Présentation de la norme H.263++ : protection du flux vidéo dans un environnement mobile

J.C. Amiel[1]

[1] France Telecom R&D/DMI/SGV (laboratoire de Service de visiophonie de Groupe et de Visioconférence)

France Télécom R&D,
39 avenue du Général Leclerc 92794 Issy Les Moulineaux Cedex 9 France

jeanchristophe.amiel@francetelecom.com

Résumé

L'objectif de ce document est de présenter les algorithmes de protection d'un flux vidéo H.263++ codé dans un environnement de mobilité. Dans un tel environnement, des erreurs binaires affectent le train vidéo et les mécanismes classiques de masquage d'erreur lors de pertes de paquets sont inefficaces. On montrera qu'il est nécessaire d'adopter une approche conjointe entre le codage source et le codage canal afin d'optimiser la qualité du flux vidéo reconstruit.

Mots clefs

Compression vidéo, H.263++, protection contre les erreurs, environnement mobile, codage conjoint source-canal.

1 Introduction

La qualité des communications multimédia dans un environnement mobile est un sujet en pleine expansion et d'importance croissante, alors que de nombreuses normes de compression vidéo prennent en compte les limitations des réseaux IP fixes en terme de qualité de transmission. Dans un scénario de communication entre un utilisateur sur un terminal fixe et un utilisateur en situation de mobilité, les paquets contenants les informations vidéo codées émis par l'utilisateur fixe vont passer d'un réseau Internet filaire à un réseau Internet sans fil où les problématiques de qualité et de protection des erreurs sont différentes.

Sur l'Internet filaire, les seules causes de pertes d'informations dans le flux vidéo codé sont les pertes de paquets. De nombreux algorithmes existent et permettent de protéger le train binaire codé contre ces pertes ; ces mécanismes sont présents sous formes d'options ou d'annexes dans les dernières normes vidéo (H.263+, MPEG4, H.264). La norme **H.263+** [1] a introduit un certain nombre de ces mécanismes ; parmi les plus efficaces, on trouve l'annexe N (ou *Back Channel*), qui permet l'utilisation d'un canal de retour pour transmettre des messages du décodeur au codeur afin qu'il puisse réagir en fonction des pertes. Elle permet également de prédire en fonction d'images de référence plus anciennes que la précédente. L'annexe K, ou *Slice Structured Mode*, autorise un découpage plus fin de l'image, et une meilleure adaptation à la mise en paquets. Ces mécanismes sont aussi présent dans MPEG4 visual, sous les noms de *Resync Marker* et *NEWPRED* [2]. Quand un paquet vidéo est perdu, le décodeur est informé de cette perte par la couche de transport : les paquets RTP reçus contiennent un nombre incrémenté d'une unité à chaque nouveau paquet, ce qui permet au décodeur de détecter un paquet manquant et d'activer des algorithmes de correction et de masquage d'erreur.

Sur l'Internet mobile, la problématique de résistance aux erreurs est différente. Le taux d'erreur binaire sur la liaison radio est si important qu'il n'est pas possible d'éliminer tous les paquets contenant des erreurs ; en plus des paquets perdus, l'application devra faire face à des paquets erronés contenants des erreurs binaires. Dans un environnement de mobilité, il faut donc prendre en compte les pertes de paquets, les paquets erronés (contenant des erreurs bits) et les paquets sans erreur. Nous allons montrer l'efficacité des mécanismes de correction d'erreurs de bits dans le cas de la norme **H.263++** et la nécessité d'une approche conjointe entre le codage source et le codage canal. Ces mécanismes sont aussi présents dans les normes **MPEG-4**, et la future norme **H-264**, ce qui en montre toute l'importance.

1.1 De H.263+ à H.263++

H.263 version 3 (2000)[3], également appelée H.263++, définit trois options et une spécification à la version précédente du codec H.263 (H.263 version 2-H.263+): l'**annexe U - Enhanced reference picture selection mode**, **l'annexe V - Data-partitioned slice mode**, et l'**annexe W - Additional supplemental enhancement information**. L'option la plus intéressante dans un environnement dont le taux d'erreur bit est très élevé, est

l'annexe V, qui introduit un certain nombre de mécanismes contres les erreurs binaires du flux vidéo. C'est cette option à laquelle nous allons nous intéresser dans cette étude ; il s'agit d'évaluer les performances de correction et de masquage d'erreur en présence d'erreurs binaires affectant le train vidéo. L'annexe V a été conçu pour des débits faibles (64 kbps)[4], et des taux d'erreurs de bits avoisinant le 1/1000. Elle repose sur quatre éléments :

