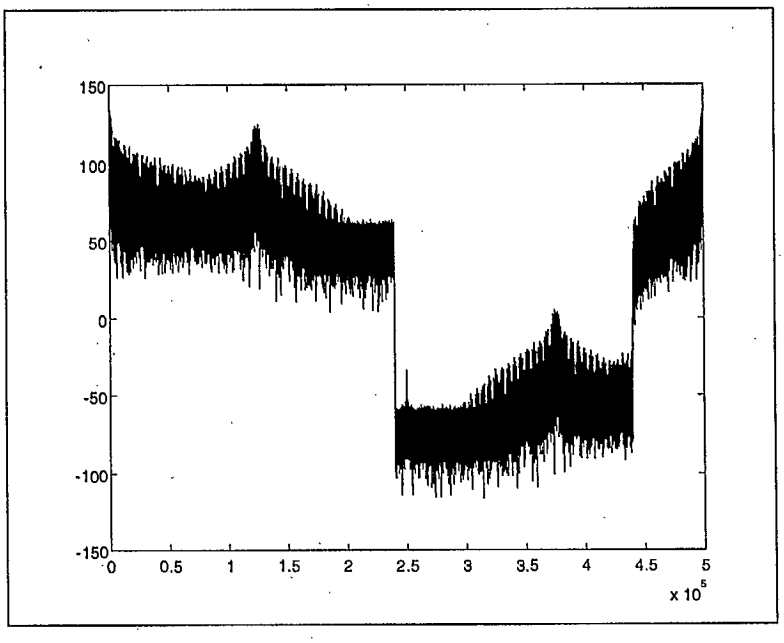

Figure 2.14: Spectre d'amplitude du signal vidéo NTSC à la sortie du filtre VSB (échelle d'amplitude en dB).

La figure 2.15 montre le spectre d'amplitude de la porteuse numérique, c'est-à-dire le signal de données numériques, à la sortie du filtre passe-bande. Enfin, la figure 2.16 montre le spectre d'amplitude du signal vidéo NTSC à la sortie du filtre VSB avec la porteuse de données numériques QPSK.

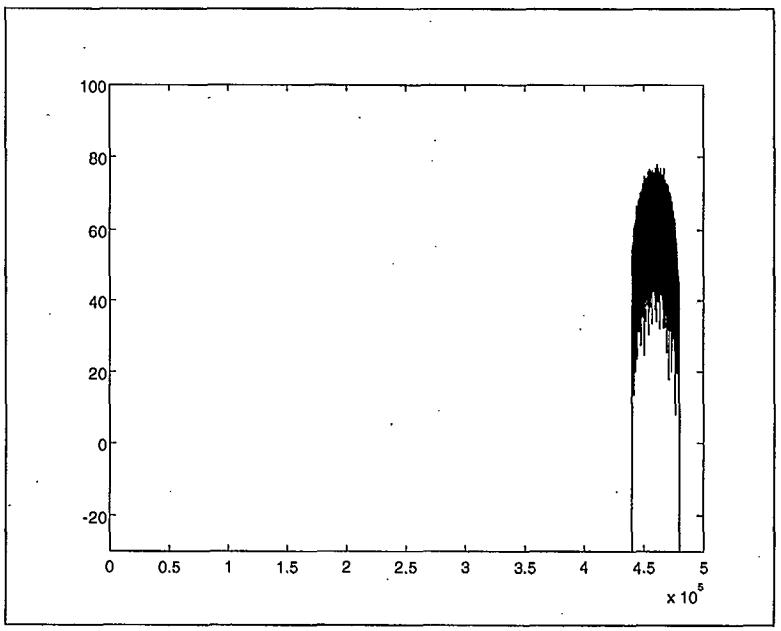

Figure 2.15: Spectre d'amplitude de la porteuse numérique (signal de données) à la sortie du filtre passe-bande (fréquence: 1 MHz, 500 kbauds/s; échelle d'amplitude en dB).

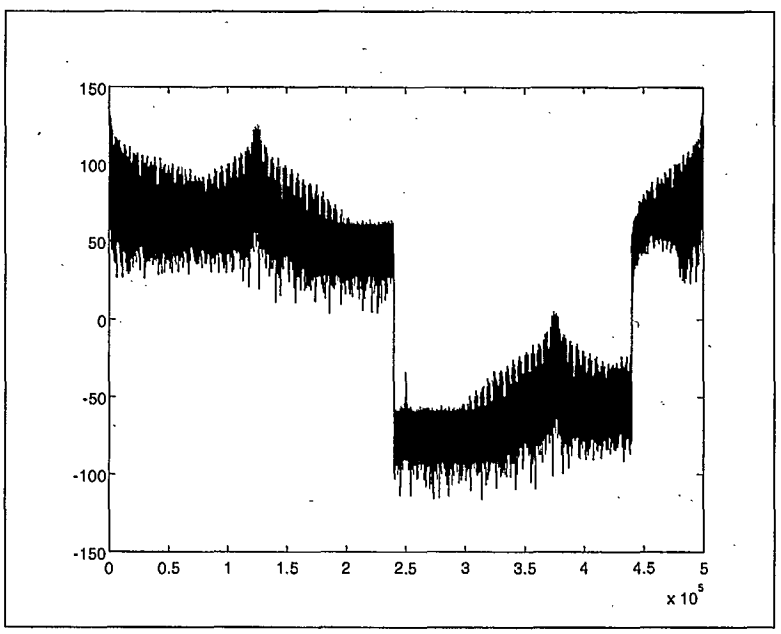

Figure 2.16: Spectre d'amplitude du signal vidéo NTSC à la sortie du filtre VSB avec la porteuse QPSK (fréquence: 1 MHz, 500 kbauds/s; échelle d'amplitude en dB).

Programmes en langage *"C"*

Comme le schéma de la figure 2.17 le montre, la simulation du système de communication est bâtie en utilisant une combinaison de programmation *"C"* et de certaines fonctions *MatLab*. Comme description générale de l'architecture du système, nous pouvons observer que l'information générée et modulée QPSK origine d'une série de deux étapes. Or, l'étape préliminaire consiste à générer de façon aléatoire des nombres de 0 à 3. Ces nombres représentent les quatre symboles (00, 01, 11 et 10) de la modulation QPSK. Le nombre de symboles générés, ainsi que l'énergie par bit, peut être défini par l'utilisateur. Pour accélérer le processus lors de la construction de la simulation, ces variables, normalement définies par l'utilisateur, sont remplacées par des constantes. Les résultats obtenus à l'aide de ces constantes sont propices à l'évaluation des étapes subséquentes. Dépendamment des résultats finaux de la simulation, il nous est toujours possible de modifier ces constantes de départ pour pouvoir évaluer leur impact sur les performances du système. Le résultat de cette première étape est conservé dans un fichier texte (i.e. avec une extension *.txt*). En procédant de cette façon nous pouvons éditer et accéder directement à l'information qui est générée par cette étape. L'avantage de pouvoir éditer l'information sous une forme texte sera discuté ultérieurement. Pour l'instant, nous savons qu'il existe un fichier qui contient l'information quaternaire qui est utilisée pour la modulation QPSK.

L'étape qui suit celle de la génération des symboles pseudo-aléatoires est celle de la modulation QPSK proprement dite. Pour concevoir ce sous-programme, trois paramètres fréquentiels ont du être pris en considération soient: la fréquence de la porteuse, le taux de transmission et la fréquence d'échantillonnage. Pour simplifier la combinaison de ces trois paramètres, ces trois fonctions sont traduites en fonctions temporelles. En effectuant ces opérations, nous pouvons appliquer les opérations de la modulation QPSK tel que s'il s'agissait d'un signal continu dans le domaine du temps. En d'autres mots, la seule différence entre la simulation numérique de ce modulateur et la modulation d'un signal analogique est que l'information résultante est échantillonnée à tous les $\frac{1}{f_s}$, où f_s est la fréquence d'échantillonnage utilisée tout au long de la simulation. Il est intéressant de noter que la fréquence de la porteuse ainsi que le taux de transmission ne sont pas des multiples de la fréquence d'échantillonnage. Donc, la référence du système doit être le temps et non pas le nombre d'échantillons. Pour ce qui est de la fréquence de la porteuse, il est proposé qu'elle soit fixée à 1 MHz en dessous de la fréquence de la porteuse vidéo du signal NTSC. Le débit de transmission prévu est de 500 kbauds/s.

L'information transmise par la porteuse numérique devra être ajoutée au canal de télédiffusion NTSC. Or, il semble important de mentionner à ce moment que le signal NTSC numérisé qui est utilisé pour cette simulation consiste en un ficher ASCII qui comporte 500,000 échantillons codés avec une précision de 8 bits. Ainsi, pour conserver ce standard, l'information qui résulte du modulateur QPSK est, elle aussi, codée en utilisant 8 bits ASCII et le nombre total d'échantillons est de 500,000. De plus, les échantillons du fichier NTSC ont été prélevés à une fréquence de 14.3 MHz. Cette constante est aussi prise en considération comme fréquence d'échantillonnage dans l'étape de modulation.

Pour résumer l'étape de génération de la porteuse numérique QPSK, nous utilisons le langage *"C"* pour générer des symboles de façon pseudo-aléatoire. Ensuite, nous utilisons ces symboles comme source d'information numérique pour la modulation QPSK. Deux fichiers ASCII résultent de ces opérations, soient un fichier contenant les symboles qui ont été modulés à un taux de transmission de 500 kbauds/s et un fichier contenant

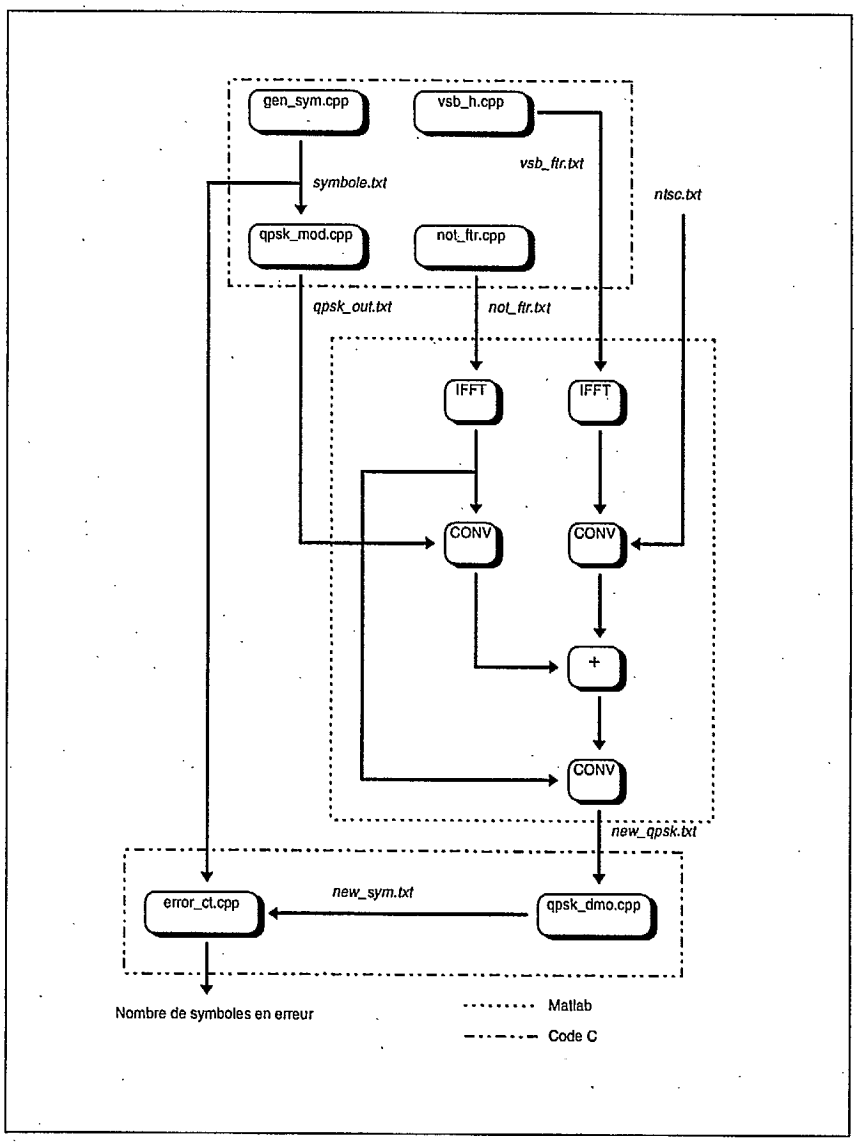

Figure 2.17: Schéma du système simulé (programmes en langage *"C"* et *MatLab*).

les échantillons du signal temporel de la porteuse QPSK modulée à une fréquence de 1 MHz. Ce dernier fichier possède les caractéristiques suivantes: 500,000 échantillons et une fréquence d'échantillonnage de 14.3 MHz.

Comme nous le verrons dans la prochaine section, le canal proprement dit est simulé avec les fonctions de *MatLab*. Toutefois, les signaux et les paramètres qui sont utilisés par *MatLab* proviennent de fichiers générés par des programmes écrits en code *"C"*, exception faite du signal NTSC. Le canal consiste en plusieurs filtres. Puisque la réponse de ces filtres nous est inconnue au départ, une réponse idéale a été utilisée en première approximation. Plusieurs fichiers on été créés à partir de courts programmes en *"C"*. Or, l'information que contient ces fichiers est la réponse fréquentielle de l'amplitude de chaque filtre. Le filtre VSB qui est utilisé pour simuler le canal est du type complexe. Typiquement, les filtres VSB utilisés pour les canaux télévisés sont de type réel. Le facteur qui impose une réponse complexe du filtre de la simulation est le fait que la simulation s'effectue en bande de base. Dans le cas d'un système analogique, le filtrage s'effectue à la fréquence

intermédiaire, soit d'environ 40 MHz (i.e. 41.75 MHz). Dans ce dernier cas, le filtre possède une image de son spectre aux fréquences négatives. Il est donc considéré réel et possède une phase constante dans la bande d'intérêt. À l'opposé, le filtre complexe utilisé pour la simulation possède une phase *erratique* dans sa bande et est sujet à corrompre de façon irrécupérable le signal que l'on introduit. Un autre filtre, dont la réponse en fréquence est générée par la programmation *"C"*, est un filtre passe-bande. Ce filtre est utilisé pour éliminer la majeure partie des composantes fréquentielles des lobes secondaires de la porteuse numérique. En fait, l'élément que l'on désire conserver est le lobe principal situé à la fréquence négative $-f_c$, où f_c est la fréquence de la porteuse. On retrouve ici les mêmes complications que dans le cas du filtre VSB simulé. Le filtre passe-bande n'ayant pas de symétrie avec l'axe $f = 0$ Hz, nous retrouvons une réponse complexe. Le même type de filtre est prévu pour extraire la porteuse du signal NTSC-QPSK.