- **la structure d'image en slices** définit par l'annexe K de H.263+ (les slices sont des structures macroscopiques de l'image et correspondent à des parties d'image de formes variables),
- **la contrainte d'indépendance des données** entre slices (annexe R de H.263+),
- **la séparation hiérarchique des données** (*data partitionning*),
- et **l'utilisation de mots de code à longueur variable réversibles** ou *Rerversible Variable Length Code* (RVLC).

1.2 Structures de slices

Dans le train vidéo, on utilise des structures permettant d'organiser les données. Les slices, introduit dans H.263+, sont des groupes de macroblocs de taille variable. Les slices permettent un découpage plus fin de l'image, une robustesse par rapport aux pertes de paquets, et une gestion plus fine de la paquetisation. Idéalement, un slice est encapsulé dans un paquet RTP. L'annexe V reprend la structure de slice en la complétant. Ainsi, la protection contre les erreurs de paquets apportée par cette structure est également présente dans un flux codé H.263++.

1.3 Data partitioning et RVLC

A l'intérieur de chaque slice, on redéfinit l'organisation et l'arrangement des données vidéo, sans changer leur signification, en trois partitions : une pour les données d'en-tête, une pour les vecteurs mouvement, et une dernière pour les coefficients de la transformation en cosinus discrète (coefficients DCT).

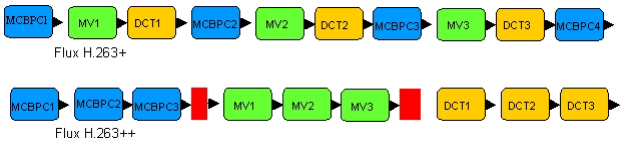

Figure 1- *Structure des flux H.263+ et H.263++*

En se basant sur cette structure hiérarchique et en utilisant un paquet par partition, on peut limiter l'impact des erreurs. Par exemple, des erreurs de bits dans la partition DCT affecteront peu la reconstruction de l'image si l'on a correctement décodé les en-têtes et les vecteurs mouvement. Par ailleurs, on utilise des RVLC pour les données d'en-têtes et les vecteurs mouvement ; entre chaque partition, on insère des marqueurs de synchronisation (Figure 1 et Figure 2). Les RVLC peuvent être décodés dans les deux sens de lecture (*backward*, au fil de l'eau du train binaire et *forward*, à rebours du train binaire) et peuvent ainsi améliorer la détection d'erreurs. Lors du démultiplexage du flux vidéo H.263++, plusieurs stratégies de correction peuvent être alors adoptées.

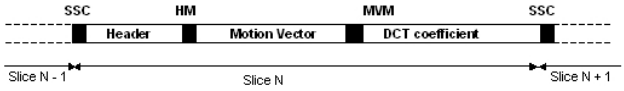

Figure 2 – *Un slice contient H.263++ trois partitions (SSC=marqueur de slice, HM=marqueur d'en-têtes, MVM=marqueur de vecteurs mouvement)*

Lors de la détection d'une erreur de décodage, une première stratégie de correction consiste à se synchroniser sur le prochain marqueur de slice. On perd toutes les données suivantes et on récupère les données manquantes à l'aide de l'image précédente. Cette approche est similaire à celle que l'on adopte dans un flux codé en H.263+, où le seul point de resynchronisation possible est le début d'un nouveau slice. Dans la suite de l'étude, nous désignerons cette stratégie sous le nom de *H.263++ - synchronisation sur prochain slice* (Figure 3).

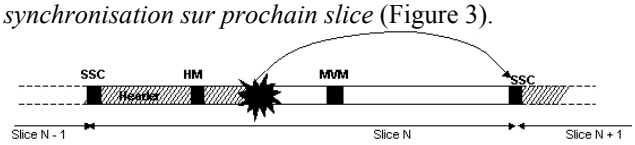

Figure 3 – *Synchronisation sur prochain slice*

Une deuxième approche consiste à utiliser les marqueurs dans le slice introduit par l'annexe V. Si l'erreur a lieu dans la partition des en-têtes et que l'on arrive à se synchroniser sur le marqueur de fin d'en-tête, on peut alors continuer à lire les données des vecteurs mouvement et des coefficients de la DCT. Une telle approche sera désignée par la suite sous le nom de *H.263++ - synchronisation sur prochain marqueur* (Figure 4).