Les deux derniers blocs programmés avec le langage *"C"* sont le démodulateur QPSK et le comparateur de symboles. Comme dans le cas du modulateur, les paramètres tels que la fréquence d'échantillonnage, le taux de transmission et la fréquence de la porteuse, sont nécessaires à la démodulation cohérente du signal QPSK. En plus des quatre états possibles pour la sortie du démodulateur, une condition d'erreur est ajoutée dans l'éventualité que le résultat de l'intégrale (en fait une sommation dans notre cas) d'une des deux composantes (i.e. en phase et en quadrature) soit nulle, ce qui est une erreur.

Le dernier bloc consiste en un compteur d'erreurs. Deux fichiers sont utilisés par ce programme, soient le fichier qui contient les symboles de départ, avant la modulation et le fichier où se trouvent les symboles qui résultent de la démodulation. Par une soustraction des symboles de départ, avec les symboles qui sont récupérés du canal, nous obtenons le nombre de symboles en erreur. Présentement, le système compare 20,325 symboles, soit le total des symboles qui sont générés et transmis dans le canal.

Simulation du canal avec *MatLab*

Comme il a été mentionné précédemment, l'ensemble des paramètres et des signaux qui sont utilisés dans le canal proviennent d'une programmation en langage *"C"*. La simulation du canal utilise la flexibilité de *MatLab* pour effectuer des opérations telles que les transformées de Fourier (FFT) ainsi que les transformées inverses de Fourier (IFFT). Les opérations de convolutions et d'additions vectorielles s'effectuent directement sans recourir à aucune programmation. Étant donné que le canal se résume à une série de convolutions et à une addition, il est évident que ce logiciel soit de choix pour cette étape de la simulation. Les fichiers requis pour la simulation du canal sont les suivants:

1. fichier de 500,000 échantillons temporels d'un signal NTSC (fichier *ntsc.txt*),
2. fichier de 500,000 échantillons temporels de la porteuse 1 MHz modulée QPSK (fichier *qpsk_out.txt*),
3. fichier de 20,000 échantillons fréquentiels du spectre du filtre VSB (fichier *vsb_h.txt*),
4. fichier de 143 échantillons fréquentiels du spectre du filtre passe-bande (fichier *not_h.txt*).

La première étape consiste à conditionner les deux signaux d'entrée. Pour ce faire, nous utilisons les propriétés de la convolution temporelle ainsi que la possibilité d'obtenir la réponse impulsionnelle d'un spectre en effectuant sa transformée inverse de Fourier. La séquence des commandes de *MatLab* qui permettent ces opérations est la suivante:

```
load ntsc.txt
load vsb_h.out
CHANNEL = conv(ntsc, (ifft(vsb_h)))

load qpsk_out.txt
load not_h.txt
CARRIER = conv(qpsk_out, (ifft(not_h)))
```

À première vue, on pourrait croire que ces quelques lignes peuvent s'effectuer sans encombrement. Il est vrai que les opérations sont peu complexes, mais les problèmes résident dans la quantité d'information traitée. Les fichiers tels que *ntsc.txt* et *qpsk_out* occupent un espace mémoire qui excède les 5,000 koctets chacun. Les fichiers *not_h.txt* et *vsb_h.txt*, quoique beaucoup moins volumineux, contribuent à occuper la mémoire de l'ordinateur. Pour des raisons de gestion de la mémoire, ces opérations ne peuvent être exécutées consécutivement. L'utilisateur doit s'assurer de sauvegarder l'information de chaque étape et faire l'effacement de la mémoire *manuellement* avant de passer à l'étape suivante. Même si les simulations sont exécutées sur un ordinateur de type *Pentium 133 MHz* avec 32 Moctets de mémoire vive, le temps nécessaire pour effectuer les opérations précitées est d'environ 5 à 6 heures.

Une fois ces étapes terminées avec succès, nous effectuons une addition vectorielle des deux vecteurs. Ici, nous employons le terme vecteur puisque *MatLab* considère une série de nombres comme une matrice de dimension $[1, N]$ ou un vecteur de longueur N. Ainsi les quatre fichiers de départ sont traités comme s'il s'agissait de quatre vecteurs. La commande qui permet une telle addition est:

```
DCHANNEL = CARRIER + CHANNEL
```

Avant de poursuivre la description des étapes de la simulation, il est propice de rappeler que les opérations de transformée inverse de Fourier nous donnent des résultats complexes dont la phase est non-linéaire. Il va de soi que les résultats des convolutions sont eux aussi complexes.

Subséquemment à l'addition des deux signaux, une convolution avec la réponse impulsionelle d'un filtre passe-bande semblable à celui utilisé pour le conditionnement du signal QPSK permet de récupérer une majeure partie du spectre de la porteuse numérique. Finalement, la dernière étape avant la démodulation est la sauvegarde du vecteur résultant dans un fichier texte. En conservant l'information résultante dans le fichier texte, nous pouvons utiliser le démodulateur qui a été décrit dans la section précédente.

Commentaires et travaux futurs

La fiabilité et la performance du système dépend, en majeure partie, des contraintes qui sont imposées lors de la conception. Puisque que cette technique est expérimentale, la possibilité de la coexistence d'une porteuse numérique à l'intérieur des canaux de télédiffusion existants reste à démontrer. La performance du système dépend notamment de:

1. la puissance transmise via la porteuse numérique,
2. le débit de transmission désiré,
3. le type de filtre utilisé pour récupérer la porteuse,
4. le type de filtre utilisé pour former le canal VSB,
5. la localisation de cette porteuse dans la région VSB.

Sachant que la simulation dont fait l'objet cette étude doit se rapprocher le plus près possible du système analogique final, il est évident que des études sur les caractéristiques des filtres analogiques sont nécessaires. Dans le cas de cette simulation, les techniques numériques utilisées pour générer les filtres VSB et passe-bande permettent beaucoup plus de flexibilité que s'il s'agissait de véritables filtres matériels analogiques. Nous savons que pour que ce système soit possible, la réponse du filtre VSB doit être modifiée de telle sorte que la puissance du signal NTSC transmise dans le canal numérique soit minimisée. Puisque les deux signaux (porteuse numérique et signal NTSC analogique) doivent se chevaucher à l'intérieur du canal, il est apparent que chaque signal voit son voisin comme un signal interférant. Il est certain que le chevauchement devient moins important au fur et à mesure que nous éloignons la porteuse numérique de la porteuse vidéo du signal NTSC. Toutefois, ce faisant, on accroît le risque de causer de l'interférence dans les canaux adjacents. Cette interférence est plus ou moins importante selon l'importance de la puissance de la porteuse numérique.

Un autre facteur important à considérer est le taux, ou débit, de transmission. Dans le cas de la modulation QPSK, il existe une relation entre le taux de transmission et la largeur de bande efficace. Cette relation est donnée par:

$$B = \frac{2}{T}$$

où T est la durée de transmission, ou période, d'un symbole et B la largeur de bande effective.

D'après cette relation, plus le taux de transmission est élevé plus la largeur de bande du lobe principal (effective) sera grande. Dans le cas qui nous intéresse, un débit de 500 kbauds/s nécessite une largeur de bande de 1 MHz. Cette largeur de bande prend autant plus d'importance lorsqu'il s'agit de récupérer la porteuse à l'aide d'un filtre passe-bande. Encore ici, la précision du filtre joue un rôle crucial dans la performance du système. Si la largeur de bande est trop étroite, on accuse une perte de puissance dans la transmission de la porteuse. En contre partie, si la largeur de bande du filtre est trop grande, une plus grande partie du signal interférant (ici le signal NTSC) demeure dans le canal, ce qui contribue à la corruption de l'information.

Chapitre 3

Techniques de codage pour la transmission de télévision numérique avec multirésolution

3.1 Codes Turbo et décodage itératif

Les développements récents en théorie des codes de correction d'erreurs démontrent que les performances peuvent approcher de très près la limite théorique de Shannon, à savoir la capacité du canal. Ainsi à la conférence ICC'93 (1993 International Conference on Communications), trois chercheurs français de l'École Nationale Supérieure des Télécommunications de Bretagne (ENST-Bretagne), Berrou, Glavieux et Thitimajshima [BGT93], ont présentés une nouvelle classe de codes correcteurs, les *codes Turbo*, qui leur ont permis d'obtenir après quelques (18) itérations des taux d'erreur par bit $BER \leq 10^{-5}$ avec un rapport signal sur bruit $\frac{E_b}{N_0}$, exprimé en énergie par bit sur la densité spectrale de puissance, ne dépassant la capacité *théorique* du canal (canal à bruit blanc gaussien additif (AWGN)) que de 0.7 dB seulement.

Ce qui est particulièrement intéressant, c'est que ces performances ont été obtenues à l'aide d'un codeur *Turbo* constitué tout simplement de deux codeurs convolutifs récursifs systématiques (37,21) de longueur de contrainte (ou mémoire) $L = 4$. Il s'agissait en fait de deux codeurs utilisant le même code (37,21) avec un entrelaceur placé à l'entrée du second codeur.

À la réception, le décodage s'effectue de manière itérative à l'aide d'une version modifiée de l'algorithme *aller-retour* (en anglais "forward-backward") développée en 1974 par Bahl, Cocke, Jelinek et Raviv [BCJR74]. L'algorithme *aller-retour* s'apparente à l'algorithme classique de Viterbi en ce qu'elle associe aux différentes transitions entre tous les états d'un treillis (représentant le contenu des mémoires d'un codeur convolutif) des *métriques de branches* qui permettent de déterminer la suite d'états, et donc la suite des blocs d'informations, qui ont été le plus *vraisemblablement transmis* étant donné l'observation des symboles, ou bits reçus. Cependant, contrairement à l'algorithme de Viterbi qui consiste à ne conserver que les *chemins survivants* en examinant le treillis une seule fois dans une unique direction (i.e. à l'*aller* seulement), l'algorithme aller-retour telle que proposée par Bahl, Cocke, Jelinek et Raviv consiste à calculer des valeurs de *fiabilité* pour chacun des états du treillis sans supprimer les transitions (i.e. l'aller) et de

recalculer ces valeurs de fiabilité en repassant dans le treillis dans le sens inverse (i.e. *le retour*).

Il est à noter qu'il n'y a, à proprement dit, de notion d'itération dans la version originale de l'algorithme aller-retour: il n'y a qu'un aller et un retour, alors que les performances obtenues par Berrou, Glavieux et Thitimajshima nécessitent l'application de l'algorithme un certain nombre de fois afin d'améliorer les valeurs de fiabilité de la suite d'états et ainsi obtenir de si bons résultats.

3.2 Codage à protection hiérarchique

À la conférence ICC'94 (1994 IEEE International Conference on Communications), Berrou et Combelles [BC94] ont proposé une méthode de codage de canal *hiérarchique* pour la transmission de la télévision numérique. Dans leur article, ils présentent un système de codage utilisant une modulation 64-QAM ayant une efficacité spectrale globale de 4 bits/s/Hz.

Le système de codage consiste à prendre le bit considéré le plus important parmi des blocs de 4 bits, de coder ce bit particulier avec un code convolutif de rendement $R_1 = \frac{k}{n} = \frac{1}{2}$ et de longueur de contrainte $L = 7$ pour former un symbole QPSK, c'est-à-dire que les deux bits à la sortie du codeur déterminent dans quel quadrant se situe le symbole complexe à transmettre. À la réception, la détermination correcte du quadrant, i.e. le décodage du bit de *haute priorité*, est *cruciale* pour la réception du signal vidéo et corresponds à la *détection grossière* de l'information. Les trois autres bits restants, correspondant à l'*information fine* de l'image vidéo numérique, ou encore aux bits de *basse priorité*, sont eux aussi codés à l'aide d'un code convolutif. Pour ce faire, les auteurs ont choisi le même code convolutif et l'ont *poinçonné* afin d'obtenir un code de rendement $R_2 = \frac{3}{4}$. À chaque bloc codé de 4 bits à la sortie du second codeur on assigne l'un des signaux complexes d'une sous-constellation à $16 = 2^4$ signaux: la sous-constellation 16-QAM. La modulation globale consiste alors à combiner les deux modulations, i.e. QPSK et 16-QAM, de la façon suivante pour former une constellation de signaux 64-QAM: les bits codés de haute priorité déterminent le quadrant de la modulation 64-QAM alors que les 4 bits codés de basse priorité déterminent lequel des 16 signaux 16-QAM est sélectionné (la sous-constellation 16-QAM étant *recopiée* dans chacun des quadrants). Le rendement global est donc de 4 bits d'information par symbole 64-QAM, soit $\eta = 4$ bits/s/Hz.

Les auteurs proposent alors l'emploi de codes Turbo [BGT93] pour améliorer les performances des codes et montrent l'avantage des méthodes de codage dites *hiérarchiques* au niveau de la dégradation progressive du signal. Ils obtiennent ainsi, pour un code Turbo *non hiérarchique* dans un canal gaussien, un taux d'erreur inférieur à 10^{-4} avec un rapport signal à bruit d'environ 15 dB pour une efficacité spectrale $\eta = 4$ bits/s/Hz. Pour un code hiérarchique avec un bit de haute priorité et 3 bits de basse priorité, le même taux d'erreur est obtenu avec un rapport signal sur bruit ≈ 6.5 dB pour le bit de haute priorité (i.e. résolution grossière de l'image) et de ≈ 18 dB pour les 3 bits de basse priorité (résolution fine de l'image). Si l'on compare les deux systèmes, on constate que le codage hiérarchique permet de détecter une image de basse résolution dès que le rapport signal sur bruit dépasse 6.5 dB. La réception à pleine résolution ne peut se faire que lorsque ce rapport atteint 18 dB. Pour le système de codage standard *non hiérarchisé*, on ne peut recevoir l'image qu'à partir de 15 dB de rapport signal sur bruit: cependant le signal est alors reçu à pleine résolution, c'est-à-dire à 4 bits par symbole.

Des résultats comparativement semblables ont été obtenus pour les canaux de transmission affectés par des affaiblissements de Rayleigh. Ainsi, pour le même code standard de rendement $\eta = 4$ bits/s/Hz, Berrou et Combelles obtiennent un taux d'erreur inférieur à 10^{-4} lorsque le rapport signal-à-bruit excède 18 dB, alors qu'avec le code hiérarchique, on reçoit déjà l'information grossière (bit de haute priorité) à moins de 10 dB de rapport signal à bruit. Toutefois l'image à pleine résolution ne s'obtient qu'avec un rapport d'environ 22 dB pour le code hiérarchisé.