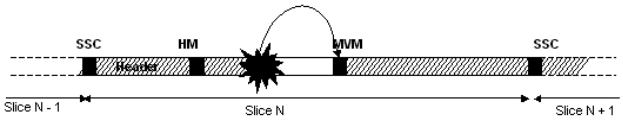

Figure 4 – *Synchronisation sur prochain marqueur*

Enfin, une dernière approche consiste à utiliser les RVLC disponibles dans les partitions d'en-têtes et de vecteur mouvement (les mots de codes de la partition des coefficients de la DCT étant identiques à ceux utilisés dans le codec H.263+(VLC), il est impossible de chercher à parcourir le train binaire de cette partition en sens inverse). Une fois synchronisé sur le marqueur de fin de partition, on peut lire le flux vidéo à l'envers pour récupérer le plus de données possibles. Le partitionnement de données est particulièrement intéressant dès l'instant où la couche canal peut réaliser de la protection inégale : il s'agira de protéger de manière plus importante les partitions d'en-têtes.

1.4 Difficultés

L'une des principales difficultés dans le masquage d'erreur vient du fait que l'endroit où un bit est erroné ne correspond pas à l'endroit où l'on se rend compte qu'il y a eu une erreur (souvent plusieurs dizaines de bits séparent l'erreur réelle et le point où l'on détecte une incohérence). De plus, certains bits faux peuvent être transparents (au sens où ils ne provoquent pas d'erreur de démultiplexage) et avoir des conséquences désastreuses sur le rendu de l'image finale. Une autre difficulté provient de l'utilisation des RVLC : à cause des contraintes de réversibilité sur ces mots de codes, la longueur moyenne des RVLC est généralement plus importante que celle des VLC, ce qui entraîne une perte d'efficacité de compression. Pourtant, on peut construire des dictionnaires de RVLC donc la longueur des mots correspond à la distribution de probabilité des symboles (cas idéal) ; ainsi dans un environnement sans erreur, l'utilisation de RVLC entraîne normalement peu de pertes d'efficacité de compression [5]. De plus, il a été démontré que la probabilité de propagation d'une erreur est plus importante pour des RVLC que pour des VLC, à cause notamment de la structure des mots de codes RVLC. L'efficacité de correction et de masquage des erreurs avec les RVLC dépend de l'endroit où est détectée l'erreur dans le train binaire et de l'étendue de la propagation des erreurs [6].

2 Résultats expérimentaux

Deux séquences sont adoptées pour les tests : la séquence *Foreman* en QCIF, 10 images par seconde, codée à un débit de 70 kbps et la séquence *Man* en QCIF, 12 images par secondes, codée à un débit de 70 kpbs. *Foreman* comporte initialement 100 images; afin d'obtenir des résultats significatifs, on copie en boucle cette séquence en vue d'avoir une séquence de 1000 images soit 1 minute 40s à 10 images par seconde. La séquence *Foreman* est une séquence de base pour des tests de performances de codage et comporte beaucoup de mouvements. En revanche, la séquence *Man* comporte 1000 images et elle est plus statique que la séquence *Foreman*. Ces deux séquences vont permettre d'étudier le comportement de l'annexe V sur des séquences très mouvementées et statiques. En se basant sur les recommandations d'implémentations pour H.263++[4], on étudie les séquences pour un débit de 70 kbps, une taille de slice allant de 100 bits à 800 bits, et une fréquence d'images peu élevée (10-12 images par seconde).

2.1 Décodage : influence des différentes stratégies

On analyse l'impact de la stratégie de décodage sur le SNR moyen des séquences, en comparant pour une même séquence le SNR moyen d'un flux H.263+ décodé, le SNR moyen d'un flux H.263++ décodé avec une synchronisation sur prochain slice et celui d'un flux H.263++ avec une synchronisation sur prochain marqueur. On introduit des erreurs de bits dans le flux codé, avec une distribution uniforme. La Figure 5 montre l'évolution du SNR moyen pour la séquence *Foreman* en configuration (70kps-10fps-4 slices par image) avec un décodage H263++ synchronisation sur prochain slice, un décodage H263++ synchronisation sur prochain marqueur et un décodage H263+ en configuration(slice,MTU=200 bits).
A partir d'un TEB[1] supérieur à $2x10^{-5}$, la synchronisation sur le prochain marqueur permet de gagner jusqu'à 4 dB (+20 %) par rapport à la séquence H.263+ et 0,70 dB par rapport à la synchronisation sur prochain slice. Pour la séquence *Man*, on obtient des résultats similaires avec un gain allant jusqu à 3 dB.