3.3 Modulation codée multiniveau

Les méthodes de modulation codée multiniveau permettent de former une catégorie de codes correcteurs pour lesquels on peut ajuster aisément l'efficacité spectrale et la capacité de correction. Le concept de code multiniveau a été présenté par Imai et Hirakawa [IH77] en 1977 et étudié par la suite par Sayegh [Say86] ainsi que Pottie et Taylor [PT89].

Le principe de la modulation codée multiniveau [Say86] consiste en tout premier lieu à partitionner l'ensemble des 2^M signaux d'une constellation en 2^M sous-ensembles contenant chacun 1 seul symbole. Pour la modulation codée multiniveau (voir Figure 3.1), les bits correspondant aux différents niveaux de partitionnement, c'est-à-dire $1 \leq m \leq M$ sont codés indépendamment les uns des autres.

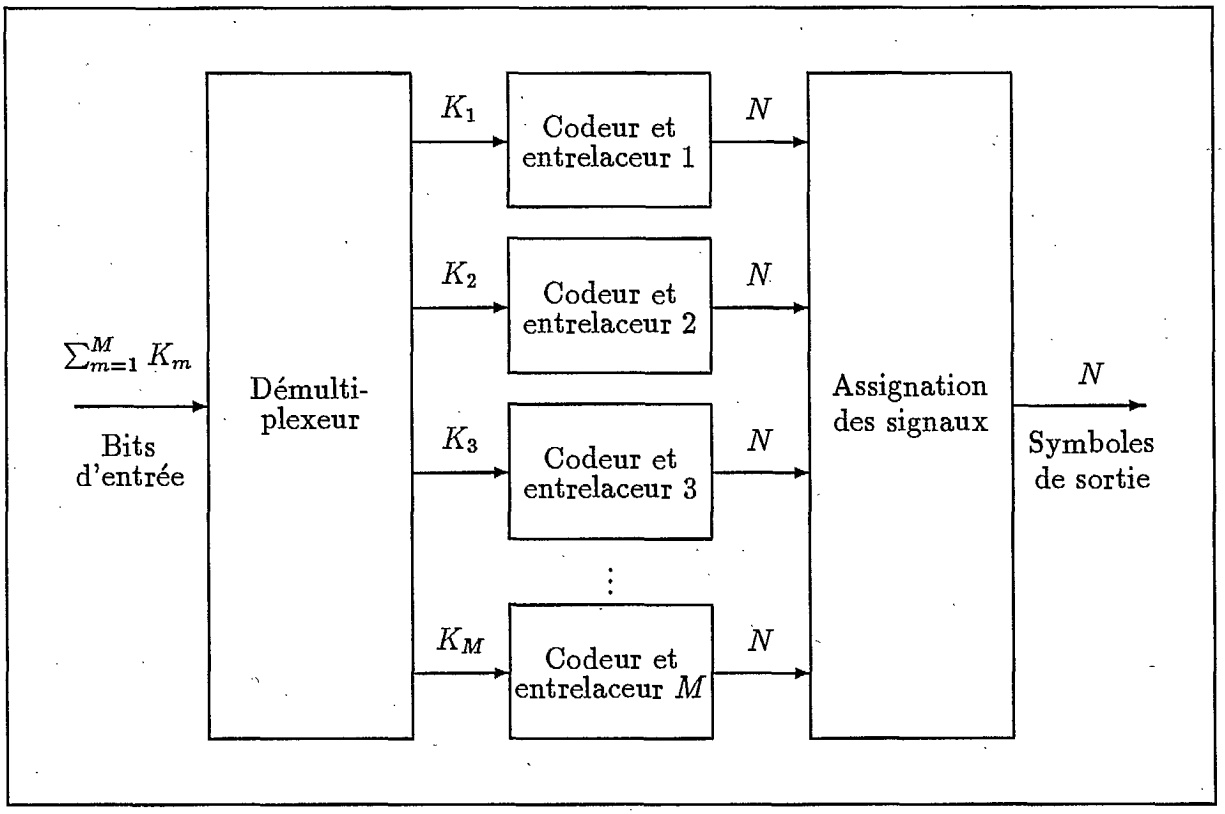

Figure 3.1: Principe de la modulation codée multiniveaux.

La transmission des symboles se fait par blocs de N symboles complexes provenant de la constellation. Pour chacun des N symboles, il y a N bits *codés et entrelacés* qui proviennent simultanément de chaque niveau de partitionnement: plus précisément, pour chaque bloc de N symboles, il y a, à chaque niveau de partitionnement, K_m bits d'information qui sont codés à l'aide d'un *code composante* C_m, $m = 1, \ldots, M$ de rendement $R_m = \frac{K_m}{N}$ qui sont codés sur exactement N bits puis entrelacés. À chaque instant, les M bits à la sortie des M codeurs (un pour chaque niveau de partitionnement) servent à assigner un symbole de la constellation de signaux. L'efficacité spectrale η en nombre de bits par symbole complexe est alors définie par:

$$\eta = \frac{1}{N} \sum_{m=1}^{M} K_m = \sum_{m=1}^{M} R_m$$

Cette méthode de codage permet d'exploiter conjointement l'efficacité spectrale de la modulation codée et la capacité de correction du décodage itératif.

La modulation codée multiniveau a conduit à des résultats intéressants dans le contexte spécifique de la transmission numérique de la parole sur un canal de transmission mobile terrestre à satellite [PYC96]. Le taux d'erreur ne devait pas excéder 10^{-3}, ce qui est considéré en pratique comme le taux d'erreur maximum admissible pour le bon fonctionnement des décodeurs de source, c'est-à-dire pour pouvoir reconstituer la trame vocale avec une bonne fiabilité.

Même si les critères de transmission de signaux de télévision numérique et de voix numérisée diffèrent considérablement et, que de surcroît, les canaux étudiés ne sont pas les mêmes, il apparait plausible d'envisager que la modulation codée multiniveau puisse s'appliquer avec succès à la transmission de la télévision numérique en permettant une bonne fiabilité et *dégradation plus graduelle* de cette dernière.

3.4 Conclusion

Dans ce chapitre, nous nous sommes intéressé à l'utilisation des nouvelles méthodes de codage adaptées la transmission *progressive* de la télévision numérique à haute définition. La technique de décodage itératif employée dans le contexte de modulation codée à niveaux multiples s'avère prometteuse pour protéger la transmission d'information, à savoir le signal de télévision numérique, en *paliers de qualité* ou encore dans le contexte particulier de la télévision numérique en *paliers de résolution vidéo*.

Annexe A

Listings des programmes

A.1 Programme "vsb_h.cpp"

Le programme *vsb_h.cpp* génère la fonction de transfert du filtre à bande résiduelle (i.e. VSB) et l'emmagasine dans le fichier *vsb_ftr.txt*.

```
//  vsb_h.cpp
//  this program generates the frequency response of
//  the VSB filter to the file vsb_ftr.txt
//  David Belcourt  for CRC
#include <stdlib.h>
#include <stdio.h>
#include <time.h>

//  prototypes

int generator(void);
void initial(void);

//  global variable

int x = 0;
float H;
FILE *pt;

void main(void)
{

 pt = fopen("vsb_ftr.txt", "w+");  //  open file
 for(x=0; x<20000 ; x++)
{
if((x>=0)&&(x<=1400))
H = ((x+1400.0)/2800.0);
if((x>1400)&&(x<=5600))
H = 1.0;
```

```
if((x>=5600)&&(x<18600))
H = 0.0;
if((x>=18600)&&(x<20000))
H = (x-18600.0)/2800.0;
fprintf(pt, "%f \n", H); //  write to file
}
 fclose(pt);                  //  close file
}
```

A.2 Programme "qpsk_mod.cpp"

Le programme *qpsk_mod.cpp* effectue la modulation d'une suite binaire en signal QPSK.

```
//  qpsk_mod.cpp
//  this program modulates a QPSK signal
//  David Belcourt  for CRC
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#include <math.h>

//  prototypes

float in_phase_comp(double A1, double n);
float quad_comp(double A2, double n);

//  global variable

int  num, sym_num;
float pi, E, I_comp, Q_comp, QPSK_out;
char symbol_in, temp[6];
FILE *pt_sym_in, *pt_QPSK_out;
long fc, fs, baude_rate, cnt=0;
double a1, a2, A1, A2, t, Ts, Time_sym, n, next_sym;

void main(void)
{

//  initial parameters

fc = 10000001; // carrier frequency at base band in Hz
fs = 123000001;  //  sampling frequency in Hz

baude_rate = 5000001; //  baude rate of symbols

printf("carrier frequency is: %ld \n", fc);
printf("sampling frequency is: %ld \n", fs);
printf("baude rate is: %ld \n", baude_rate);

//  constants

pi = 3.141592654;
Time_sym = 1.0/baude_rate;
Ts = 1.0/fs;
next_sym = 0;
t = 0.0;
```

```
// inital proc.

pt_sym_in = fopen("symboles.txt", "r+");        // open files
pt_QPSK_out = fopen("qpsk_out.txt", "w+");
 /*
printf("Enter the number of symbols to be modulated: ");
num = atoi(gets(temp));   // get the number of symbols
*/
num = 20330;                        // number of symbols generated
/*
printf("Enter the energy per bit: ");
E = atof(gets(temp));             // get energy of each bit
*/
E = 0.000001;                    // Energy of each bit

// do this loop until the num. of sym. is equal to that specified

for(sym_num = 0; sym_num <= num; )

  {
  t += Ts;
  if(t >= next_sym)
{
fscanf(pt_sym_in, "%d", &symbol_in); // get symbol from file
next_sym += Time_sym;   // set time of next symbol
sym_num +=1; // cnt. the number of sym. modulated
switch(symbol_in)       // set phase arguments of I & Q
{
case 0:
a1 = -1 * sqrt(E);
a2 = -1 * sqrt(E);
break;
case 1:
a1 = -1 * sqrt(E);
a2 = 1 * sqrt(E);
break;
case 2:
a1 = 1 * sqrt(E);
a2 = -1 * sqrt(E);
break;
case 3:
a1 = 1 * sqrt(E);
a2 = 1 * sqrt(E);
break;
/* case  :             // Indicate error
```

```
printf("ERROR, not enought symbols ");
break;
*/
default:
printf("ERROR with the conversion #1");
break;
}
}

I_comp = in_phase_comp(a1, t);  //  call for fcts. for I & Q components
Q_comp = quad_comp(a2, t);
QPSK_out = I_comp + Q_comp;
cnt++;
fprintf(pt_QPSK_out, "%f\n", QPSK_out);  //  write to file
      if(cnt >= 5000001)
break;
}
fclose(pt_sym_in);          //  close files
fclose(pt_QPSK_out);
printf("END   %ld samples generated",cnt);
}

//  functions  generating I & Q components

float in_phase_comp(double A1,double n)
{
float Si;
Si = A1 * sqrt(2 * baude_rate) * cos(2 * pi *fc * n);
return(Si);
}

float quad_comp(double A2,double n)
{
float Sq;
Sq = A2 * sqrt(2 * baude_rate) * sin(2 * pi *fc * n);
return(Sq);
}
```

A.3 Programme "qpsk_dmo.cpp"

Le programme *qpsk_dmo.cpp* effectue la démodulation d'une suite de N symboles QPSK en suite binaire.

```
//  qpsk_dmo.cpp
//  this program demodulates N symbols a QPSK signal
//  David Belcourt for CRC
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#include <math.h>

//  prototypes

float in_phase_comp(float sample_in, double n);
float quad_comp(float sample_in, double n);


//  global variable


int  num, sym_num, new_sym;
float pi, I_comp, Q_comp, A1, A2;
char temp[12];
FILE *pt_sample,*pt_data_out;
long fc, fs, baude_rate, cnt=0, sample_num;
double  t, Ts, T, Tb, Time_sym, n, next_sym, sample_in, int_I, int_Q;

void main(void)
{

//  initial parameters

fc = 10000001; // carrier frequency at base band in Hz
fs = 123000001;  //  sampling frequency in Hz

baude_rate = 5000001; //  baude rate of symbols

//  constants
T = 1.0/fc;
pi = 3.141592654;
Time_sym = 1.0/baude_rate;
Ts = 1.0/fs;
next_sym = 0;
t = 0.0;
Tb = 0.0;
```

```
//  inital proc.

pt_sample = fopen("qpsk_out.txt", "r");        //  open files
pt_data_out = fopen("new_sym.txt", "w+");

//  do this loop until the num. of samples is equal to that specified

for(sample_num = 0; sample_num <= 5000001; )

  {
  int_I = 0.0;
  int_Q = 0.0;
     Tb = Tb + T;
  next_sym = next_sym + Time_sym;   //  set time of next symbol
  while(t <= next_sym)
{
fscanf(pt_sample, "%s", &temp);  //  get sample from file
sample_in = atof(temp);
I_comp = in_phase_comp(sample_in, t);  //  call for fcts. for I & Q components
Q_comp = quad_comp(sample_in, t);
t = t + Ts;
sample_num++;   // sampling time
int_I = int_I + I_comp;       //  integral over one symbole
int_Q = int_Q + Q_comp;
}

//  Decision device

  if((int_I < 0.0) && (int_Q < 0.0))
new_sym = 0;
  if((int_I < 0.0) && (int_Q > 0.0))
new_sym = 1;
  if((int_I > 0.0) && (int_Q < 0.0))
new_sym = 2;
  if((int_I > 0.0) && (int_Q > 0.0))
new_sym = 3;
  if((int_I == 0.0) || (int_Q == 0.0))
new_sym = 9;    // error

 // printf("I %f   Q  %f    sym  %i  next %lf \n",int_I, int_Q, new_sym, next_sym);
  cnt++;     //  number of symboles
  fprintf(pt_data_out, "%i\n", new_sym);  //  write to file
  int_I = 0.0;
  int_Q = 0.0;
  }
fclose(pt_sample);          //  close files
```

```
fclose(pt_data_out);
printf("END  %i symboles written",cnt);
}

//  functions  generating I & Q components

float in_phase_comp(float A1,double n)
{
float Si;
Si = A1 * sqrt(2 * baude_rate) * cos(2 * pi *fc * n);
return(Si);
}

float quad_comp(float A2,double n)
{
float Sq;
Sq = A2 * sqrt(2 * baude_rate) * sin(2 * pi *fc * n);
return(Sq);
}
```