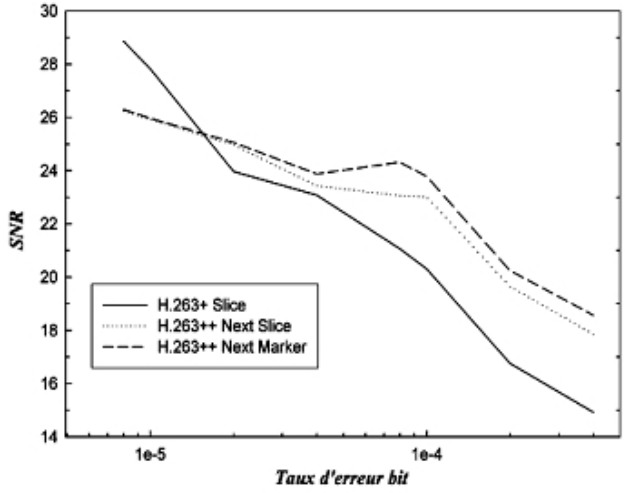

Figure 5 - *Evolution du SNR pour Foreman en fonction des stratégies de décodage*

En conclusion, la synchronisation sur prochain marqueur permet de limiter significativement l'influence des erreurs (+4dB), à condition que celles ci soient distribuées uniformément.

2.2 Influence du nombre de slices à l'encodage

Il s'agit maintenant d'étudier l'influence de l'encodage sur la robustesse du flux codé. L'un des paramètres sur lequel on peut intervenir est l'organisation de l'image en slice, et plus particulièrement, le nombre de slices par image et leur taille. Sur la Figure 6, on compare l'évolution du SNR pour la séquence *Foreman*, en fonction du nombre de slice par image et du TEB, avec une distribution d'erreur uniforme. Pour un taux d'erreur bit assez faible, la meilleure qualité est obtenue pour une configuration comportant peu de slice. Pour de tels taux, le flux codé est trop robuste et privilégie la protection au détriment de la

[1] TEB : taux d'erreur bit : représente la probabilité pour qu'un bit soit erroné dans le train binaire, on parle aussi de BER (Bit Error Rate).

qualité. Dès que le taux d'erreur bit est assez élevé, les dégradations introduites par les erreurs bits perturbent très fortement le signal codé si celui si n'est pas suffisamment protégé.

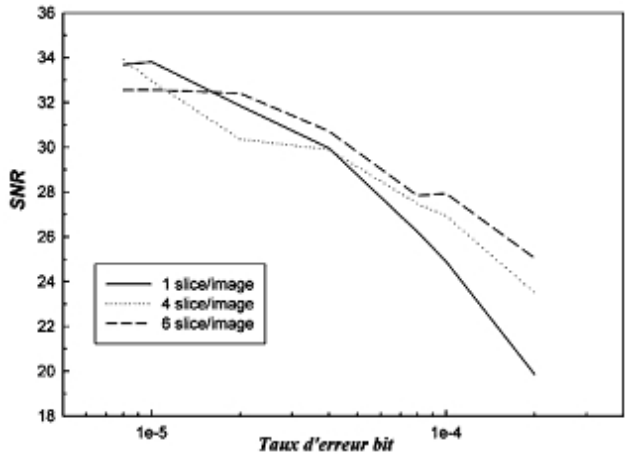

Figure 6 – *Evolution du SNR pour la séquence Foreman en fonction du nombre de slice par image et du TEB*

Ainsi, pour un taux d'erreur bit supérieur à $2x10^{-5}$, on obtient la meilleure protection avec une configuration de 6 slices par images, mais pour des taux d'erreurs inférieurs, cette protection est trop forte. **Il convient donc d'adapter dynamiquement le nombre de slices par image à l'état du réseau, ce qui ne peut se faire que si le réseau sait remonter des informations sur son état à la couche de codage source, ou si l'on dispose d'un modèle fiable de distribution des erreurs**.