A.4 Programme "not_ftr.cpp"

Le programme *not_ftr.cpp* produit la réponse en fréquence du filtre à encoche ("notch filter") et la place en mémoire dans le fichier *not_ftr.txt*.

```
//  not_ftr.cpp
//  this program generates the frequency response of
//  the notch filter to the file not_ftr.txt
//  David Belcourt for CRC
#include <stdlib.h>
#include <stdio.h>
#include <time.h>

//  prototypes

int generator(void);
void initial(void);

//  global variable

int x = 0;
float H;
FILE *pt;

void main(void)
{

 pt = fopen("not_ftr.txt", "w+");  //  open file
 for(x=0; x<143 ; x++)
{
if(x==133)
H = 1;
else
H = 0;

fprintf(pt, "%f \n", H); //  write to file
}
 fclose(pt);                //  close file
}
```

A.5 Programme "im_lpf.cpp"

Le programme *im_lpf.cpp* produit la réponse en fréquence du filtre pour le signal modulé en QPSK (i.e. la porteuse QPSK).

```
//  im_lpf.cpp
//  this program generates the frequency response of
//  the filter for the qpsk carrier
//  David Belcourt  for CRC
#include <stdlib.h>
#include <stdio.h>
#include <time.h>

//  prototypes

int generator(void);
void initial(void);

//  global variable

long x = 0;
int H;
FILE *pt;

void main(void)
{

 pt = fopen("im_lpf.txt", "w+");  //  open file
 for(x=0.0; x<5000001 ; x++)
{
if((x>=0)&&(x<4400001))
H = 0;
if((x>=4400001) && (x < 4800001))
H = 1;
if((x>4800001)&&(x<5000001))
H = 0;
fprintf(pt, "%d \n", H); //  write to file
}
 fclose(pt);                  //  close file
}
```

A.6 Programme "error_ct.cpp"

Le programme *error_ct.cpp* compare les données transmises et reçues et compte le nombre d'erreurs résultant.

```
//  error_ct.cpp
//  this program counts the errors
//  David Belcourt for CRC
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>

char temp[2];
FILE *pt_old, *pt_new;
int old_in = 0, new_in = 0, diff = 0, error = 0, cnt = 0;

void main(void)
{
pt_old = fopen("symboles.txt", "r");        //  open files
pt_new = fopen("new_sym.txt", "r");

do                  //  check all sym.
{
fscanf(pt_old, "%s", &temp);  //  get sym from  old file
old_in = atoi(temp);
fscanf(pt_new, "%s", &temp);  //  get sym from new file
new_in = atoi(temp);
diff = old_in - new_in;
if(diff != 0)                 //  check difference
error++;
cnt++;                        //  cnt errors
if(cnt > 20400)
break;
 // clrscr();
// printf("%i",cnt);
}
while(cnt < 20326);
printf("End   %i errors counted  out of %i symboles", error, cnt);
fclose(pt_old);     //  close files
fclose(pt_new);
}
```

A.7 Programme "gen_sym.cpp"

Le programme *gen_sym.cpp* génère N symboles quaternaires et les emmagasine dans le fichier *symboles.txt*.

```
//  gen_sym.cpp
//  this program generates N quaternary symbols in to the file symboles.txt
//  David Belcourt for CRC
#include <stdlib.h>
#include <stdio.h>
#include <time.h>

//  prototypes

int generator(void);
void initial(void);

//  global variable

int loop = 0;
int out;
char symbol;
FILE *pt;
long int j, num;

void main(void)
{
 initial();                                  //  initial proc. for random gen.
 pt = fopen("symboles.txt", "w+");  //  open file
 printf("Enter the number of symbols to be generated:  ");
 scanf("%d",&num);    //  get the number of symbols

 do //  generate N symbols
 {
loop++;
out = generator(); //  call generator fct
fprintf(pt, "%d\n", out); //  write to file
 }
 while( loop < num );
 fclose(pt);                    //  close file
    printf("end");
}

//  subroutine generator

int generator(void)
```

```
{
j = random(32767);            //  call for pseudo-random gen.
if(j < 8192)
symbol = 0;
if((j < 16383)&&(j >= 8192))              //  4 possible symbols
symbol = 1;
if((j >= 16383)&&(j < 24576))
symbol = 2;
if(j >= 24576)
symbol = 3;
return(symbol);
}                             //  end generator()

//  subroutine ini1

void initial(void)
{
randomize();          //  initalize rand. gen. with rand. value
}
```

Bibliographie

[Bat93] G. Battail. Pseudo-Random Recursive Convolutional Coding for Near-Capacity Performances. In *International Symposium on Communications Theory and Applications*, juillet 1993.

[BC94] C. Berrou and P. Combelles. Digital Television: Hierarchical Channel Coding Using Turbo Codes. In *International Communications Conference (ICC'94)*, pages 1255–1259. Institute of Electrical and Electronics Engineers (IEEE), 1994.

[BCJR74] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv. Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate. *IEEE Transactions on Information Theory*, 20:284–287, mars 1974.

[BDMS91] E. Biglieri, D. Divsalar, P.J. McLane, and M.K. Simon. *Introduction to Trellis-Coded Modulation with Applications.* Macmillan Publishing Company, New-York, 1991.

[BGT93] C. Berrou, A. Glavieux, and P. Thitimajshima. Near Shannon Limit Error-Correcting Coding: Turbo Codes. In *ICC'93*, pages 1064–1070, 1993. Genève.

[Bla84] R.E. Blahut. *Theory and Practice of Error Control Codes.* Addison-Wesley, Reading, Massachusetts, 1984.

[Bla87] R. Blahut. *Principles and Practice of Information Theory.* Addison-Wesley, Reading, Massachusetts, 1987.

[Bor94] Borland. *Borland C++ Reference Manual.* Borland International Inc., Scotts Valley, Californie, 1994.

[CCT96] J.-Y. Chouinard, D. Cayer, and M. Trichard. Codes de correction d'erreurs pour la transmission de la télévision numérique - phase 2. Technical Report 67qsg-5-6181, Université d'Ottawa, Ottawa, Ontario, Canada, mars 1996.

[CDG92] G. Cohen, J.-L. Dornstetter, and P. Godlewski. *Codes correcteurs d'erreurs: une introduction au codage algébrique.* Masson, Paris, 1992. Collection Technique et Scientifique des Télécommunications *CNET-ENST.*

[CETM95] A. Chini, M. El-Tanany, and S. Mahmoud. Application of a Filtered Decision Feedback Channel Estimator in Multi Carrier Digital TV Broadcasting. Technical Report 67CRC-4-0577, Carleton University, Ottawa, Ontario, Canada, mars 1995.

[CL95] D. Cygan and E. Lutz. A Concatenated Two-Stage Adaptive Error Control Scheme for Data Transmission in Time-Varying Channel. *IEEE Transactions on Communications*, 43:795–803, avril 1995.

[CT95] J.-Y. Chouinard and M. Trichard. Codes de correction d'erreurs pour la transmission de la télévision numérique - phase 1. Technical Report 67CRC-4-0423, Université d'Ottawa, Ottawa, Ontario, Canada, mars 1995.

[Dig94] Digideck. D-channel: A Technique for NTSC-Compatible Data Broadcasting. Technical report, Digideck Inc., Menlo Park, Californie, avril 1994.

[For66] G. D. Forney. *Concatenated Codes*. MIT Press, 1966.