2.3 Influence de la modélisation de la distribution des erreurs bits

L'un des paramètres les plus importants de cette étude est la distribution des erreurs sur le flux vidéo codé. Ce paramètre est primordial car il conditionne le réglage des paramètres au codeur, ainsi que l'efficacité du codage. Deux types de distributions ont été étudiées : une distribution par paquets d'erreurs suivant une loi gaussienne (fourni par NOKIA, résultats d'une simulation d'erreurs de bits sur un canal physique AMRC à large bande [7]) et une distribution de loi uniforme. **Ces fichiers correspondent à des erreurs sur la couche physique et ne tiennent pas compte des éventuelles corrections apportées par les couches supérieures** (comme pourrait le faire le multiplexeur H.223 pour des terminaux H.324M). Ces fichiers correspondent donc à des scénarii « dans le pire des cas » et catastrophiques.

Dans la Figure 7, on a étudié l'évolution du rapport signal sur bruit ainsi que l'écart type pour la séquence *Foreman*, codée à 70 kbps, 10 fps, configuration 4 slices. Dans la plage de TEB étudiée, le meilleur SNR moyen est obtenu pour un flux vidéo codé en H.263+. Pour des taux d'erreurs voisins de $1x10^{-6}$ bit, on retrouve une différence de SNR de l'ordre de 4 dB entre le flux H.263+ et le flux H.263++ (même différence que pour une distribution uniforme).

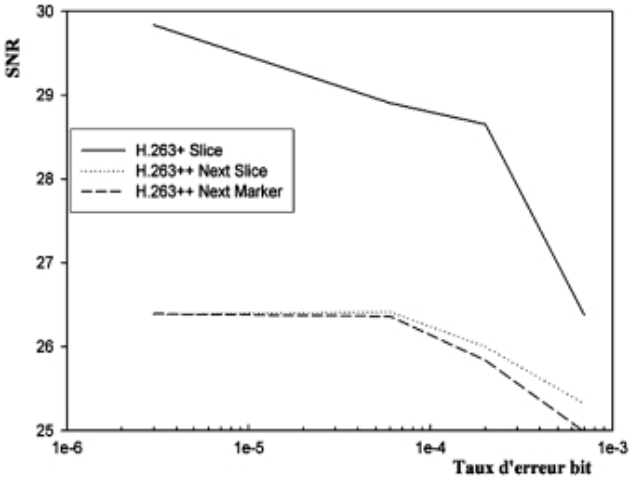

Figure 7 – *Evolution du SNR pour une distribution d'erreur gaussienne*

Donc, la protection du flux H.263++ est trop pénalisante à ce niveau de taux d'erreur. Pour un taux proche de $7x10^{-4}$, la différence entre le flux H.263++ et H.263+ n'est plus que de 1 dB, à l'avantage toujours du flux H.263+. Cependant, l'évolution de l'écart type en fonction du TEB révèle que la dégradation moyenne est beaucoup plus importante pour le flux H.263+. Pour la plage de TEB allant de $3x10^{-6}$ à $2x10^{-4}$, l'écart type reste pratiquement constant pour le flux H.263++ (proche de 1 dB), alors que pour cette même plage il croit de 1,5 à 2,7 pour le flux H.263+(Figure 8).

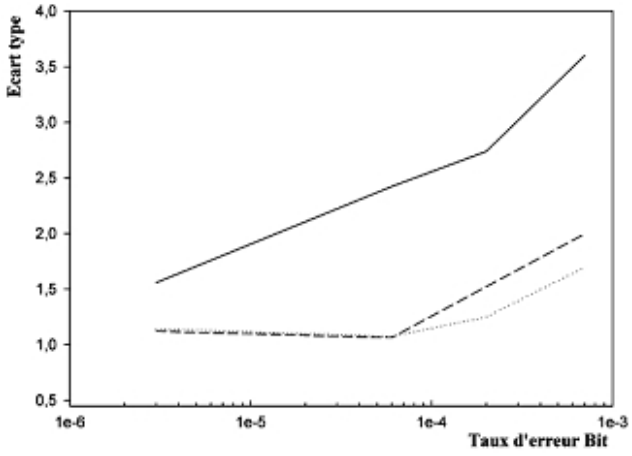

Figure 8 - *Evolution de l'écart type pour une distribution d'erreur gaussienne*

Le SNR moyen du codec H.263++ est moins important que le codec H.263+ pour une distribution gaussienne mais la qualité varie moins et on peut assurer une qualité quasi-constante sur une plus large plage de taux d'erreur bit. Enfin, les résultats expérimentaux avec

des distributions d'erreurs uniformes montrent que la synchronisation sur le prochain marqueur est plus efficace que la synchronisation sur le prochain slice. Dans le cas d'une distribution d'erreurs par paquets, on obtient de moins bons résultats avec la synchronisation sur prochain slice qu'avec la synchronisation sur prochain marqueur, et la meilleure qualité est obtenue pour les séquences codées en H.263+.