[For70] G.D. Forney. Convolutional Codes I: Algebraic Structure. *IEEE Transactions on Information Theory*, IT-16(6):720–738, novembre 1970.

[Hag88] J. Hagenauer. Rate Compatible Punctured Convolutional Codes (RCPC Codes) and Their Applications. *IEEE Transactions on Communications*, 36:389–400, avril 1988.

[Hay94] S. Haykin. *Communication Systems*. John Wiley and Sons, New-York, 1994.

[HCH88] M. A. Herro, D. J. Costello, and L. Hu. Capacity and Cutoff Rate Calculations for a Concatenated Coding System. *IEEE Transactions on Information Theory*, 34:212–222, mars 1988.

[HHH89] J. Hagenauer, P. Hoeher, and J. Huber. A Viterbi Algorithm with Soft Decision Outputs and its Applications. In *GLOBECOM'89*, novembre 1989.

[HLP93] C. Heegard, S.A. Lery, and W.H. Paik. Practical Coding for QAM Transmission of HDTV. *IEEE Journal on Selected Areas in Communications*, JSAC-11(1):111–118, janvier 1993.

[IH77] H. Imai and S. Hirakawa. A New Multilevel Coding Method using Error Correcting Codes. *IEEE Transactions on Information Theory*, 23:371–377, 1977.

[KZS94] Y. Kofman, E. Zehavi, and S. Shamai. Performance Analysis of A Multilevel Coded Modulation System. *IEEE Transactions on Information Theory*, 42:299–312, février 1994.

[Lab95] Cable Television Laboratories. Grand Alliance HDTV System Completes Laboratory and Field Tests. Technical report, Cable Television Laboratories Inc., Louisville, Colorado, septembre-octobre 1995.

[LC83] S. Lin and D. Costello. *Error Control Coding: Fundamentals and Applications*. Prentice-Hall, New-Jersey, 1983.

[Lee77] L.-N. Lee. Concatenated Coding Systems Employing a Unit-Memory Convolutional Code and a Byte-Oriented Decoding Algorithm. *IEEE Transactions on Communications*, COM-25(10):1064–1074, octobre 1977.

[LGB94] S. Le Goff, A. Glavieux, and C. Berrou. Turbo-Codes and High Spectral Efficiency Modulation. In *ICC'94*, pages 645–649, 1994.

[LY93] J. Lodge and R. Young. Separable Concatenated Codes with Iterative MAP Decoding for Rician Fading Channel. In *International Mobile Satellite Conference*, pages 467–472, 1993.

[LYHH93] J. Lodge, R. Young, P. Hoeher, and J. Hagenauer. Separable MAP "Filters" for the Decoding of Product and Concatenated Codes. In *ICC'93*, pages 1740–1745, 1993. Genève.

[MMV95] V. Mignone, A. Morello, and M. Visintin. CD3-OFDM: A New Channel Estimation Method to Improve the Spectrum Efficiency in Digital Terrestrial Television Systems. In *International Broadcasting Convention (IBC'95)*, number 413 in IEE Conference Publication, pages 122–128, Amsterdam, 14 au 18 septembre 1995. Institution of Electrical Engineers (IEE).

[Orf96] S.J. Orfanis. *Introduction to Signal Processing.* Prentice-Hall Signal Processing Series. Prentice-Hall, Upper Saddle River, New-Jersey, 1996.

[Ple89] V. Pless. *Introduction to the Theory of Error-Correcting Codes.* Wiley-Interscience Series, New-York, 1989.

[Pro95] J. Proakis. *Digital Communications.* McGraw-Hill, 1995. troisième édition.

[PT89] G. Pottie and D. Taylor. Multilevel Codes Based on Partitioning. *IEEE Transactions on Information Theory*, 35:89–98, janvier 1989.

[PYC96] M. Pénicaud, A. Yongaçoglu, and J.-Y. Chouinard. Iterative Decoding of Rate Adaptive Multilevel Coded Modulation for Mobile Satellite Communication. In *1996 Global Telecommunications Conference (GLOBECOM'96)*, pages 415–419. Institute of Electrical and Electronics Engineers (IEEE), novembre 1996.

[RHG93] S.A. Raghavan, Y. Hebron, and I. Gurantz. On the Application of Appropriate APP Decoding to Digital Video Transmission. *IEEE Journal on Selected Areas in Communications*, JSAC-11(1):136–145, janvier 1993.

[RL93] S. Rajpal and S. Lin. Product Coded Modulation. In *GLOBECOM'93, Communication Theory Mini-Conference*, pages 7–11, 1993.

[Rze95] T.S. Rzeszewski. *Digital Video: Concepts and Applications Across Industries.* IEEE Press, Piscataway, 1995.

[Say86] S. Sayegh. A Class of Optimum Block Codes Based on Partitioning. *IEEE Transactions on Communications*, pages 1043–1045, octobre 1986.

[Tri95] M. Trichard. Study of Trellis Coded Modulation and Error Control Coding. Master's thesis, Université d'Ottawa, Ottawa, Ontario, Canada, octobre 1995.

[Ung82] G. Ungerboeck. Channel Coding with Multilevel/Phase Signals. *IEEE Transactions on Information Theory*, IT-28(1):55–67, janvier 1982.

[Ung87a] G. Ungerboeck. Trellis-Coded Modulation with Redundant Signal Sets; Part I: Introduction. *IEEE Communications Magazine*, 25(2):5–11, février 1987.

[Ung87b] G. Ungerboeck. Trellis-Coded Modulation with Redundant Signal Sets; Part II: State of the Art. *IEEE Communications Magazine*, 25(2):12–21, février 1987.

[VWZP89] A. Viterbi, J. Wolf, E. Zehavi, and R. Padovani. A Pragmatic Approach to Trellis-Coded Modulation. *IEEE Communications Magazine*, 27:11–19, juillet 1989.

[WB94] S.B. Wicker and V.K. Bhargava. *Reed-Solomon Codes and Their Applications*, chapter 11, pages 242–271. IEEE Press, Piscataway, 1994. Chapitre rédigé par J. Hagenauer, E. Offer et L. Papke et intitulé *Matching Viterbi Decoders and Reed-Solomon Decoders in a Concatenated System.*

[WC94] Y. Wu and B. Caron. Digital Television Terrestrial Broadcasting. *IEEE Communications Magazine*, 32(5):46–52, mai 1994.

[WGCT95] Y. Wu, M. Guillet, J.-Y. Chouinard, and M. Trichard. COFDM for Digital ATV Terrestrial Distribution over 6 MHz Channels. In *International Broadcasting Convention (IBC'95)*, number 413 in IEE Conference Publication, pages 29–34, Amsterdam, 14 au 18 septembre 1995. Institution of Electrical Engineers (IEE).

[WH92] T. Woerz and J. Hagenauer. Iterative Decoding for Multilevel Codes using Reliability Information. In *GLOBECOM'92*, pages 1779–1784, 1992.

[WLC94] Y. Wu, B. Ledoux, and B. Caron. Evaluation of Multi-Carrier Transmission Systems for Advanced Television in North America. In *NAB 1994 Broadcast Engineer Conference Proceedings*, mars 1994.

[YKH84] Y. Yasuda, K. Kashiki, and Y. Hirata. High Rate Punctured Convolutional Codes for Soft Decision Viterbi Decoding. *IEEE Transactions on Communications*, 32:315–319, mars 1984.