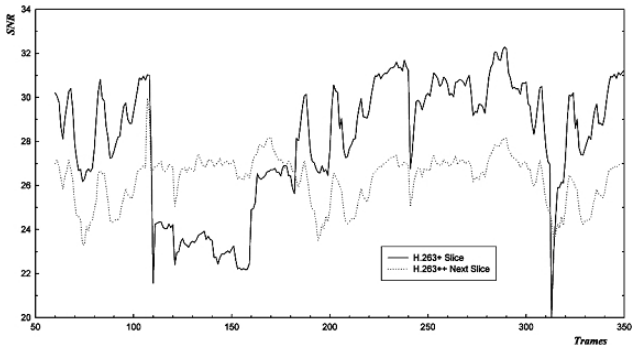

Figure 9- *Evolution du SNR d'une séquence au cours du temps en H.263+ et H.263++ : le flux codé en H.263++ a une amplitude de variation plus faible (qualité constante)*

La distribution par paquets concentre les erreurs par endroits : on a de longues séquences de trames sans erreur pour lesquelles H.263+ permet une meilleure qualité (car moins de restrictions), et une concentration d'erreur où il est difficile de récupérer les données justes. D'autre part, lorsqu'on se synchronise sur le prochain marqueur et que la concentration d'erreur est assez élevée, il y a plus de risque de se synchroniser sur des marqueurs émulés par les erreurs et donc de décoder des données erronées.
Dans le cas d'une distribution par paquets d'erreurs, le flux H.263++ est donc plus robuste aux erreurs, mais cette robustesse pénalise la qualité du flux reconstruit lorsque ces erreurs ne sont pas fréquentes (Figure 9). **Là encore, il ne s'agit pas de faire de la protection n'importe quand, mais quand l'état du canal se détériore.**

3 Conclusion

D'autres améliorations vont permettre d'accroître l'efficacité de codage H.263++. Par exemple, il serait nécessaire d'améliorer l'organisation des slices au codage. Une segmentation de l'image par groupe de vecteurs mouvement permettrait de réduire le nombre de vecteurs mouvement filtrés et donc d'augmenter l'efficacité de compression. Enfin, il faudra confronter les résultats à des expérimentations réelles sur des canaux radios. L'un des paramètres les plus importants de l'efficacité de la chaine codage-décodage-correction est la distribution des erreurs, c'est aussi celui qui est le moins bien connu. Une chaine de codage optimal contre les différents types d'erreurs devra s'adapter dynamiquement au réseau et utiliser les mécanismes de protection au bon moment afin de ne pas dégrader la qualité du flux en continu. Ce qui implique que le réseau devra savoir, à tout moment, remonter des informations sur son état. Une approche conjointe codage source-codage canal est donc nécessaire et primordiale pour tirer parti des mécanismes de correction d'erreur pour un flux codé en situation de mobilité.

Références

[1] ITU-T H.263 (02/98) Video Coding for low bit rate communication

[2] INFORMATION TECHNOLOGY – CODING OF AUDIO-VISUAL OBJECTS – Part 2: Visual ISO/IEC 14496-2:2001

[3] ITU-T H.263 (11/2000) Annex U, V, W

[4] ITU-T Appendix III : Examples for H.263 encoder/decoder implementations (06/2001), section III.5.2.5 Use of Data Partitioned Slices (Annex V)

[5] "Reversible variable length codes for efficient and robust image and video coding",Proc. IEEE Data Compression Conf., 1998, pg 471-480

[6] J.Wen et J.Villasenor, "Reversible variable lenght codes for robust image and video transmission" in Proc.iEEE 31st Asilomar Conf. Signals, Systems &Computers, vol.2 1998, pg 973-979.

[7] Simulated Wideband CDMA error Patterns for ITU-T Q15/SG16, q11ir1.doc